Application Work Group Z-Wave Specifications

Release 2024B

Z-Wave Alliance

Oct 25, 2024

Table of Contents

1 Introduction

1.1 Disclaimer

THIS SPECIFICATION IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NON-INFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE ALLIANCE, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS SPECIFICATION.

1.2 Purpose

This document specifies the Application layer of the Z-Wave and Z-Wave Long Range protocols.

1.3 Audience and Requirements

The audience of this document is the Z-Wave Alliance members and Z-Wave developers.

1.4 Terms

This document describes mandatory and optional aspects of the required compliance of a Z-Wave product to the Z-Wave standard.

The guidelines outlined in RFC 2119 with respect to key words used to indicate requirement levels are followed. Essentially, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

1.5 Terminology And Abbreviations

Terminology and abbreviations used in this document are listed in Table 1.1

Table 1.1: Terminology and Abbreviations

Term	Abbreviation	Description
Association Group Information (Command Class)	AGI	Additional information related to Association Groups
Always Listening	AL (node)	Z-Wave node that is Always Listening. Refer to [35] for details.
Always On End Node	AOEN	End nodes always powered, participating in routing.
Central Static Controller	CSC	Central Controller application Role Type.
Device Type	DT	A pre-defined set of functionalities for an application
Frequently Listening	FL (node)	Z-Wave node that is Frequently Listening. Refer to [35] for details.
Internet Protocol	IP	Internet Protocol
Least Significant Byte	LSB	Byte in the bytestream that has the lowest weight
Local Area Network	LAN	Local Area Network
Most Significant Bit	MSB	Most Significant Bit
Most Significant Byte	MSB	Byte in the bytestream that has the highest weight
Node Information Frame	NIF	Special frame indicating the capabilities of a Z-Wave node
No Operation (Command Class)	NOP	Special frame used to verify Z-Wave connectivity
Non-Listening	NL (node)	Z-Wave node that is Non-Listening. Refer to [35] for details.
Network Wide Inclusion	NWI	Inclusion method leveraging Explore NDPU to include through repeaters. Refer to [35] for details.
Personal Area Network	PAN	Personal Area Network
Portable Controller	PC	Portable Controller
Portable End Node	PEN	Portable End Node
Quick Response	QR	Format use for scannable codes
Reporting Portable Controller	RPC	Reporting Portable Controller
Reporting Sleeping End Node	RSEN	Reporting Sleeping End Node
Role Type	RT	Role Type
Security 0 Command Class	S0	Security 0 Command Class
Security 2 Command Class	S2	Security 2 Command Class
SUS NodeID Server	SIS	SUS NodeID Server
Sub Static Controller	SSC	Sub Static Controller
Static Update Controller	SUC	Static Update Controller
Transmitter	Tx	RF Transmitter
Z-Wave for IP	Z/IP	Z-Wave for IP

2 Application Command Classes

2.1 Command Class Overview

The following sections present a general Command Class overview and their associated rules. The requirements and recommendations apply to all Command Classes.

2.1.1 Overview

The Z-Wave Plus Device Type [34], Section 7 specification indicate combinations of command classes
CC:0000.00.00.11.001 which MUST be supported for a particular product class or type.

2.1.2 Command class format

2.1.2.1 Frame format

All commands classes have a common header consisting of a Command Class identifier and a Command identifier. Each command can have from zero to N bytes of command data. The bit numbering starts at zero for the least significant bit. LSB is denoted as 'Bit 0' and MSB is denoted as 'Bit 7 throughout the document.

LSB and MSB notations are also used for multi bytes fields, indicating which byte is the most significant.

Table 2.1 and Table 2.2 below show the generic command frame for the two possible formats.

Table 2.1: Generic Command format

7	6	5	4	3	2	1	0
Command Class (0x20..0xEE)							
Command							
Command Data 1							
...							
Command Data N							

Table 2.2: Generic extended Command format

7	6	5	4	3	2	1	0
Command Class MSB (0xF1..0xFF)							
Command Class LSB (0x00..0xFF)							
Command							
Command Data 1							
...							
Command Data N							

2.1.2.1.1 Command class

The Command Class identifier range is shown in Table 2.3.

Table 2.3: Command Class identifier range

Command Class	Description
0x00	No Operation. Used by Z-Wave Protocol. MAY be used by the application
0x01..0x1F	Reserved for the Z-Wave protocol
0x20..0xFF	Application Command Classes
0xEF	Support/Control mark
0xF0	Non-interoperable
0xF100	Security Mark
0xF101..0xFFFF	Extended Application Command Classes (2 byte-long Command Class identifier)

A Command Class can contain up to 255 different Commands. If the Command Class field is in the range 0xF1..0xFF, the Command Class identifier is therefore 2 bytes long. This allows for future extensions of the Command Classes and provides the possibility of having more than 4000 Command Classes.

2.1.2.1.2 Command

The command field contains the specific command indicating a node how to parse and interpret the command data. The command field length is always 1 byte.

2.1.2.1.3 Command data (N bytes)

The command data field contains data related to the command. The description for the command data is defined in each individual command.

Simple commands, such as Get commands, usually contain no command data. Other commands, such as Set or Report commands can contain several bytes of command data.

2.1.2.2 Command class versioning

All command classes have a version number. The following rules apply to avoid interoperability issues when introducing the same Command Class with different versions:

CC:0000.00.00.11.002
- A node MUST NOT discard a frame based on the length field. A receiving node MUST use the Command Class identifier and the Command identifier to interpret the application frame. This enables a device, which supports version 1 of a Command Class to interpret the version 1 part of a received version 2 of the command.

CC:0000.00.00.11.003
- All implementations of a Command Class version higher than 1 MUST initialize all parameters associated with the version higher than 1. Thereby a node implementing the version 2 of a Command Class can interpret a version 1 received Command.

CC:0000.00.00.11.004
- A device supporting a Command Class having a version higher than 1 MUST support the *Version Command Class, version 1 [OBSOLETED]* to be able to identify the supported version. If a node does not support the *Version Command Class, version 1 [OBSOLETED]* at its highest security level, it can be assumed that all Command Classes version are equal to 1.
- It is allowed to certify nodes supporting an older version of a Command Class despite a newer version exists as long as the generic/specific device specification does not require a specific version implemented.

2.1.3 Controlled and Supported Command Classes

A node can **support** and/or **control** a given Command Class.

If a Command Class is **supported**:

The node implements all the Command Class functionalities and can be set and read back by other nodes. When a Command Class is supported, it is REQUIRED to implement the whole Command Class.

CC:0000.00.00.11.005

If a Command Class is **controlled**:

The node implements the ability to interview, read and/or set other nodes supporting the Command Class. Nodes controlling Command Classes may use only a subset of the Commands within a Command Class (for example only Set commands). Even if using a Command Class partially for control, the use must comply with the Command Class requirements.

For example, a Set Command sent to Association Group destinations is a form of Command Class control.

2.1.4 Node Information Frame

The Node Information Frame (NIF) is used to inform other devices about the node capabilities. The NIF contains a structure with a protocol specific part that is handled by the Z-Wave protocol and an application specific part that is filled in by the application. The protocol specific part consists of a bit telling if the node is a continuously listening device, the Basic Device Class the node is based on etc. The application specific part consists of the Generic and Specific Device Class and the Command Classes that are supported and/or controlled by the node.

Generic and Specific Device Class are defined in [34], Section 7 for Z-Wave Plus nodes.

A NIF will be sent to the controller when a node is to be included in the network, excluded from the network or upon request.

Table 2.4, Node Information Frame Format shows the NIF structure.

Table 2.4: Node Information Frame Format

Byte descriptor / bit number	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Capability	Listening	Routing	Max Speed			Protocol Version		
Security	Opt. Func.	Sensor 1000ms	Sensor 250ms	Beam Capability	Routing End Node	Specific Device	Controller	Security
Reserved	Z-Wave Protocol Specific Part							
Basic *)	Basic Device Class (Z-Wave Protocol Specific Part)							
Generic	Generic Device Class							
Specific	Specific Device Class							
NodeInfo[0]	Command Class 1							
…	…							
NodeInfo[n-1]	Command Class N							

*) The “Basic” field is only included when the NIF is sent by a controller

The Z-Wave Protocol in a controller saves all the Node Information except the supported and controlled Command Classes when a node is included in the network. The reserved field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

CC:0000.00.00.11.006

2.1.4.1 Z-Wave Protocol Specific Part

The protocol specific part of the NIF is handled by the Z-Wave protocol. This information is automatically inserted in the packet by the protocol layer when transferring data using the API.

Basic Device Class

The Basic Device Class field contains an identifier that identifies what Basic Device Class this node is based on and is set by the Z-Wave protocol. A detailed description of all available Basic Device Classes is given in [34], Section 7 for Z-Wave Plus devices. The Z-Wave Plus devices have an additional parameter Role Type defining device role in the network. The Role Type parameter is announced via the Z-Wave Plus Info Command Class (Section 3).

CC:0000.00.00.11.007 This field is only included when the NIF frame is sent by a controller. An end node MUST omit this field. Refer to [35] for controller and end node definition.

2.1.4.2 Application Specific Part

The application specific part of the NIF is handled by the application. The information must be in accordance with the defined classes to obtain interoperability.

Listening Flag

The Listening flag is used to indicate that the node is always listening if set to 1. An always listening node must be powered continuously and reside on a fixed position in the installation. An always listening node is included in the routing table to assist as repeater in the network. The routing table is static during normal operation. In case the Listening flag is set to 0, the node is non-listening. This is typically used for battery operated nodes being asleep when the protocol is idle to prolong battery lifetime. A battery operated node is not included in the routing table and is not used as a router in the network. In some instances the node's position in the network is still determined, and stored by the protocol.

Optional Functionality Flag

The Optional Functionality flag is used to indicate that this node supports other Command Classes than the mandatory for the selected generic/specific device class and that a controlling node needs to look at the supported Command Classes to fully control this device.

Generic Device Class

The Generic Device Class field contains an identifier that identifies what Generic Device Class this node is part of and must be set by the application. For a detailed description of all available Generic Device Classes, refer to [34], Section 7 for Z-Wave Plus devices.

Specific Device Class

The Specific Device Class field specifies what Specific Device Class this application is part of and must be set by the application. For a detailed description of all available Specific Device Classes, refer to [34], Section 7 for Z-Wave Plus devices.

Command Class

CC:0000.00.00.11.008 The Command Class field is used to advertise Command Classes implemented by the node. The field MUST NOT be longer than 35 bytes.

CC:0000.00.00.11.009 The field MUST advertise the list of Command Classes that the node supports.

CC:0000.00.00.13.002 CC:0000.00.00.11.00A CC:0000.00.00.12.004 The field MAY advertise the list of Command Classes that the node can control in other nodes. If present, the list of controlling Command Classes MUST be prepended by the COMMAND_CLASS_MARK Command Class identifier. It is NOT RECOMMENDED to advertise controlled Command Classes.

CC:0000.00.00.12.001 It has been found that legacy controllers may read as little as 6 lines from this list. For backwards compatibility, the list SHOULD advertise supported command classes in the order indicated in Table 2.5, NIF :: Command Class advertising priorities :

Table 2.5: NIF::Command Class advertising priorities

Priority	Command Class
1 (First line)	COMMAND_CLASS_ZWAVEPLUS_INFO Applies only to Z-Wave Plus products
2 (if supported)	COMMAND_CLASS_SWITCH_MULTILEVEL or COMMAND_CLASS_SWITCH_BINARY
3 (if supported)	COMMAND_CLASS_SWITCH_ALL
4 (if supported)	COMMAND_CLASS_ASSOCIATION
5	All other command classes

Table 2.6, NIF::Command Class list structure shows the Command Class list structure of the Node Information frame:

Table 2.6: NIF::Command Class list structure

Description	Command Class list field content							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Non-secure Supported Command Classes	Command Class 1 *)							
	...							
	Command Class M *)							
Support / Control Mark	0xEF							
Non-secure Controlled Command Classes	Command Class 1 *)							
	...							
	Command Class K *)							

*) Command classes may be extended -> spanning two bytes for one command class

CC:0000.00.00.13.003 The Command Class list MAY start or finish with the identifier COMMAND_CLASS_MARK.

2.1.4.3 NIF and Multi Channel/Security Command Classes

CC:0000.00.00.11.00B The NIF MUST represent command classes supported without Multi Channel or Security encapsulation (Section 4).

After inclusion in a Z-Wave network, the NIF represents the node's Command Class capabilities when using no Security and no Multi Channel encapsulation.

Multi Channel Root Devices advertise their capabilities via the NIF, but they advertise the Command Class capabilities of their End Points via the Multi Channel Command Class.

Security bootstrapped nodes advertise their capabilities using security encapsulation via the Security 0 or Security 2 Command Class.

The Command Class list advertised in the NIF may vary depending on the inclusion state and S0/S2 bootstrapping state of the node.

CC:0000.00.00.11.00C A node supporting security (S0 and/or S2) MUST advertise command classes in the NIF according to Table 2.7, NIF content depending on inclusion and security bootstrapping.

Table 2.7: NIF content depending on inclusion and security bootstrapping

Network inclusion	Security bootstrapping	Command Classes advertised in the NIF
Not included in a network	N/A	At least all Command Classes that MUST always be non securely supported. All supported Command Classes MAY be advertised
Included in a network	No security supported by the included node	All supported Command Classes
Included in a network	Before the node times out waiting for Security bootstrapping	At least all Command Classes that MUST always be non securely supported. All supported Command Classes MAY be advertised
Included in a network	Timed out waiting for Security Bootstrapping (i.e. the process never started)	The same Command Class list as when security bootstrapping is successful but no key was granted *)
Included in a network	Security bootstrapping failed. (i.e. the process started but did not complete without error)	All Command Classes that MUST always be non-securely supported Command Classes that are supported when no key is granted MAY be added or removed from the list. *)
Included in a network	Bootstrapped with S0/S2	All Command Classes that MUST always be non securely supported only. (i.e. Command Classes supported even if not using S0/S2 encapsulation).

*) Refer to each individual Device Type [34], Section 7 for the list of Command Classes supported when no security key is granted

CC:0000.00.00.11.00D Command Classes advertised as supported in the NIF after S0/S2 bootstrapping MUST also be supported at higher security levels unless encapsulated outside security.

2.1.4.3.1 Examples

In the case of a node requesting Access Control, they require a certain security level before supporting their command class. For example a Door Lock Device Type will support its command classes as shown in Figure 2.1

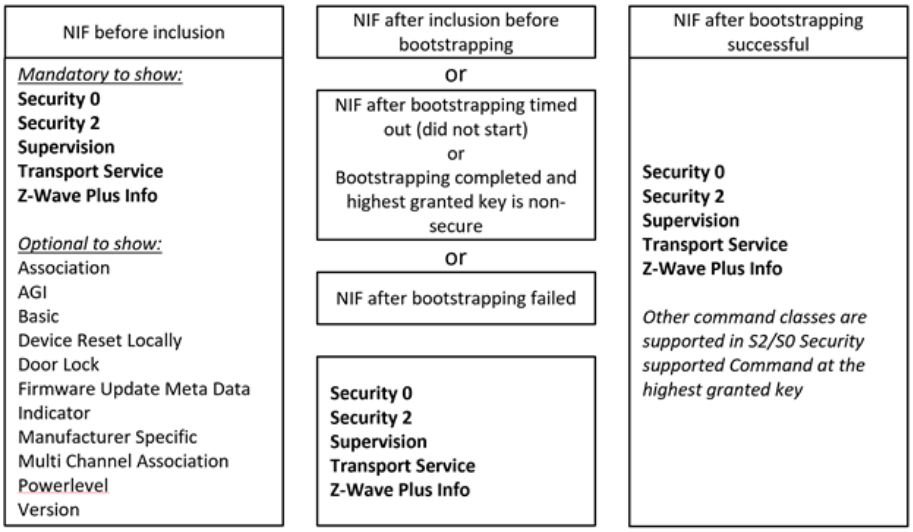


Figure 2.1: Access Control node NIF contents (always non-secure Command Classes shown in bold)

Nodes requesting a lower security level than Access Control have more freedom about advertising their regular command classes in the different scenarios. An example of a Binary Switch Device Type is shown in Figure 2.2

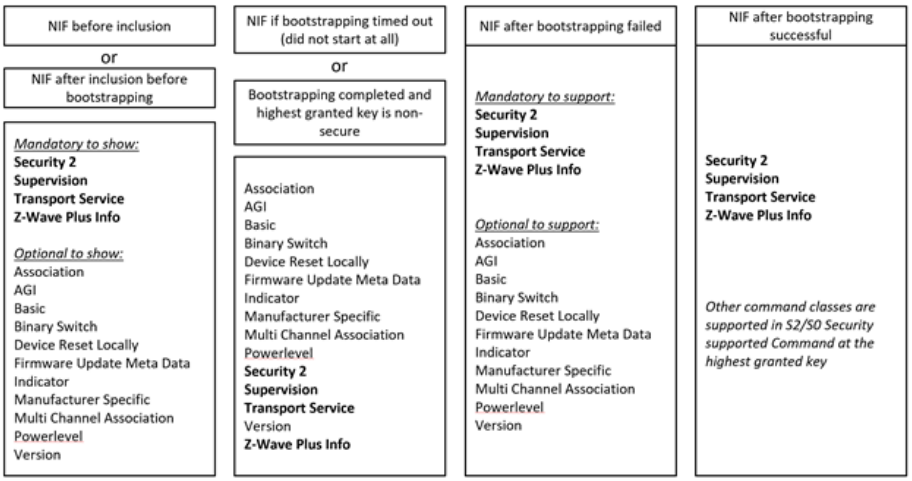


Figure 2.2: Non-Access Control node NIF contents (always non-secure Command Classes shown in bold)

2.1.4.4 Command Class specific NIF rules

CC:0000.00.00.13.004 A given Command Class MAY have additional requirement with respects to the NIF, Multi Channel Capability Report or S0/S2 Commands Supported Report.

For example, the Basic Command Class is never advertised in the NIF or the Z-Wave Plus Info is always in the NIF if supported.

In this case, requirements are listed in each individual Command Class definition, in the compatibility considerations section.

2.1.5 Multicast and broadcast commands

CC:0000.00.00.13.006 A node MAY send a command to several destinations using a Multicast or Broadcast frame. This is allowed only if the actual command does not require the responding node to return a response.

Unless specified otherwise for a particular command:

- CC:0000.00.00.11.00E Commands requiring another command to be returned in response by a receiving node MUST NOT be issued via multicast addressing in a Z-Wave network
- CC:0000.00.00.11.00F A receiving node MUST NOT return a response if a command is received via multicast addressing.

The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

2.1.6 Actuator Control

An actuator device may support one or more actuator command classes. This section presents terminology and conventions applying to all actuator command classes.

The actuator control group comprises the following command classes:

- Barrier Operator Command Class
- Basic Command Class
- Binary Switch Command Class
- Color Switch Command Class
- Door Lock Command Class
- Multilevel Switch Command Class
- Simple AV Command Class
- Sound Switch Command Class
- Thermostat Setpoint Command Class
- Thermostat Mode Command Class
- Window Covering Command Class

2.1.6.1 Terminology

Actuators may be controlled in two possible manners.

A **position actuator** responds to a **position control command**, which specifies a **target value** and optionally a **duration** for the **transition** from the **current value** to the target value. The value may be **binary** or **multilevel**.

State control commands may be encoded as special multilevel position control values, e.g. 0xFF for “On” and 0x00 for “Off”. The most recent (non-zero) target value is **restored** in response to the “On” state control command.

A **motion actuator** responds to start/stop commands. A **motion control command** specifies a **direction**, optionally a start value, and a **transition rate**.

2.1.6.2 Reporting values

A position control command may be used to initiate a transition to a new target value. A position report advertises the current value of the device hardware, optionally the target value and the remaining transition duration. If the transition is initiated by a motion control command, the reported target value is the min or max value and the duration is the time needed to reach the target value at the actual transition rate.

A controlling device should not assume that the position value advertised in a Report is identical to a value previously issued with a position control command when a transition has ended.

CC:0000.00.00.53.002 A controlling node MAY want to receive application-level acknowledgements after issuing actuating commands. It is the responsibility and choice of the controlling node whether or not to ensure that
CC:0000.00.00.52.001 a actuating command has been successfully executed. If doing so, it is RECOMMENDED to use the *Supervision Command Class, version 1* or MAY read back the node's state at a later time with a Get Command.

CC:0000.00.00.12.005 A supporting node receiving an actuating Command SHOULD NOT return a Report Command advertising its status or level unless specified in the Command Class specification.

2.1.6.3 Command values vs. hardware values

A device may implement fewer hardware levels than supported by a position control Command Class. The hardware levels should be distributed uniformly over the entire range. The mapping of command
CC:0000.00.00.11.010 values to hardware levels must be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level. An example is found in Table 2.8, Mapping command values to hardware levels (example).

Table 2.8: Mapping command values to hardware levels (example)

Multilevel Value	Hardware level	State
0	0	Off
1..33	33%	On
34..66	66%	On
67..99	100%	On

2.1.6.4 Supporting multiple actuator Command Classes

CC:0000.00.00.13.007 Several actuator Command Classes MAY co-exist in a node or Multi Channel End Point. For example,
CC:0000.00.00.13.008 a node MAY support both Window Covering Command Class and Multilevel Switch Command Class.

CC:0000.00.00.12.002 In this case, the two actuator Command Classes SHOULD actuate the same resource. If a node
CC:0000.00.00.12.003 implements several actuating resources, e.g. a LED and a power switch, it is RECOMMENDED to use Multi Channel End Points for each individual resource.

2.1.7 Common fields and encoding

2.1.7.1 Reserved and Res fields

CC:0000.00.00.11.011 Fields named 'Reserved' or 'Res' MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.1.7.2 Reserved values and reserved bits

CC:0000.00.00.11.012 Values of fields in commands that are marked as “reserved” or “res” MUST NOT be used by devices sending commands and MUST be ignored by devices receiving commands.

CC:0000.00.00.11.013 Bits in commands that are marked as “reserved” or “res” MUST be set to 0 by devices sending commands and MUST be ignored by devices receiving commands.

CC:0000.00.00.11.014 Undefined values for a given parameter MUST be treated as “reserved”. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

2.1.7.3 Duration encoding

CC:0000.00.00.11.015 Some actuator command classes allow controlling nodes to specify a duration for reaching the target value. The duration MUST be encoded according to Table 2.9, Actuator Command Class Duration Set encoding.

Table 2.9: Actuator Command Class Duration Set encoding

Value	Description
0x00	Instantly
0x01..0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80..0xFE	1 minute (0x80) to 127 minutes (0xFE) in 1 minute resolution.
0xFF	Factory default duration.

CC:0000.00.00.11.016 A node supporting an actuator Command Class may advertise the duration left to reach the target value advertised in a report. The duration MUST be encoded according to Table 2.10, Actuator Command Class Duration Report encoding.

Table 2.10: Actuator Command Class Duration Report encoding

Value	Description
0x00	0 seconds. Already at the Target Value.
0x01..0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80..0xFD	1 minute (0x80) to 126 minutes (0xFD) in 1 minute resolution.
0xFE	Unknown duration
0xFF	Reserved

2.1.7.4 Unsigned encoding

CC:0000.00.00.11.017 Unless specified otherwise in a field description, the field encoding is using unsigned representation. Fields using unsigned encoding MUST comply with Table 2.11, Default unsigned encoding examples

Table 2.11: Default unsigned encoding examples

Value (hex)	**unsigned 8-bit representation (decimal)**		**Value (hex)**	**Unsigned 16-bit representation (decimal)**		**Value (hex)**	**Unsigned 32-bit representation (decimal)**
0x00	0		0x0000	0		0x00000000	0
0x01	1		0x0001	1		0x00000001	1
0x02	2		0x0002	2		0x00000002	2
0x7F	127		0x7FFF	32767		0x7FFFFFFF	2147483647
0x80	128		0x8000	32768		0x80000000	2147483648
0xFE	254		0xFFFE	65534		0xFFFFFFFE	4294967294
0xFF	255		0xFFFF	65535		0xFFFFFFFF	4294967295

2.1.7.5 Signed encoding

Some fields use the signed encoding representation.Signed encoding is using the representation. Fields using signed encoding MUST comply with Table 2.12.

Table 2.12: Signed field encoding (two's complement representation)

Value	Signed 8-bit representation		Value	Signed 16-bit representation		Value	Signed 32-bit representation
0x7F	127		0x7FFF	32767		0x7FFFFFFF	2147483647
0x02	2		0x0002	2		0x00000002	2
0x01	1		0x0001	1		0x00000001	1
0x00	0		0x0000	0		0x00000000	0
0xFF	-1		0xFFFF	-1		0xFFFFFFFF	-1
0xFE	-2		0xFFFE	-2		0xFFFFFFFE	-2
0x80	-128		0x8000	-32768		0x80000000	-2147483648

2.1.7.6 Fields values and version

CC:0000.00.00.11.019 New Command Class versions may define new possible values for a field introduced previously. In that case, a version column is added to the value table. A supporting node MUST support the values associated its version and previous versions and MUST NOT support values associated to future versions. An example is given in Table 2.13 *Field values and Command Class version (example)*.

Table 2.13: Field values and Command Class version (example)

Value	Description	Version
0x00	Value introduced in version 1, valid and mandatory in all versions	1
0x01	Value introduced in version 2, valid and mandatory in version 2 and newer. Version 1 MUST consider this value as reserved	2
0x02	Value introduced in version 2, valid and mandatory in version 2 and newer. Version 1 MUST consider this value as reserved	2
0xFF	Value introduced in version 3, valid and mandatory in version 3 and newer. Version 1 and 2 MUST consider this value as reserved	3

2.2 Command Class Definitions

2.2.1 Alarm Command Class, version 1 [DEPRECATED]

> **Warning: THIS COMMAND CLASS VERSION HAS BEEN DEPRECATED**
>
> A device MAY implement this Command Class version, but it is RECOMMENDED that new implementations comply with Notification Command Class, version 8

CC:0071.01.00.12.001

The Alarm Command Class allows applications to report alarm or service conditions. Since these
CC:0071.01.00.11.001 parameters are not standardized across devices the alarms/service parameters MUST be described in the user manual (or an installer manual).

2.2.1.1 Interoperability considerations

The Alarm Command Class has been superseded by the Notification Command Class. Refer to most recent version of Notification Command Class.

2.2.1.2 Alarm Get Command

This command is used to get the value of an alarm.

CC:0071.01.04.11.001 The Alarm Report Command MUST be returned in response to this command if the alarm type is supported.

CC:0071.01.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0071.01.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.14: Alarm Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM (0x71)							
Command = ALARM_GET (0x04)							
Alarm Type							

Alarm Type (8 bits)

The Alarm Type field specifies which alarm is being requested. The alarm types are specific for each application.

2.2.1.3 Alarm Report Command

This command is used to report the type and level of an alarm.

Table 2.15: Alarm Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM (0x71)							
Command = ALARM_REPORT (0x05)							
Alarm Type							
Alarm Level							

Alarm Type (8 bits)

Refer to explanation under the *Alarm Get Command*.

Alarm Level (8 bits)

The alarm level is application specific.

2.2.2 Alarm Command Class, version 2 [DEPRECATED]

Warning: THIS COMMAND CLASS VERSION HAS BEEN DEPRECATED A device MAY implement this Command Class version, but it is RECOMMENDED that new implementations comply with Notification Command Class, version 8

CC:0071.02.00.12.001

The Alarm Command Class is intended for Z-Wave enabled devices capable of reporting alarm reports.

Version 2 of the Alarm Command Class is improved with the following functionalities:

- Alarm Types defined by the Z-Wave Alliance
- Interview process of supported Alarm Types

The commands not described in this version remain unchanged from *Alarm Command Class, version 1 [DEPRECATED]*.

2.2.2.1 Interoperability considerations

The interoperability considerations from version 1 also apply for this version. Refer to Section 2.2.1.1.

2.2.2.2 Alarm Set Command

This command is used to set the activity of the Z-Wave Alarm Type and Status.

Table 2.16: Alarm Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM (0x71)							
Command = ALARM_SET (0x06)							
Z-Wave Alarm Type							
Z-Wave Alarm Status							

Z-Wave Alarm Type (8 bits)

Refer to Section 2.2.1.3 Alarm Report Command

Z-Wave Alarm Status (8 bits)

This field is used to set the state of the Alarm Type. The value 0x00 will deactivate the alarm and 0xFF will activate the alarm i.e. unsolicited Alarm Report Command will be transmitted to the node(s) defined in the Node field(s) when triggered by an event. Any other value is reserved for future use.

CC:0071.02.06.11.001 **Note**: The factory default state MUST be described in the product manual. All Z-Wave enabled
CC:0071.02.06.11.002 devices MUST be able to operate based on factory default settings i.e. an end-user MUST NOT be forced to set-up the states of the device in order to operate. The factory default state of the Z-Wave
CC:0071.02.06.12.001 Alarm Status SHOULD be enabled.

CC:0071.02.06.11.003 Products that do not allow deactivation of a specific Alarm Type, MUST respond to a Alarm Configuration Set deactivating the Alarm Type in question by returning an Application Rejected Request Command of the Application Status Command Class.

2.2.2.3 Alarm Get Command

The Alarm Get Command is used to request the alarm state for a specific alarm type announced as supported through the Alarm Type Supported Report Command.

CC:0071.02.04.11.001 The Alarm Report Command MUST be returned in response to this command if the alarm type is supported.

CC:0071.02.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0071.02.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.17: Alarm Get Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM (0x71)							
Command = ALARM_GET (0x04)							
Alarm Type							
Z-Wave Alarm Type							

Alarm Type (8 bits)

This field refers to the Alarm Type of Alarm Command Class (Version 1) i.e. the application specific Alarm Type which is not defined by the Z-Wave Alliance. If the 'V1 Alarm' field is set to '0' as reported
CC:0071.02.04.11.004 via the Alarm Type Supported Report Command, this field MUST be set to '0' when requesting the report.

Z-Wave Alarm Type (8 bits)

CC:0071.02.04.11.005 The Z-Wave Alarm Type field MUST contain the Alarm Type identifier described in Alarm Report Command. This parameter refers to the Alarm Types defined by the Z-Wave Alliance.

CC:0071.02.04.11.006 A node receiving a non-supported Z-Wave Alarm Type MUST ignore the command. A controlling
CC:0071.02.04.12.001 node SHOULD interview the device for supported Alarm Types by means of Alarm Type Supported Get Command prior to Alarm Get.

2.2.2.4 Alarm Report Command

This command is used by the application to report the alarm state.

Table 2.18: Alarm Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM (0x71)							
Command = ALARM_REPORT (0x05)							
Alarm Type							
Alarm Level							
Zensor Net Source Node ID							
Z-Wave Alarm Status							
Z-Wave Alarm Type							
Z-Wave Alarm Event							
Number of Event Parameters							
Event Parameter 1							
...							
Event Parameter N							

Alarm Type (8 bits)

Refer to Section 2.2.2.4 Alarm Report Command

Alarm Level (8 bits)

Refer to Section 2.2.2.4 Alarm Report Command

Zensor Net Source Node ID (8 bits)

Specify the Zensor Net Source Node ID, which detected the alarm condition. In Zensor Net it is not possible to determine the Source Node ID due to the broadcast forwarded frame is without this
CC:0071.02.05.11.001 information on protocol level. If the device is not based on Zensor Net this field MUST be set to '0'.

Z-Wave Alarm Status (8 bits)

Refer to Section 2.2.2.2 Alarm Set Command

Number of Event Parameters (8 bits)

Indicates the Number of Event Parameters fields used in bytes.

Z-Wave Alarm Type (8 bits), Z-Wave Alarm Event (8 bits) and Event Parameters (N Byte)

[36] specifies the Alarm Types and its subordinate parameters defined by the Z-Wave Alliance. The
CC:0071.02.05.11.002 fields that do not contain any definition of the Z-Wave Alarm Type MUST be set to '0' in the Alarm Report Command.

CC:0071.02.05.11.003 The device MUST advertise support of the Command Class which is included for the specific Alarm Type. Example: for Smoke Alarm, Smoke Detected the Node Naming and Location Command Class MUST be advertised as supported in the Node Information Frame.

Alarm Type = 0xFF is used by the Alarm Get Command to retrieve the first alarm detection from the list of pending alarms.

Example: a device supports the Z-Wave Alarm Types: Smoke, CO2 and Heat. The Heat Alarm is active e.g. overheat has been detected. When the device receives Alarm Get, Z-Wave Alarm Type (0xFF), it must return Alarm Report, Z-Wave Alarm Type (0x04), Z-Wave Alarm Event (0x01/0x02) and the accompanied parameters.

2.2.2.5 Alarm Type Supported Get Command

This command is used to request the supported alarm types.

CC:0071.02.07.11.001 The Alarm Type Supported Report Command MUST be returned in response to an Alarm Type Supported Get command.

CC:0071.02.07.11.002 This command MUST NOT be issued via multicast addressing.

CC:0071.02.07.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.19: Alarm Type Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM (0x71)							
Command = ALARM_TYPE_SUPPORTED_GET (0x07)							

2.2.2.6 Alarm Type Supported Report Command

This command is used to advertise the supported alarm types in the application.

Table 2.20: Alarm Type Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM (0x71)							
Command = ALARM_TYPE_SUPPORTED_REPORT (0x08)							
V1 Alarm	Reserved		Number of Bit Masks				
Bit Mask 1							
...							
Bit Mask N							

V1 Alarm (1 bit)

0 = the device implements only Notification CC V2 (or newer) Notification Type(s).

1 = the device implements Notification CC V2 Notification Types as well as proprietary Alarm CC V1 Alarm Types and Alarm Levels.

Reserved

CC:0071.02.08.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Bit Masks (5 bits)

Indicates the Number of Bit Masks fields used in bytes.

Bit Mask (N Bytes)

The Bit Mask fields describe the supported **Z-Wave Alarm** Types by the device.

CC:0071.02.08.11.002

- Bit 0 in Bit Mask 1 is not allocated to any Z-Wave Alarm Type and MUST therefore be set to zero.
- Bit 1 in Bit Mask 1 indicates if Z-Wave Alarm Type = 1 (Smoke) is supported.
- Bit 2 in Bit Mask 1 indicates if Z-Wave Alarm Type = 2 (CO) is supported.
- Bit 3 in Bit Mask 1 indicates if Z-Wave Alarm Type = 3 (CO2) is supported
- ...

CC:0071.02.08.11.003 If the Z-Wave Alarm Type is supported the corresponding bit MUST be set to 1. If the Z-Wave Alarm Type is not supported the corresponding bit MUST be set to 0.

CC:0071.02.08.11.004 Z-Wave Alarm Type = 0xFF (Return first Alarm on supported list) MUST NOT be advertised in the Bit Masks.

CC:0071.02.08.11.005 The number of Bit Mask fields MUST match the value advertised in the Number of Bit Masks field.

Note that the mapping of bit 1 to Alarm Type =1 differs from the support mapping used by the Multilevel Sensor Command Class. The Multilevel Sensor Command Class maps bit 0 to Sensor Type = 1.

2.2.3 Alarm Sensor Command Class, version 1 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Notification Command Class.

If implementing this Command Class, it is RECOMMENDED that the Notification Command Class is also implemented.

The Alarm Sensor Command Class is used to realize Sensor Alarms.

2.2.3.1 Alarm Sensor Get Command

This command is used to request the status of a sensor.

The Alarm Sensor Report Command MUST be returned in response to this command if the sensor type is supported.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.21: Alarm Sensor Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM (0x9C)							
Command = SENSOR_ALARM_GET (0x01)							
Sensor Type							

Sensor Type (8 bits)

Sensor Type specifies what type of sensor this command originates from. Refer to Table 2.22. The sensor type value 0xFF returns the first found supported sensor type in the bit mask (starting from bit 0 in Bit Mask 1) by the Alarm Sensor Supported Report.

Table 2.22: Alarm Sensor Get::Sensor Type encoding

Value	Sensor Type
0x00	General Purpose Alarm
0x01	Smoke Alarm
0x02	CO Alarm
0x03	CO2 Alarm
0x04	Heat Alarm
0x05	Water Leak Alarm
0xFF	Return first Alarm on supported list

2.2.3.2 Alarm Sensor Report Command

This command is used to advertise the alarm state.

Table 2.23: Alarm Sensor Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM (0x9C)							
Command = SENSOR_ALARM_REPORT (0x02)							
Source Node ID							
Sensor Type							
Sensor State							
Seconds 1 (MSB)							
Seconds 2 (LSB)							

Source Node ID (8 bits)

Specify the source node ID, which detected the alarm condition. In a Zensor Net is it not possible to determine the source node ID because the frame is broadcast forwarded without this information on protocol level.

Sensor Type (8 bits)

Refer to Section 2.2.3.1 Alarm Sensor Get Command.

The Sensor Type 0xFF MUST NOT be advertised in this command.

Sensor State (8 bits)

The Sensor State parameter returns the current alarm state. The value 0x00 indicates no alarm and 0xFF indicates alarm. Furthermore it can return values from 0x01 to 0x64 to indicate severity of the alarm in percentage.

The values 0x65…0xFE are reserved and MUST be ignored by receiving nodes.

Seconds (16 bits)

The field Seconds indicates time the remote alarm must be active since last received report. The value 0x0000 indicates that the time field MUST be ignored.

2.2.3.3 Alarm Sensor Supported Get Command

This command is used to request the supported sensor types from the device.

The Alarm Sensor Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.24: Alarm Sensor Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM (0x9C)							
Command = SENSOR_ALARM_SUPPORTED_GET (0x03)							

2.2.3.4 Alarm Sensor Supported Report Command

This command is used to report the supported sensor types from the device.

Table 2.25: Alarm Sensor Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM (0x9C)							
Command = SENSOR_ALARM_SUPPORTED_REPORT (0x04)							
Number of Bit Masks							
Bit Mask 1							
...							
Bit Mask N							

Number of Bit Masks (8 bits)

Indicates the Number of Bit Masks fields used in bytes.

Bit Mask (N Bytes)

The Bit Mask fields describe the supported sensor types by the device.

- Bit 0 in Bit Mask 1 indicates if Sensor Type = 0 (General Alarm) is supported.
- Bit 1 in Bit Mask 1 indicates if Sensor Type = 1 (Smoke Alarm) is supported.
- ...

The sensor type is supported if the bit is 1 and the opposite if 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported sensor type. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

Note that the mapping of bit 1 to Sensor Type =1 differs from the support mapping used by the Multilevel Sensor Command Class. The Multilevel Sensor Command Class maps bit 0 to Sensor Type = 1.

2.2.4 Alarm Silence Command Class, version 1

The Alarm Silence Command Class may be used to temporarily disable the sounding of the alarm but still keep the alarm operating.

2.2.4.1 Alarm Silence Set Command

This command is used to remotely silence the sensor alarm.

Table 2.26: Alarm Silence Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SILENCE_ALARM (0x9D)							
Command = SENSOR_ALARM_SET (0x01)							
Mode							
Seconds 1 (MSB)							
Seconds 2 (LSB)							
Number of Bit Masks							
Bit Mask 1							
...							
Bit Mask N							

Mode (8 bits)

Mode specifies the different options to silence sensor alarms. Modes are defined by the Z-Wave Alliance.

Table 2.27: Alarm Silence Set Command Modes

Value	Mode
0x00	Disable sounding of all sensor alarms independent of bit mask
0x01	Disable sounding of all sensor alarms independent of bit mask which have received the alarm via the Sensor Alarm Report command
0x02	Disable sounding of all sensor alarms according to bit mask
0x03	Disable sounding of all sensor alarms according to bit mask which have received the alarm via the Alarm Sensor Report Command

Seconds (16 bits)

The field Seconds indicates the duration sounding of the alarm must be disable but still keep the alarm operating. If silence is engaged, the alarm will come back on when the duration expires unless the originating sensor clears the alarm. The value 0x0000 indicates that the time field MUST be ignored.

Number of Bit Masks (8 bits)

Indicates the Number of Bit Masks fields used in bytes.

Bit Mask (N Bytes)

The Bit Mask fields describe the sensor types to disable sounding from.

- Bit 0 in Bit Mask 1 indicates if Sensor Type = 0 (General Alarm) is disabled.
- Bit 1 in Bit Mask 1 indicates if Sensor Type = 1 (Smoke Alarm) is disabled.
- ...

If the sensor type is disabled the bit MUST be set to 1. If the sensor type is not disabled the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last sensor type to be disabled.

2.2.5 All Switch Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED New implementations MUST NOT support this Command Class. Controlling nodes MUST use S2 Multicast mechanism in order to achieve the All On and All Off functionality

The All Switch Command Class is used to switch all devices on or off. Devices may be excluded/included from the all on/all off functionality. The application determines which devices there are included in the all on/all off functionality as default.

2.2.5.1 All Switch Set Command

This command is used to instruct a device if it is included or excluded from the all on/all off functionality.

Table 2.28: All Switch Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL (0x27)							
Command = SWITCH_ALL_SET (0x01)							
Mode							

Mode (8 bits)

The mode field used to set the all on/all off functionality of the device.

Table 2.29: All Switch Set Command Modes

Mode	Description
0x00	Indicate that the switch is excluded from the all on/all off functionality.
0x01	Indicate that the switch is excluded from the all on functionality but not all off.
0x02	Indicate that the switch is excluded from the all off functionality but not all on.
0x03..0xFE	Reserved
0xFF	Indicates that the switch is included in the all on/all off functionality.

2.2.5.2 All Switch Get Command

This command is used to ask a device if it is included or excluded from the all on/all off functionality.

The All Switch Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.30: All Switch Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL (0x27)							
Command = SWITCH_ALL_GET (0x02)							

2.2.5.3 All Switch Report Command

This command is used to report if the device is included or excluded from the all on/all off functionality.

Table 2.31: All Switch Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL (0x27)							
Command = SWITCH_ALL_REPORT (0x03)							
Mode							

Mode (8 bits)

Refer to the *All Switch Set Command*.

2.2.5.4 All Switch On Command

This command is used to inform a switch that it SHOULD be turned on. A receiving device MUST NOT react to this command if the actual operation has been prohibited via the Switch All Set command. Like the Basic Set On command, this command MUST cause the device to restore the most recent (non-zero) level.

Table 2.32: All Switch On Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL (0x27)							
Command = SWITCH_ALL_ON (0x04)							

2.2.5.5 All Switch Off Command

This command is used to inform a switch that it SHOULD be turned off. A receiving device MUST NOT react to this command if the actual operation has been prohibited via the Switch All Set command.

Table 2.33: All Switch Off Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL (0x27)							
Command = SWITCH_ALL_OFF (0x05)							

2.2.6 Anti-theft Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS VERSION HAS BEEN OBSOLETED New implementations MUST use the *Anti-theft Command Class, version 2 [DEPRECATED]* or newer.

The Anti-theft Command Class is used to disable a subset of supported/controlled command classes in a device if the device is being excluded and re-included into a Z-Wave network again. This command class is typically used when installing a Z-Wave device in a public location such as a hotel room or conference center. The command class allows the user to lock the device to the actual Z-Wave network and to render it useless if it is removed from the local network without being unlocked. Another application would be to protect service provider owned products from leaving the service providers network before they are paid for.

Version 2 limits the Magic Code and Anti-theft Hint maximum bytes to 10. This makes it possible to embed Anti-theft Command Class Version 2 in one Security Command Class and thereby avoid splitting it.

2.2.7 Anti-theft Command Class, version 2 [DEPRECATED]

Warning: THIS COMMAND CLASS VERSION HAS BEEN DEPRECATED

New implementations SHOULD use the *Anti-theft Command Class, version 3*.

The Anti-theft Command Class MUST NOT be supported unless the Device Class or Device Type implemented by the device explicitly allows for support of the Anti-theft Command Class.

The Anti-theft Command Class is intended for devices operating in public locations such as hotel rooms or a conference center. The purpose of the Anti-theft Command Class is to render a device useless if it is removed from its actual network without being unlocked by the owner or a service provider.

The Anti-theft Command Class is used to disable all command classes related to the actual application functionality of a device if is it excluded and later included in another network. It does not matter if the device implements a single resource addressed via the Root Device or a collection of resources addressed via individual Multi Channel End Points Enabling anti-theft protection in a device MUST NOT change any operation with respect to supported/controlled command classes as long as the device stays in the actual network.

If a locked device is excluded, it MUST enter the protected state. When in the protected state, the node information frame (NIF) MUST NOT advertise support of the protected command classes.The NIF MUST however continue advertising support of the Anti-theft Command Class and all other non-application specific command classes; just as when the device operates in its home network.

The device MUST NOT respond to application commands while in the protected state. A device in protected state MUST NOT leave its protected state if it is re-included into its home network. Another Anti-Theft Set command MUST be used to either disable locking or to clear the protected state.

The following non-device specific command classes must not be protected by the anti-theft functionality (i.e. will always be available in the device regardless of protection state):

- Manufacturer Specific Command Class
- Version Command Class
- Anti-theft Command Class
- Security Command Class (optional)

  Security encapsulated command classes are allowed to be protected. In that case they must be removed from the *Security Commands Supported Report Command* when in the protected state in a foreign network.

The protection state may be updated by sending the set command with the correct magic code to the device at any time and in any network. When the protection state is updated the device must return to normal operation, regardless of whether the update is to disable or re-enable protection. However, it is not possible to update protection state when device is excluded because it must be able to receive a command.

The Anti-theft protection state must be preserved in the following situations:

- Exclusion of a network
- Reset to factory default
- OTA update of firmware

If secure device, supports Anti-theft Command Class, Security Command Class needs to be supported regardless of anti-theft protection state. A security enabled device MUST be able to join any secure Z-Wave network regardless of its anti-theft protection state.

It is RECOMMENDED for a device that supports anti-theft protection to have physical mark that indicates that this device is capable of being locked. It is further RECOMMENDED that the device is capable of signaling via a LED or other means if the device refuses inclusion in a network because the

device is locked to another network. Finally, the user guide and installation manual MUST advertise support of anti-theft protection.

2.2.7.1 Anti-theft Set Command

This command is used to enable/disable anti-theft protection in a device already included into a Z-Wave Network by sending a magic code to device in question. The same magic code MUST be used to disable anti-theft protection again. A new magic code may be used the next time to enable anti-theft protection in the device, but only if protection is disabled at the time. A new device MUST have anti-theft protection disabled. Enabling anti-theft protection in an already-enabled device restores it to normal operation if it is in reduced functionality mode, but otherwise has no effect.

Table 2.34: Anti-theft Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT (0x5D)							
Command = ANTITHEFT_SET (0x01)							
Enable	Number of Magic Code bytes						
Magic Code 1							
...							
Magic Code N							
Manufacturer ID MSB							
Manufacturer ID LSB							
Anti-theft Hint Number Bytes							
Anti-theft Hint Byte 1							
...							
Anti-theft Hint Byte N							

Enable (1 bit)

The value MAY be 0 (Attempt to disable anti-theft protection in device) or 1 (Attempt to enable or re-enable anti-theft protection in device). It is not necessary to first disable an exclude device having protection enable; it can be re-enabled directly in a new network by using correct magic code again.

Number of Magic Code bytes (7 bits)

Indicates the Number of Magic Code fields N used in bytes. Maximum number of Magic Code fields MUST NOT exceed 10 bytes.

Magic Code (N bytes)

The Magic Code fields hold the code to enable/disable the Z-Wave device in question.

Manufacturer ID (2 bytes)

The Manufacturer ID of the company's product having a central role in the application requiring anti-theft protection enabled. Device should report 0xFFFF if anti-theft protection is disabled.

Manufacturer identifiers can be found in [32].

Number of Anti-theft Hint bytes (8 bits)

Indicates the Number of Anti-theft Hint fields N used in bytes. If length is 0 no Hint provided. Maximum number of Anti-theft Hint fields MUST NOT exceed 10 bytes.

Anti-theft Hint Byte (N bytes)

Anti-theft Hint Bytes that may be used as an identifier or key value for retrieving the Magic Code. The exact format and meaning of these Bytes is specific to the product or service that enabled anti-theft protection on the device, as identified by the Manufacturer ID above. If it is necessary to render the Hint Bytes for display, each byte should be interpreted as an unsigned integer value and represented in hexadecimal.

2.2.7.2 Anti-theft Get Command

This command is used to get an Anti-theft Report Command showing status of the Z-Wave device in question.

The Anti-theft Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.35: Anti-theft Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT (0x5D)							
Command = ANTITHEFT_GET (0x02)							

2.2.7.3 Anti-theft Report Command

This command is used to report status of the Z-Wave device in question.

Table 2.36: Anti-theft Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT (0x5D)							
Command = ANTITHEFT_REPORT (0x03)							
Anti-theft Protection Status							
Manufacturer ID MSB							
Manufacturer ID LSB							
Anti-theft Hint Number Bytes							
Anti-theft Hint Byte 1							
...							
Anti-theft Hint Byte N							

Anti-theft Protection Status (8 bits)

Anti-theft Protection Status specifies the actual status of Z-Wave device in question. Refer to the table below with respect to defined status values.

Table 2.37: Anti-theft Protection Status

Value	Anti-theft Protection Status
0x00	Reserved.
0x01	Anti-Theft Protection is currently DISABLED, and the Z-Wave Device is fully functional.
0x02	Anti-Theft Protection is currently ENABLED, and the Z-Wave Device is fully functional.
0x03	Anti-Theft Protection is currently ENABLED, and the Z-Wave Device is NOT fully functional (i.e., the Device was excluded from a network without disabling protection, and an ANTITHEFT_SET command with the correct Magic Code has not yet been received in the current network).
0x04..0xFF	Reserved

Manufacturer ID (2 bytes)

The Manufacturer ID of the company's product having a central role in the application requiring anti-theft protection enabled. Device should report 0xFFFF if anti-theft protection is disabled.

Number of Anti-theft Hint bytes (8 bits)

Indicates the Number of Anti-theft Hint fields N used in bytes. If length is 0 no Hint provided. Maximum number of Anti-theft Hint fields MUST NOT exceed 10 bytes.

Anti-theft Hint Byte (N bytes)

Anti-theft Hint Bytes. See the Anti-Theft Set Command for more details.

2.2.7.4 Examples

Following examples are for reference only.

2.2.7.4.1 Example of a non-secure Thermostat

Below is shown an example of the Node Information Frame (NIF) content for a non-secure thermostat. The first NIF shows a device having anti-theft protection disabled. The device may never been included into a network or reside in a network or excluded from a network:

Table 2.38: Disabled Anti-theft Protection

Disabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Anti-theft Command Class
Thermostat Operating State Command Class
Thermostat Mode Command Class
Association Command Class
Battery Command Class

The second NIF shows a device having anti-theft protection enabled. The device may be excluded from network in which it was originally anti-theft protection enabled or re-included into a network. This also applies in case device is re-included into the network, which device originally was anti-theft protection enabled:

Table 2.39: Enabled Anti-theft Protection

Enabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Anti-theft Command Class

2.2.7.4.2 Example of a security enabled Thermostat

Below is shown an example of the Node Information Frame (NIF) content for a security enabled thermostat. The first NIF shows a device having anti-theft protection disabled. The device may never been included into a network or reside in a network or excluded from a network:

Table 2.40: Disabled Anti-theft Protection, example 2

Disabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Security Command Class

Finally, the *Security Commands Supported Report Command* reports support of the following command classes:

- Anti-theft Command Class
- Thermostat Operating State Command Class
- Thermostat Mode Command Class
- Association Command Class
- Battery Command Class

The second NIF shows a device having anti-theft protection enabled. The device may be excluded from network in which it was originally anti-theft protection enabled or re-included into a network. This also applies in case device is re-included into the network, which device originally was anti-theft protection enabled. The NIF is unchanged because all application oriented command classes are security encapsulated except the default command classes:

Table 2.41: Enabled Anti-theft Protection, example 2

Enabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Security Command Class

Finally, the Security Commands Supported Report Command reports support of at least the Anti-theft Command Class to be able to disable anti-theft protection:

- Anti-theft Command Class

The Anti-theft Command Class is supported securely making malicious attempts to enable anti-theft protection very difficult.

2.2.8 Anti-theft Command Class, version 3

This Command Class is used to lock (and possibly unlock) a node.

2.2.8.1 Compatibility Considerations

2.2.8.1.1 Command Class dependencies

A node supporting this Command Class MUST also support the *Anti-theft Unlock Command Class, version 1*.

2.2.8.1.2 Lock/Unlock requirements

The Anti-Theft Command Class is intended for devices operating in public locations such as hotel rooms or a conference center. The purpose of the Anti-theft Command Class is to render a device useless if it is removed from its actual network without being unlocked by the owner or a service provider.

A node can be *locked* (or protected) or *unlocked*.

The unlocked state means that the node will operate normally in any Z-Wave network.

The locked state means that the node will restrict usage of most of its command classes after being excluded from a network, so that it cannot be used by any controlling unit. The locked state MUST persist after the following operations:

- Network inclusion/exclusion
- Factory reset to default.
- Firmware upgrade

When a node is locked and either excluded from a network or factory reset to default, it will enter its **restricted** mode. In the restricted mode, the node will not grant access to its application functionalities. The restricted mode MUST stop when the node is unlocked again.

When a node runs in the restricted mode, the NIF MUST adapt its content to reflect the currently supported Command Classes. The following Command Classes MUST still be supported when a node is running in restricted mode:

- Anti-Theft Command Class
- Anti-Theft Unlock Command Class
- Manufacturer Specific Command Class
- Version Command Class
- Wake-Up Command Class
- All Command Classes that MUST always be supported at the non-secure level, refer to List of defined Z-Wave Command Classes (see “This Command Class MUST always be in the NIF if supported”)

All other supported command classes SHOULD be removed in the restricted mode.

The locked state may be changed back to unlocked with this Command Class or the Anti-Theft Unlock Command Class.

When in restricted mode, a node MUST still be able to enter learn mode and join or leave a Z-Wave Network and support Z-Wave protocol operations.

2.2.8.1.3 Multi Channel Considerations

Multi Channel End Points SHOULD NOT support Anti-theft Command Class.

2.2.8.2 Anti-Theft Set Command

This command is used to lock or unlock a node.

Table 2.42: Anti-theft Set Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT (0x5D)							
Command = ANTITHEFT_SET (0x01)							
State	Number of Magic Code bytes						
Magic Code 1							
...							
Magic Code N							
Manufacturer ID MSB							
Manufacturer ID LSB							
Anti-theft Hint Length							
Anti-theft Hint							
...							
Anti-theft Hint M							
Z-Wave Alliance locking entity ID (MSB)							
Z-Wave Alliance locking entity ID (LSB)							

State (1 bit)

This field MUST indicate the desired locked/unlocked state for the receiving node.

The value 0 MUST indicate that the supporting node MUST change its state to unlocked.

The value 1 MUST indicate that the supporting node MUST change its state to locked.

If the receiving node is currently locked and this field is set to 0 (unlock), the receiving node MUST change its state to unlocked if the Magic Code field value matches the Magic Code value that was used to lock the node.

If the receiving node is currently locked and this field is set to 1 (lock), the receiving node MUST ignore this command.

If the receiving node is currently unlocked and this field is set to 0 (unlock), the receiving node MUST ignore this command.

If the receiving node is currently unlocked and this field is set to 1 (lock), the receiving node MUST change its state to lock and save the associated Magic code, Manufacturer ID, Anti-Theft Hint and Z-Wave Alliance locking entity ID.

Magic Code Length (7 bit)

This field MUST indicate the length (in byte) of the *Magic Code* field.

This field MUST be in the range 1..10.

Magic Code (N bytes)

This field contains the Magic Code used to lock or unlock the node.

The length of this field in byte MUST be according to the *Magic Code Length* field.

The receiving node MUST change its state from locked to unlocked if and only if this field's value matches the Magic Code that was used to lock the node.

Manufacturer ID (2 bytes)

This field describes the Z-Wave Manufacturer ID of the company's product that has locked the node.

This field MUST be set to 0x00 by a sending node and ignored by a receiving node if the State field is set to 0.

If a controlling node has a Z-Wave Manufacturer ID assigned, this field MUST be set to its assigned Manufacturer ID in Z-Wave Plus Assigned Manufacturer IDs.

If a controlling node has no Z-Wave Manufacturer ID assigned, this field MUST be set to 0x00.

Anti-theft Hint Length (8 bits)

This field MUST indicate the length (in byte) of the *Anti-Theft Hint* field.

This field MUST be in the range 0..10.

Anti-theft Hint (N bytes)

This field is used as an identifier or key value to help retriving the Magic Code.

The length of this field in byte MUST be according to the *Anti-Theft Hint Length* field. This field MUST be omitted if the *Anti-Theft Hint Length* field is set to 0.

The format of this field is manufacturer/controlling node specific.

Z-Wave Alliance locking entity ID (2 bytes)

This field MUST specify a unique identifier for the entity that has locked the node.

A supporting node MUST NOT change its state to locked if this field is omitted or set to 0x00.

A controlling node MUST NOT lock a device without having a valid Z-Wave Alliance locking entity ID value.

Contact the Z-Wave Alliance to get an ID assigned. With this ID, the Z-Wave alliance will be able to provide the following information:

- Name of the organization
- Contact Phone for unlock information
- Optional website/url
- Optional unlock support email address

This field MUST be set to 0x00 if the *state* field is set to 0.

This field MUST NOT be set to 0x00 if the *state* field is set to 1.

2.2.8.3 Anti-theft Get Command

This command is used to request the locked/unlocked state of a supporting node.

The Anti-Theft Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.43: Anti-theft Get Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT (0x5D)							
Command = ANTITHEFT_GET (0x02)							

2.2.8.4 Anti-Theft Report Command

This command is used to advertise the lock/unlock state of a supporting node.

Table 2.44: Anti-theft Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT (0x5D)							
Command = ANTITHEFT_REPORT (0x03)							
Anti-theft Protection Status							
Manufacturer ID MSB							
Manufacturer ID LSB							
Anti-theft Hint Length							
Anti-theft Hint 1							
...							
Anti-theft Hint N							
Z-Wave Alliance locking entity ID (MSB)							
Z-Wave Alliance locking entity ID (MSB)							

Anti-theft Protection Status (8 bits)

Anti-theft Protection Status specifies the actual status of Z-Wave device in question. Refer to the table below with respect to defined status values.

Table 2.45: Anti-theft Protection Status, version 3

Value	Description
0x00	Reserved.
0x01	Anti-Theft Protection is currently disabled, the node is unlocked
0x02	Anti-Theft Protection is currently enabled, the node is locked. However, it did not change network so it is fully functional.
0x03	Anti-Theft Protection is currently enabled, the node is locked. The node was reset and/or changed network and runs in restricted mode.
0x04..0xFF	Reserved

Manufacturer ID (2 bytes)

This field describes the Z-Wave Manufacturer ID of the company's product that has locked the node.

This field MUST be set to 0x00 if the *Anti-theft Protection Status* field is set to 0x01.

If the *Anti-theft Protection Status* field is set to 0x02 or 0x03 and a Z-Wave Manufacturer listed in [32] has locked the node, this field will advertise the Manufacturer ID that was provided in the Anti-theft Set Command.

If the *Anti-theft Protection Status* field is set to 0x02 or 0x03 and an entity who's not a Z-Wave Manufacturer listed in [32] has locked the node, this field will be set to 0x00.

Anti-theft Hint Length (8 bits)

This field MUST indicate the length (in byte) of the *Anti-Theft Hint* field.

This field MUST be in the range 0..10.

If the *Anti-theft Protection Status* field is set to 0x01, a sending node MUST set this field to 0.

If the *Anti-theft Protection Status* field is set to 0x02 or 0x03, a sending node MUST advertise the length of the Anti-Theft Hint field that was used to lock the node.

Anti-theft Hint (N bytes)

This field is used as an identifier or key value to help retrieving the Magic Code.

The length of this field in byte MUST be according to the *Anti-Theft Hint Length* field. This field MUST be omitted if the *Anti-Theft Hint Length* field is set to 0.

A supporting node MUST advertise the value that was specified in the Anti-Theft Set Command when it got locked.

Z-Wave Alliance locking entity ID (2 bytes)

This field advertises a unique identifier for the entity that has locked the node. Anti-Theft Command Class, list of assigned Locking Entity IDs registry contains the valid assigned IDs with additional information about the entity.

Contact the Z-Wave Alliance to get an ID assigned. With this ID, the Z-Wave alliance will be able to provide the following information:

- Name of the organization
- Contact phone for unlock information
- Optional website/url
- Optional unlock support email address

This field MUST be set to 0x00 if the *Anti-theft Protection Status* field is set to 0x01.

2.2.9 Anti-theft Unlock Command Class, version 1

This Command Class is used to unlock a device that has been locked by the Anti-theft Command Class. Controllers that do not lock devices still can control this command class in order to unlock devices that have been locked by previous owners.

2.2.9.1 Compatibility Considerations

A node supporting this Command Class MUST also support the Anti-Theft Command Class, version 3.

2.2.9.1.1 Multi Channel Considerations

Multi Channel End Points SHOULD NOT support the Anti-Theft Unlock Command Class.

2.2.9.2 Anti-Theft Unlock State Get Command

This command is used to request the locked/unlocked state of the node.

The Anti-Theft Unlock State Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.46: Anti-theft Unlock State Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT_UNLOCK (0x7E)							
Command = ANTITHEFT_UNLOCK_STATE_GET (0x01)							

2.2.9.3 Anti-Theft Unlock State Report Command

This command is used to advertise the current locked/unlocked state of the node with some additional information.

Table 2.47: Anti-theft Unlock State Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT_UNLOCK (0x7E)							
Command = ANTITHEFT_UNLOCK_STATE_REPORT (0x02)							
Reserved		Anti-theft Hint Length				Restricted	State
Anti-theft Hint							
...							
Anti-theft Hint N							
Manufacturer ID (MSB)							
Manufacturer ID (LSB)							
Z-Wave Alliance locking entity ID (MSB)							
Z-Wave Alliance locking entity ID (MSB)							

State (1 bit)

This field MUST indicate the current locked/unlocked state of the device.

The value 0 MUST indicate that the supporting node is unlocked.

The value 1 MUST indicate that the supporting node is locked.

The value 0x01 from the *Anti-theft Protection Status* field from the Anti-theft Report Command MUST be mapped to the value 0 for this field

The values 0x02 and 0x03 from the *Anti-theft Protection Status* field from the Anti-theft Report Command MUST be mapped to the value 1 for this field

Restricted (1 bit)

This field MUST indicate if the node currently runs in restricted mode.

The value 0 MUST indicate that the supporting node is not restricted.

The value 1 MUST indicate that the supporting node is restricted.

This field MUST be set to 0 if the *State* field is set to 0.

The values 0x01 and 0x02 from the *Anti-theft Protection Status* field from the Anti-theft Report Command MUST be mapped to the value 0 for this field

The value 0x03 from the *Anti-theft Protection Status* field from the Anti-theft Report Command MUST be mapped to the value 1 for this field

Anti-theft Hint Length (4 bits)

This field MUST indicate the length (in byte) of the *Anti-Theft Hint* field.

This field MUST be in the range 0..10.

If the state field is set to 0, a sending node MUST set this field to 0.

If the state field is set to 1, a sending node MUST advertise the length of the Anti-Theft Hint field that was used to lock the node.

Anti-theft Hint (N bytes)

This field is used as an identifier or key value to help retrieving the Magic Code.

The length of this field in byte MUST be according to the *Anti-Theft Hint Length* field. This field MUST be omitted if the Anti-Theft Hint Length field is set to 0.

A supporting node MUST advertise the value that was specified in the Anti-Theft Set Command when it got locked.

Manufacturer ID (2 bytes)

This field describes the Z-Wave Manufacturer ID of the company's product that has locked the node.

This field MUST be set to 0x00 if the state field is set to 0.

If the state field is set to 1 and a Z-Wave Manufacturer listed in [32] has locked the node, this field will advertise the Manufacturer ID that was provided in the Anti-theft Set Command.

If the state field is set to 1 and an entity who's not a Z-Wave Manufacturer listed in [32] has locked the node, this field will be set to 0x00.

Z-Wave Alliance locking entity ID (2 bytes)

This field advertises a unique identifier for the entity that has locked the node. Anti-Theft Command Class, list of assigned Locking Entity IDs registry contains the valid assigned IDs with additional information about the entity.

Contact the Z-Wave Alliance to get an ID assigned. With this ID, the Z-Wave alliance will be able to provide the following information:

- Name of the organization
- Contact phone for unlock information
- Optional website/url
- Optional unlock support email address

This field MUST be set to 0x00 if the state field is set to 0.

If the state field is set to 1, this field MUST advertise the ID that was provided in the Anti-Theft Set Command.

2.2.9.4 Anti-Theft Unlock Set Command

This command is used to unlock a node that is currently locked.

A receiving node MUST ignore this command if its current state is unlocked.

Table 2.48: Anti-theft Unlock Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT_UNLOCK (0x7E)							
Command = ANTITHEFT_UNLOCK_SET (0x03)							
Reserved				Magic code length			
Magic Code 1							
...							
Magic Code N							

Magic Code Length (4 bits)

This field MUST indicate the length in byte of the Magic Code field.

This field MUST be in the range 1..10

Magic Code (N bytes)

This field contains the Magic Code used to unlock the node.

The length of this field in byte MUST be according to the Magic Code Length field.

The receiving node MUST change its state to unlocked (and exit restricted mode) if this field's value matches the Magic Code that was used to lock the node.

2.2.10 Authentication Command Class, version 1 [NEVER CERTIFIED]

Warning: THIS COMMAND CLASS HAS NEVER BEEN CERTIFIED

This command class has never been implemented and certified by a Z-Wave product Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.

Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

Authentication Command Class is a generic Command Class which can be used for transporting different technology authentication data contents (e.g., RFID, Magnetic Cards, etc.) in a Z-Wave network for access control systems.

2.2.10.1 Terminology

Authentication Data contents are initialized on a supporting node using a unique **Authentication Data ID**. Each Authentication Data ID also has an **Authentication Technology Type**, representing the input or technology with which the Authentication Data can be used. A user having access with several types of input technologies will have a Data Identifier for each of the technologies.

A node may support one or more authentication technologies including User Code. **Combination of valid sets of Authentication Data** can be configured on a supporting node using a unique **Authentication ID**. The Authentication ID will for example allow a user only if he/she inputs both an RFID tag and the corresponding User Code. An actual user may be associated with several Authentication IDs.

An example of Authentication Data ID and Authentication ID configurations are described in Figure 2.4.

A node may support a **Checksum** functionality. A controlling node can request a checksum representing all authentication data content (either Authentication IDs or Authentication Data IDs) defined at the supporting node to ensure that the authentication data content databases are synchronized.

2.2.10.2 Interoperability Considerations

This Command Class can be used in conjunction with the Entry Control Command Class, to report the Authentication Data input to a controlling application.

This Command Class can be used in conjunction with the Door Lock Command Class or Barrier Operator Command Class. In this case, a supporting node MUST reflect Authentication Data inputs in the door lock status when relevant. (e.g. when the door becomes unsecured by Authentication Data input, the Door Lock Operation mode is updated to unsecure with an optional timeout).

A node supporting this Command Class MAY have an Association group issuing the corresponding Door Lock Operation Set Commands or Basic Set Commands when there are a valid Authentication Data input in order to control other door locks.

A supporting node may support several authentication technologies such as User Code, RFID, Cards and etc. A combination these technologies may also be needed to get access. In this case, a supporting node can be configured to accept certain combinations of authentication technologies.

An assigned User Code in the User Code Command Class may be part of an Authentication ID. In this case, all the Authentication credentials (including the User Code) defined in the Authentication ID MUST be input at the node in order to trigger the outcome defined by the User Code Status.

In other words, the User Code input alone is no longer enough, it will require all the credentials defined in the Authentication ID to get access. An example is given in Figure 2.3.

2.2.10.3 Authentication Capability Get Command

This command is used to request the Authentication technology capabilities, number of Authentication ID and Data ID supported by the receiving node.

The Authentication Capability Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.49: Authentication Capability Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_CAPABILITY_GET (0x01)							

2.2.10.4 Authentication Capability Report Command

This command is used to advertise the Authentication Command Class capabilities of sending node.

Table 2.50: Authentication Capability Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_CAPABILITY_REPORT (0x02)							
Supported Data ID Entries (MSB)							
Supported Data ID Entries (LSB)							
Supported Authentication ID Entries (MSB)							
Supported Authentication ID Entries (LSB)							
MAR	MADR	OR	Reserved	Supported Authentication Technology Type Bit Mask length			
Supported Authentication Technology Type Bit Mask 1							
...							
Supported Authentication Technology Type Bit Mask N							
Supported Checksum Type Bit Mask							
Reserved				Supported Fallback Status Bit Mask length			
Supported Fallback Status Bit Mask 1							
...							
Supported Fallback Status Bit Mask N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Supported Data ID Entries (16 bits)

This field is used to advertise the number of supported Authentication Data ID Entries.

The first byte MUST carry the most significant byte of the 16 bits value.

This field MUST be set to the total amount of supported Authentication Data IDs. This field MUST be in the range 1…65535.

Supported Authentication ID Entries (16 bits)

This field is used to advertise the number of supported Authentication ID Entries.

The first byte MUST carry the most significant byte of the 16 bits value.

This field MUST be set to the total amount of supported combinations entries. This field MUST be in the range 1…65535.

MAR (Multiple Authentication ID Report) Support (1 bit)

This field indicates if the sending node supports reporting the Multiple Authentication ID Blocks at once in a single Authentication Technologies Combination Report Command. This functionality should be supported by node supporting large amount of Authentication ID Entries.

The value 1 MUST indicate that the Multiple Authentication ID Report functionality is supported.

The value 0 MUST indicate that the Multiple Authentication ID Report functionality is not supported.

MADR (Multiple Authentication Data ID Report) Support (1 bit)

This field indicates if the sending node supports reporting the Multiple Authentication Data ID Blocks at once in a single Authentication Data Report Command. This functionality should be supported by node supporting large amount of Authentication Data ID Entries.

The value 1 MUST indicate that the Multiple Authentication Data ID Report functionality is supported.

The value 0 MUST indicate that the Multiple Authentication Data ID Report functionality is not supported.

OR Support (1 bit)

This field is used to advertise if the node supports being set Authentication ID combinations using a OR logic (User Code OR Authentication data ID 1 OR … OR Authentication data ID N)

The value 1 MUST indicate that the OR flag in the Authentication Technologies combination is supported.

The value 1 MUST indicate that the OR flag in the Authentication Technologies combination is not supported.

Supported Authentication Technology Type Bit Mask length (4 bits)

This field MUST advertise the length in bytes of the *Supported Authentication Technology Type Bit Mask* field carried in the command.

This field MUST be set to the minimum value which allows advertising all supported Authentication Technology types.

Supported Authentication Technology Type Bit Mask (N bytes)

This field advertises the supported Authentication Technology Type values. The length of this field in bytes MUST match the value advertised in the *Supported Authentication Technology Bit Mask Length* field.

Authentication Technology Type values are described in Table 2.52. In this field the supported Authentication Technology Type values MUST be encoded as follow:

- Bit 0 in Bit Mask 1 represents Authentication Technology Type 0x00 (reserved) and MUST be set to 0.
- Bit 1 in Bit Mask 1 represents Authentication Technology Type 0x01.
- Bit 2 in Bit Mask 1 represents Authentication Technology Type 0x02.
- …

If an Authentication Technology Type is supported, the corresponding bit MUST be set to '1'.

If an Authentication Technology Type is not supported, the corresponding bit MUST be set to '0'.

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Supported Checksum Type Bit Mask (8 bits)

This field advertises the supported Checksum Type values.

Checksum Type values are described in Table 2.60. The supported Checksum Type values MUST be encoded as follow:

- Bit 0 in Bit Mask 1 represents Checksum Type 0x00.
- Bit 1 in Bit Mask 1 represents Checksum Type 0x01.
- …

If a Checksum Type is supported, the corresponding bit MUST be set to '1'.

If a Checksum Type is not supported, the corresponding bit MUST be set to '0'.

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Supported Fallback Status Bit Mask length (4 bits)

This field MUST advertise the length in bytes of the *Supported Fallback Status Bit Mask* field carried in the command.

This field MUST be set to the minimum value which allows advertising all supported Fallback Statuses.

Supported Fallback Status Bit Mask (N bytes)

This field advertises the supported Fallback Status values. The length of this field in bytes MUST match the value advertised in the *Supported Fallback Status Bit Mask Length* field.

Fallback Status values are described in Table 2.56. This field MUST be encoded as follow:

- Bit 0 in Bit Mask 1 represents Fallback Status 0x00.
- Bit 1 in Bit Mask 1 represents Fallback Status 0x01.
- …

Fallback Statuses 0x00 and 0x01 MUST be supported by all supporting nodes.

If a Fallback Status is supported, the corresponding bit MUST be set to '1'.

If a Fallback Status is not supported, the corresponding bit MUST be set to '0'.

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

2.2.10.5 Authentication Data Set Command

This command is used to define Authentication Data for a given Data Identifier at the receiving node.

Table 2.51: Authentication Data Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_DATA_SET (0x03)							
Authentication Data ID 1 (MSB)							
Authentication Data ID 2 (LSB)							
Reserved				Authentication Technology Type			
Authentication Data length							
Authentication Data 1							
…							
Authentication Data N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Authentication Data ID (16 bits)

This field is used to identify an Authentication Data entry.

This field MUST be in the range 0..Max Number of Supported Data ID Entries advertised by the node in the Authentication Capability Report Command.

The first byte MUST carry the most significant byte of the 16 bits.

Values in the range 1..Max Number of Supported Data ID Entries MUST indicate the actual Authentication Data ID record to update.

The value 0 MUST indicate that the receiving node MUST erase the Authentication Data for all supported Authentication Data ID that are associated to specified **Authentication Technology Type**. If this field is set to 0x00, the Authentication Data length MUST be set to 0x00.

Authentication Technology Type (4 bits)

This field is used to specify the authentication medium associated to the Authentication Data ID.

The field MUST be encoded according to Table 2.52.

Table 2.52: Authentication Data Set::Authentication Technology Type encoding

Value	Authentication Technology	Version
0x00	Reserved	1
0x01	RFID tag	1
0x02	Magnetic card	1
0x03..0x0F	Reserved	1

Authentication Data length (8 bits)

This field is used to describe the length in bytes of the *Authentication Data* field.

The value 0 MUST indicate to erase the data entry represented by the Data Identifier. ** Authentication** Data (N bytes)

This field is used to advertise the Authentication Data contents to be set the specified Authentication Data ID.

The length of this field in bytes MUST be according to the corresponding *Authentication Data Length* field value. If the *Authentication Data Length* field is set to 0x00, this field MUST be omitted.

2.2.10.6 Authentication Data Get Command

This command is used to request the Authentication Data defined for a given Authentication Data ID.

The Authentication Data Report MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.53: Authentication Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_DATA_GET (0x04)							
Authentication Data ID 1 (MSB)							
Authentication Data ID 2 (LSB)							
Reserved							Report More

Authentication Data ID (16 bits)

This field is used to describe the requested authentication Data Identifier.

The first byte MUST carry the most significant byte of the 16 bit value.

A supporting node MUST return this value in the *Authentication Data ID* field of the first Authentication Data ID Block in the returned Authentication Data Report Command.

Report More (1 bit)

This field is used to instruct the receiving node to report as many Authentication Data ID as possible within a single Z-Wave command (that fits in a single Z-Wave frame) because the sending node intends to read the whole (or a large part of the) Authentication Data database.

This field MUST be ignored by a receiving node if it advertises no support for Multiple Authentication Data Report (MADR) in the Authentication Capability Report Command.

The value 0 MUST indicate to return a report for the requested Authentication Data ID only.

The value 1 MUST indicate to return a report for the requested Authentication Data ID and additionally report as many Authentication Data ID blocks as possible in the response.

When this field is set to 1, a node advertising support for Multiple Authentication Data Report (MADR) in the Authentication Capability Report Command:

- SHOULD return at least 2 Authentication Data ID blocks in the Authentication Data Report Command unless the last used Authentication Data ID or a non-supported Authentication Data ID is requested.
- SHOULD return as many consecutive non-empty Authentication Data ID record as possible.
- The subsequent Authentication Data ID blocks MUST contain consecutive Authentication Data ID having Authentication Data length different than 0. For example, a node supports 5 authentication Data IDs and both RFID (0x01) and Magnetic card (0x02) authentication technologies, and the setting is defined as follow. In this case, the supporting node MUST report Authentication Data ID 1 and 4 when requested about Authentication Data ID 1. - Authentication Data ID 1, Technology Type = 0x01, Data length = 3, Data = C10001 - Authentication Data ID 2, Technology Type = 0x01, Data length = 0 - Authentication Data ID 3, Technology Type = 0x01, Data length = 0 - Authentication Data ID 4, Technology Type = 0x02, Data length = 2, Data = CDF1 - Authentication Data ID 5, Technology Type = 0x02, Data length = 0

2.2.10.7 Authentication Data Report Command

This command is used to advertise the Authentication Data assigned at the supporting node for a given Authentication Data ID entry.

Table 2.54: Authentication Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_DATA_REPORT (0x05)							
Number of Authentication Data ID Blocks							
Authentication Data ID 1 (MSB)							
Authentication Data ID 2 (LSB)							
Reserved				Authentication Technology Type			
Authentication Data length							
Authentication Data 1 - 1							
...							
Authentication Data N - 1							
...							
Authentication Data ID 1 (MSB) M							
Authentication Data ID 2 (LSB) M							
Reserved				Authentication Technology Type M			
Authentication Data Length M							
Authentication Data 1 - M							
...							
Authentication Data N - M							
Next Authentication Data ID 1 (MSB)							
Next Authentication Data ID 2 (LSB)							

Number of Authentication Data ID Blocks (8 bits)

This field is used to specify how many Authentication Data ID blocks are contained in the actual command.

This field MUST be in the range 1..255. A node MUST respect the Z-Wave MAC frame size or Transport service limits when sending this command.

The number of Authentication Data ID contained in the command MUST be according to this field. An Authentication Data ID block MUST comprise the following fields:

- Authentication Data (16 bits)
- Authentication Technology Type (4 bits)
- Authentication Data Length (8 bits)
- Authentication Data (N bytes)

This field MUST be set to 1 by a node advertising no support for Multiple Authentication Data ID Block Report (MADR) in the User Code Capabilities Report Command.

Authentication Data ID (16 bits)

This field is used to describe the advertised authentication Data Identifier.

If this field is set to 0x00 or a non-supported Authentication Data ID, the *Authentication Technology Type* and the *Authentication Data Length* fields MUST be set to 0x00.

Authentication Technology Type (4 bits)

This field is used to advertise the authentication medium associated to the Authentication Data ID.

The field MUST be encoded according to Table 2.52.

Authentication Data length (8 bits)

This field is used to describe the length in bytes of the *Authentication Data* field.

The value 0 MUST indicate that the specified Authentication Data ID is not supported.

Authentication Data (N bytes)

This field is used to advertise the Authentication Data contents set for the specified Authentication Data ID.

The length of this field in bytes MUST be according to the corresponding *Authentication Data Length* field value. If the *Authentication Data Length* field is set to 0x00, this field MUST be omitted.

Next Authentication Data ID (16 bits)

This field is used to advertise the next Authentication Data ID set at the sending node with non-empty Authentication Data.

This field MUST be set to 0x00 if there are no more Data Identifiers are defined after the Authentication Data ID value of the *Authentication Data ID* field of the last Authentication Data ID block.

2.2.10.8 Authentication Technologies Combination Set Command

This command is used to define the Combination Entries at the receiving node.

Table 2.55: Authentication Technologies Combination Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_TECHNOLOGIES_COMBINATION_SET (0x06)							
Authentication Data ID 1 (MSB)							
Authentication Data ID 2 (LSB)							
Fallback Status							
User Identifier 1 (MSB)							
User Identifier 2 (LSB)							
Schedule ID 1 (MSB)							
Schedule ID 2 (LSB)							
OR logic	Number of Authentication Data ID						
Authentication Data ID 1 (MSB) 1							
Authentication Data ID 2 (LSB) 1							
...							
Authentication Data ID 1 (MSB) M							
Authentication Data ID 2 (LSB) M							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Authentication ID (16 bits)

This field is used to specify the Authentication ID that is to be updated.

This field MUST be in the range 0..max number of Supported Authentication ID Entries

The first byte MUST carry the most significant byte of the 16 bits.

The value 0 is used to erase all the Authentication Data for all supported Authentication ID Entries. If this field is set to 0x00 and all the other fields MUST set to 0x00, and a receiving node MUST erase all configured Authentication ID Entries.

Values in the range 1..Max Number of Supported Authentication ID Entries MUST indicate the actual Authentication ID record to update.

Fallback Status (8 bits)

This field is used to configure the status of Authentication ID at the receiving node if a User Code entry is not defined (i.e. *User Identifier* field set to 0x00).

The status represents the outcome to perform at the node when all the Authentication Data defined in the Authentication ID has been input at the supporting node, during the defined Schedule ID.

This field MUST set but ignored if the User Identifier is set to a value greater than 0x00. Instead, the User ID Status defined for the corresponding User Identifier in the User Code Command Class MUST take precedence over this field.

If the *User Identifier* field is set to 0x00, this field MUST be encoded and interpreted according to Table 2.56.

Table 2.56: Authentication Technologies Combination Set::Fallback Status encoding

Value	Description	Version
0x00	**Available:** The database entry represented by this Authentication ID is not in use. This status MUST be used to delete Authentication ID entries.	1
0x01	**Enabled / Grant Access** The database entry represented by this Authentication ID is in use and enabled. When a user has input all required Authentication credentials associated to this Authentication ID, the node: • MUST indicate that the input was accepted • MUST grant access to the user. • MUST secure the Door Lock again after a timeout (either configured if set in Timed Operation or pre-defined if the Door Lock is set in Constant Operation)	1
0x02	**Disabled:** The database entry represented by this Authentication ID is in use but disabled. E.g. It allows a controller to push Authentication ID combinations to a supporting node and let these credentials be subsequently activated by a local interface. When a user has input all required Authentication credentials associated to this Authentication ID, the node: • MUST indicate that the input was not accepted • MUST NOT grant access to the user.	1
0x03	**Messaging:** The database entry represented by this Authentication ID is in use. The value is used to relay notifications to a controlling application. E.g. A user may enter a messaging code for notifying that maintenance started or an application should activate the alarm system. When a user has input all required Authentication credentials associated to this Authentication ID, the node: • SHOULD indicate that the input was accepted • MUST NOT grant access to this user • MUST NOT take preventive actions for further attempts to enter input. • MUST send an Entry Control Notification or a Notification Report via the Lifeline Group to identify the messaging User Identifier or Authentication ID.	1

continues on next page

Table 2.56 – continued from previous page

Value	Description	Version
0x04	**Passage Mode:** The database entry represented by this Authentication ID is in use. The value is used to let the Door Lock permanently grant access. Passage Mode can be activated and deactivated using the Door Lock Operation Set (Unsecured without timeout / Secured) or via a Passage Mode credential input. E.g. A receptionist comes in the morning and enters a Passage Mode credentials to allow anybody to subsequently enter the premises without entering special access codes until Passage Mode is deactivated again. When a user has input all required Authentication credentials associated to this Authentication ID, the node: • MUST indicate that the code is accepted • MUST toggle the Door Lock operation state between secured and unsecured without timeout (and also disable any auto-relock functionality). • MUST send a Door Lock Operation Report and Door Lock Configuration Report via the Lifeline Association Group to advertise the new Door Lock state.	1
0x05..0xFF	Reserved	1

User Identifier (16 bits)

This field is used to specify the identity of the User Code (User Code Command Class) which MUST be used in combination with authentication data to get access.

If User Identifier has a User ID Status set to 0x00 in the User Code Command Class, the combination will not allow access as it cannot be input at the supporting node.

This field is set to a value larger than 0x00 and the supporting node does not support the User Code Command Class, the Authentication ID will be ignored as such a User Code cannot be input at the supporting node.

If a User Code is defined in an Authentication ID with an AND logic, the input of the User Code alone is no longer sufficient to trigger the outcome defined by the User ID Status. All Authentication Data indicated by the Authentication Data IDs in this command MUST be input to trigger the outcome specified by the User ID Status.

Schedule ID (16 bits)

This field is used specify the Schedule during which the Authentication ID is active.

The value 0x00 MUST indicate that the Authentication ID is active at all times.

Values greater than 0 MUST indicate that the Authentication ID is valid or active only during the corresponding Schedule ID, defined in the Generic Schedule Command Class.

OR logic (1 bit)

This field is used to specify if OR logic is to be applied for the actual Authentication ID.

Nodes advertising no support for the OR in the Authentication Capability Report Command MUST ignore this field and apply an AND logic.

If this field is set to 0, an AND logic MUST be applied for the actual Authentication ID.

In this case, the **User Code (if defined) and all the defined Authentication Data** must be input (during the Schedule ID) in order to get access or provide the outcome defined in the corresponding User Code Status / Fallback Status.

If this field is set to 1, an OR logic MUST be applied for the actual Authentication ID.

In this case, the **User Code (if defined) or any of the defined Authentication Data** must be input (during the Schedule ID) in order to get access or provide the outcome defined in the corresponding User Code Status / Fallback Status.

Number of Authentication Data ID (7 bits)

This field is used to specify how many Authentication Data ID are contained in the the actual Authentication ID. Each Authentication Data ID MUST be 16 bits long.

This field MUST be set to 0, if no Authentication Data ID MUST be used for the combination and the User Identifier (User Code) alone is enough to get access.

If this field is set to 0, the Authentication Data ID field MUST be omitted.

For example, if this field is set to 3, the corresponding Authentication Data ID field must be 6 bytes long.

Authentication Data ID (M x 16 bits)

This field is used to identify the list of Authentication Data ID that are required for the specified Authentication ID.

The first byte MUST carry the most significant byte of the 16 bits.

In an Authentication ID, if any of the Authentication Data ID does not have any Authentication Data defined (Authentication Data Length = 0), the Authentication ID MUST NOT permit any access.

2.2.10.9 Authentication Technologies Combination Get Command

This command is used for requesting defined combination entries of a specific Authentication ID.

The Authentication Technologies Combination Report MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.57: Authentication Technologies Combination Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_TECHNOLOGIES_COMBINATION_GET (0x07)							
Authentication ID (MSB)							
Authentication ID (LSB)							
Reserved							Report More

Authentication ID (16 bits)

This field is used to describe the requested Authentication ID.

The first byte MUST carry the most significant byte of the 16 bits.

A supporting node MUST return this value in the *Authentication Data ID* field of the first Authentication Data ID Block in the returned Authentication Technologies Combination Report Command.

Report More (1 bit)

This field is used to instruct the receiving node to report as many Authentication ID as possible within a single Z-Wave command (that fits in a single Z-Wave frame) because the sending node intends to read the whole (or a large part of the) Authentication ID database.

This field MUST be ignored by a receiving node if it advertises no support for Multiple Authentication Report (MAR) in the Authentication Capability Report Command.

The value 0 MUST indicate to return a report for the requested Authentication ID only.

The value 1 MUST indicate to return a report for the requested Authentication ID and additionally report as many Authentication ID blocks as possible in the response.

When this field is set to 1, a node advertising support for Multiple Authentication Report (MAR) in the Authentication Capability Report Command:

- SHOULD return at least 2 Authentication ID blocks in the Authentication Technologies Combination Report Command unless the last used Authentication ID or a non-supported Authentication ID is requested.
- SHOULD return as many consecutive non-empty Authentication ID record as possible.
- The subsequent Authentication ID blocks MUST contain consecutive Authentication ID having User Identifier or Fallback Status different than 0.

2.2.10.10 Authentication Technologies Combination Report Command

This command is used to advertise defined Combination Entries of a specific Authentication ID.

Table 2.58: Authentication Technologies Combination Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_TECHNOLOGIES_COMBINATION_REPORT (0x08)							
Number of Authentication ID blocks							
Authentication ID block length 1							
Authentication ID 1 (MSB) 1							
Authentication ID 2 (LSB) 1							
Fallback Status 1							
User Identifier 1 (MSB) 1							
User Identifier 2 (LSB) 1							
Schedule ID 1 (MSB) 1							
Schedule ID 2 (LSB) 1							
OR logic 1	Number of Authentication Data IDs 1						
Authentication Data ID 1 (MSB) 1 - 1							
Authentication Data ID 2 (LSB) 1 - 1							
...							
Authentication Data ID 1 (MSB) M - 1							
Authentication Data ID 2 (LSB) M - 1							
...							
Authentication ID block length N							
Authentication ID 1 (MSB) N							
Authentication ID 2 (LSB) N							
Fallback Status N							
User Identifier 1 (MSB) N							
User Identifier 2 (LSB) N							
Schedule ID 1 (MSB) N							
Schedule ID 2 (LSB) N							
OR logic N	Number of Authentication Data IDs N						
Authentication Data ID 1 (MSB) 1 - N							
Authentication Data ID 2 (LSB) 1 - N							
...							
Authentication Data ID 1 (MSB) M - N							
Authentication Data ID 2 (LSB) M- N							

continues on next page

Table 2.58 – continued from previous page

7	6	5	4	3	2	1	0
Next Authentication ID 1 (MSB)							
Next Authentication ID 2 (LSB)							

Number of Authentication ID Blocks (8 bits)

This field is used to specify how many Authentication ID blocks are contained in the actual command.

This field MUST be in the range 1..255. A node MUST respect the Z-Wave MAC frame size or Transport service limits when sending this command.

The number of Authentication ID contained in the command MUST be according to this field. An Authentication Data ID block MUST comprise the following fields:

- Authentication ID Block Length (8 bits)
- Authentication ID (16 bits)
- Fallback Status (8 bits)
- User Identifier (16 bits)
- Schedule ID (16 bits)
- OR logic / Number of Authentication Data IDs (8 bits)
- Authentication Data ID (M x 16 bits)

This field MUST be set to 1 by a node advertising no support for Multiple Authentication ID Block Report (MAR) in the Authentication Capability Report Command.

Authentication ID block length (8 bits)

This field is used to indicate the total length of the actual Authentication ID Block. This field MUST advertise the length in bytes of the Authentication ID Block, including itself.

Authentication ID (16 bits)

This field is used to specify the Authentication ID for the actual Authentication ID Block.

The first byte MUST carry the most significant byte of the 16-bits value.

If this field is set to 0x00 or a non-supported Authentication ID, the *Fallback Status*, *User Identifier*, *Schedule ID* and *Number of Authentication Data ID* fields MUST be set to 0x00.

Fallback Status (8 bits)

This field is used to advertise the status of Authentication ID defined at the receiving node.

The status represents the outcome to perform at the node when all the Authentication Data defined in the Authentication ID block has been input at the supporting node, during the defined Schedule ID.

This field MUST be ignored if the User Identifier field in the corresponding Authentication ID Block is set to a value greater than 0x00. Instead, the User ID Status defined for the corresponding User Identifier in the User Code Command Class MUST take precedence over this field.

If the *User Identifier* field is set to 0x00, this field MUST be encoded and interpreted according to.

User Identifier (16 bits)

This field is used to advertise the identity of the User Code (User Code Command Class) which is used in combination with authentication data to get access for this Authentication ID block.

If User Identifier has a User ID Status set to 0x00 in the User Code Command Class, the the combination will not allow access as it cannot be input at the supporting node.

This field is set to a value larger than 0x00 and the supporting node does not support the User Code Command Class, the Authentication ID will be ignored as such a User Code cannot be input at the supporting node.

If a User Code is defined in an Authentication ID, the input of the User Code alone is no longer sufficient to trigger the outcome defined by the User ID Status. All Authentication Data indicated by the Authentication Data IDs in this command MUST be input to trigger the outcome specified by the User ID Status.

Schedule ID (16 bits)

This field is used specify the Schedule during which the Authentication ID is active.

The value 0 MUST indicate that the Authentication ID is active at all times.

Values greater than 0 MUST indicate that the Authentication ID is valid or active only during the corresponding Schedule ID, defined in the Generic Schedule Command Class.

OR logic (1 bit)

This field is used to advertise if OR logic is to be applied for the actual Authentication ID.

Nodes advertising no support for the OR in the Authentication Capability Report Command MUST ignore this field and set this field to 0.

If this field is set to 0, an AND logic MUST be applied for the actual Authentication ID block.

If this field is set to 1, an OR logic MUST be applied for the actual Authentication ID block.

Number Of Authentication Data IDs (7 bits)

This field is used to specify how many Authentication Data ID are contained in the the corresponding Authentication ID Block. Each Authentication Data ID MUST be 16 bits long.

For example, if this field is set to 3, the corresponding Authentication Data ID field must be 6 bytes long.

Authentication Data ID (M x 16 bits)

This field is used to identify the list of Authentication Data ID entries that compose the specified Authentication ID.

The length of this field MUST be according to the *Number Of Authentication Data IDs* field in the corresponding Authentication ID Block.

The first byte MUST carry the most significant byte of the 16 bits.

If any of the Authentication Data ID does not have any Authentication Data defined, the Authentication ID MUST NOT permit any access.

Next Authentication ID (16 bits)

This field is used to advertise the next Authentication ID set at the sending node with non-empty Authentication ID data.

This field MUST be set to the next Authentication ID which has a Fallback or User Identifier status is different from 0.

This field MUST be set to 0x00 if there are no more Authentication IDs are defined after the value of the *Authentication ID* field of the last Authentication ID Block.

2.2.10.11 Authentication Checksum Get Command

This command is used to request checksum(s) representing the current authentication data or combinations set at the supporting node.

A supporting node MUST return a response with the *Checksum* field set to 0x00 if the checksum type or the authentication type is not supported.

The Authentication Checksum Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.59: Authentication Checksum Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_CHECKSUM_GET (0x09)							
Res	Checksum type			Authentication Technology Type			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Checksum Type (3 bits)

This field is used to indicate which checksum to return. This field MUST be encoded according to Table 2.60.

Table 2.60: Authentication Checksum Get::Checksum type encoding

Value	Checksum Type	Version
0x00	Authentication Data checksum	1
0x01	Combination Entries checksum	1
0x02..0x07	Reserved	1

Authentication Technology Type (4 bits)

This field is used to specify which type of the authentication technology to use for the checksum calculation. The field MUST be encoded according to Table 2.52.

This field MUST be set to 0 by a sending node and ignored by a supporting node if the *Checksum Type* field is set to 1.

2.2.10.12 Authentication Checksum Report Command

This command is used to advertise the checksum representing current authentication data that correspond to a specific technology type or combination entries defined at the supporting node

Table 2.61: Authentication Checksum Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION (0xA1)							
Command = AUTHENTICATION_CHECKSUM_REPORT (0x0A)							
Res	Checksum type			Authentication Technology Type			
Checksum 1 (MSB)							
Checksum 2 (LSB)							

Authentication Technology Type (4 bits)

This field is used to advertise which type of the authentication technology has been used for the checksum calculation. The field MUST be encoded according to Table 2.52.

This field MUST be set to 0x00 by the sending node if the Checksum type is set to 0x01.

The other values are reserved and MUST NOT be used by sending node. Reserved values MUST be ignored by a receiving node.

Checksum Type (3 bits)

This field is used to indicate which checksum to return. This field MUST be encoded according to Table 2.60.

Authentication Data Checksum (16 bits)

This field is used to advertise the checksum value. The checksum MUST be calculated using the CRC-CCITT polynomial using initialization value equal to 0x1D0F and 0x1021 (normal representation).

This field MUST be set to 0x0000 if no authentication data or combination entries are defined at sending node.

The calculation of the checksum depends on the *Checksum Type* and *Authentication Technology Type* field values.

If the checksum type field is set to 0x00 Authentication Data checksum:

Similar to the User Code checksum data format (refer to User Code Command Class, version 2), the Authentication Data checksum data MUST be built by concatenating initialized Authentication Data content for a given technology type (with Authentication Data length different than 0x00) in the ascending Data ID order.

If the Authentication Technology Type is set to 0x00, the checksum field MUST be set to 0x00.

Else, for example for RFID based authentication technology, each initialized RFID data MUST be formatted as follows:

Data ID (16 bits) | RFID Data (1..32 bytes)

For example, a node supporting 4 RFID IDs that are set as follow:

- Data ID 1, entry Data Content length 4, RFID code ABC10001
- Data ID 2, entry Data Content length 0
- Data ID 3, entry Data Content length 5, RFID code AB58198301
- Data ID 4, entry Data Content length 0

In this case, the defined RFID data content for Data ID 1 and 3 MUST be concatenated to obtain the checksum data:

0x0001 | 0xAB C1 00 01 | 0x0003 | 0xAB 58 19 83 01

The checksum data MUST be: 0x0001ABC100010003AB58198301 and the checksum field MUST be set to 0x17FE.

If the checksum type field is set to 0x01 Combination Entries checksum:

The checksum data MUST be built by concatenating defined Combination Entries whose status or User Identifier is different than 0x00 in the ascending Authentication ID order. Each entry MUST be comprised of all the fields included in an Authentication ID Block for the version of the supporting node, including reserved fields.

For example, a node supports both User Code and RFID based authentication technology, the Combination Entries checksum calculation MUST be formatted as follows:

- Authentication ID Block Length (8 bits)
- Authentication ID (16 bits)
- Fallback Status (8 bits)
- User Identifier (16 bits)
- Schedule ID (16 bits)
- Number of Authentication Data IDs (8 bits)
- Authentication Data ID (M x 16 bits)

Authentication ID Block Length (8 bits) | Authentication ID (16 bits) | Fallback Status (8 bits) |User Identifier (16 bits) | Schedule ID (16 bits) | Number of Authentication Data IDs (8 bits) | Authentication Data ID (M x 16 bits)

For example, a node supporting 3 Authentication IDs that are set as follow:

- Length 11, Authentication ID 1, Fallback status 0, User Identifier 1, Schedule ID 0, Number of Authentication Data IDs 1, Data Identifier 1.
- Length 9, Authentication ID 2, Fallback status 0, User Identifier 0, Schedule ID 0, Number of Authentication Data IDs 0.
- Length 13, Authentication ID 3, Fallback status 0, User Identifier 200, Schedule ID 1, Number of Authentication Data IDs 2, Data Identifier 4 and 5

In this case, the defined Combination Entries checksum data for Authentication ID 1 and 3 MUST be concatenated to obtain the checksum data:

0x0B | 0x0001 | 0x00 | 0x0001 | 0x0000 | 0x01 | 0x0001 | 0x0D | 0x0003 | 0x00 | 0x00C8 | 0x0001 | 0x02 | 0x0004 | 0x0005

The checksum data MUST be: 0x0B0001000001000001000010D00030000C800010200040005 and the checksum must be 0x0A53.

2.2.10.13 Examples and use-cases

An example of configurations with User Code CC and RFID tags is shown in Figure 2.3.

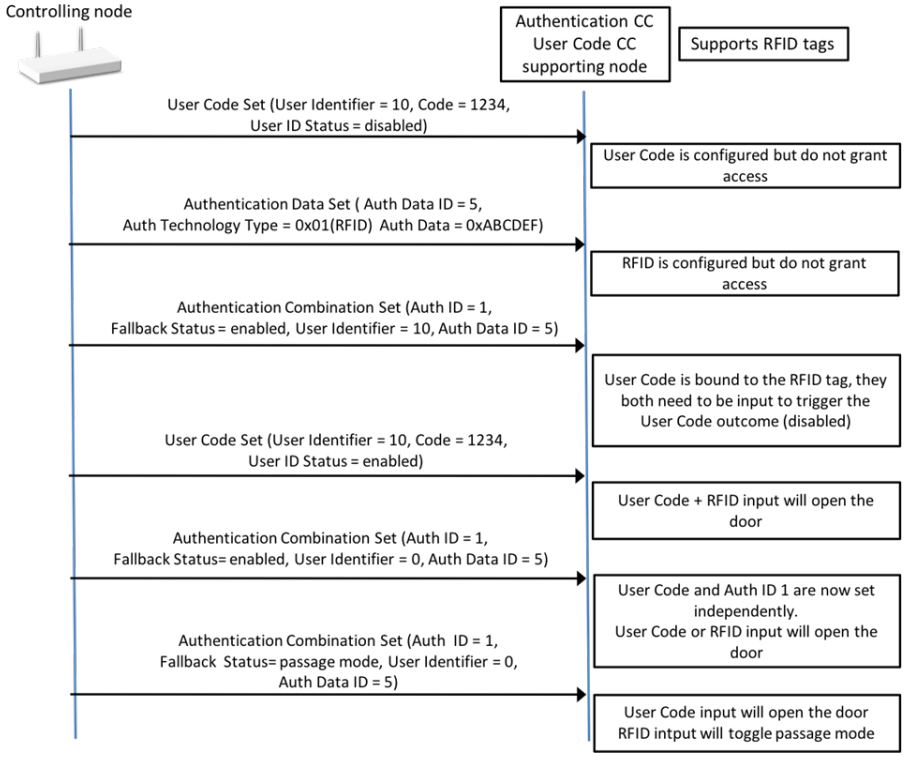


Figure 2.3: Authentication Command Class example: User Code and RFID configuration

An example of Authentication Data configurations with RFID and Magnetic card technology types is shown in Figure 2.4.

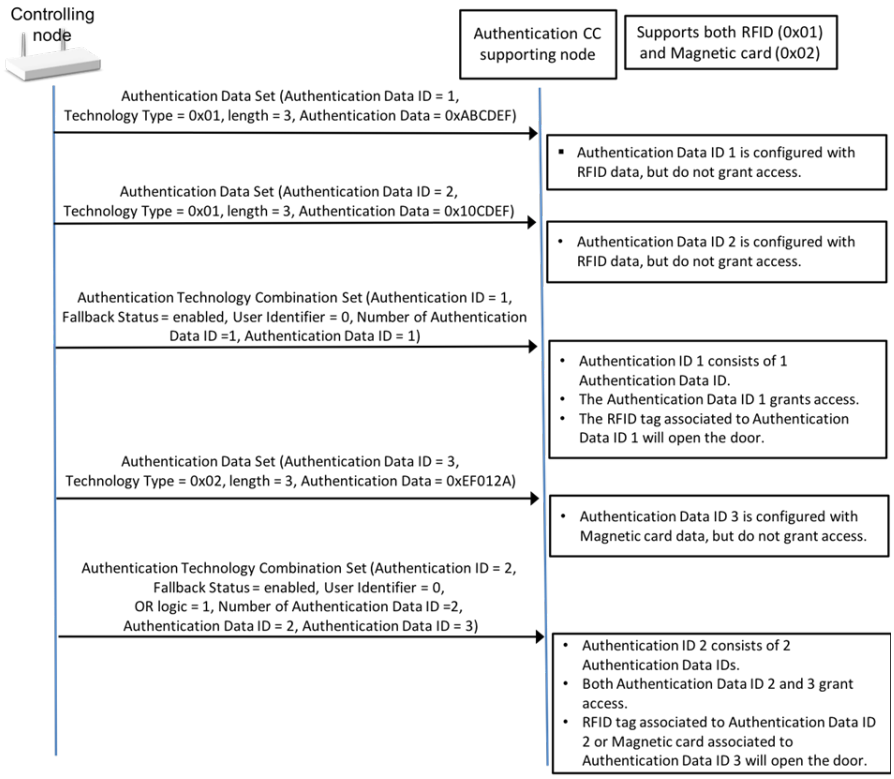


Figure 2.4: Authentication Command Class example: Authentication Data configuration

2.2.11 Authentication Media Write Command Class, version 1 [NEVER CERTIFIED]

Warning: THIS COMMAND CLASS HAS NEVER BEEN CERTIFIED

This command class has never been implemented and certified by a Z-Wave product Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.

Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

This command class is used to write raw Authentication Media data, such as RFID data contents on tags.

The supported authentication medium itself is specific to individual nodes. Supporting nodes can use the Entry Control Command Class to report the corresponding readings of their supported Authentication Media.

2.2.11.1 Authentication Media Capability Get Command

This command is used to request the Authentication Media capabilities supported by the receiving node.

The Authentication Media Capability Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.62: Authentication Media Capability Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION_MEDIA_WRITE (0xA2)							
Command = AUTHENTICATION_MEDIA_CAPABILITY_GET (0x01)							

2.2.11.2 Authentication Media Capability Report Command

This command is used to advertise the Authentication Media Command Class capabilities supported by the sending node.

Table 2.63: Authentication Media Capability Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION_MEDIA_WRITE (0xA2)							
Command = AUTHENTICATION_MEDIA_CAPABILITY_REPORT (0x02)							
Reserved							DGS

DGS (Data Generation support) (1 bit)

This field is used to advertise if the supporting node has the capability to generate authentication data that can be written on its authentication media.

This feature is used if the authentication media has a particular format and would not accept any arbitral hexadecimal data.

If this field is set to 1, the supporting node MUST support creating authentication data for their write operations.

If this field is set to 0, the supporting node MUST NOT support creating authentication data for their write operations.

2.2.11.3 Authentication Media Write Start Command

This command is used to write Authentication Media contents (i.e., RFID data to a tag).

The Authentication Media Write Status Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.64: Authentication Media Write Start Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION_MEDIA_WRITE (0xA2)							
Command = AUTHENTICATION_MEDIA_WRITE_START (0x03)							
Sequence number							
Timeout							
Generate-data	Reserved						
Data Length							
Data 1							
...							
Data N							

Sequence number (8 bits)

This field is used to keep track of the Authentication Media Write Operation session.

A supporting node MUST return a Authentication Media Write Status Command with the status field set to 0x03 if receiving this command with a different sequence number than the ongoing write operation.

Timeout (8 bits)

This field is used to specify how many seconds the receiving node MUST wait between the reception of this command and aborting the write operation.

For nodes with a sensing/feeback capability for their authentication media write operation, this value MUST also be used to wait that a media is accessible to the receiving node before the write operation is aborted.

For Nodes without any feedback capability for their authentication media write operation MUST ignore this timeout value after and try to write the data on the authentication media target immediately.

Generate data (1 bit)

This field is used to indicate the supporting node to generate the Authentication data that will be written on the Authentication Media.

If this field is set to 1, the *Data Length* field MUST be set to 0.

If this field is set to 1, the supporting node MUST generate valid authentication data and return the written data in the *Data* field of the returned response.

Data Length (8 bits)

This field is used to describe the length in bytes of the corresponding *Data* field.

This field MUST be in the range 0..255.

This field MUST be set to a value higher than 0 if the *Generate Data* field is set to 0.

Data (N bytes)

This field is used to advertise the Data contents that MUST be written to the media.

The length of this field in bytes MUST be according to the corresponding *Data Length* field value.

2.2.11.4 Authentication Media Write Stop Command

This command is used to interrupt an ongoing Authentication Media Write operation.

The Authentication Media Write Status Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.65: Authentication Media Write Stop Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION_MEDIA_WRITE (0xA2)							
Command = AUTHENTICATION_MEDIA_WRITE_STOP (0x04)							

2.2.11.5 Authentication Media Write Status Command

This command is used to advertise the status of an Authentication Media Write operation.

Table 2.66: Authentication Media Write Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_AUTHENTICATION_MEDIA_WRITE (0xA2)							
Command = AUTHENTICATION_MEDIA_WRITE_STATUS (0x05)							
Sequence number							
Status							
Data Length							
Data 1							
...							
Data N							

Sequence number (8 bits)

This field is used to advertise the sequence number of the last Write operation.

If an Authentication Media Write Stop Command was received while no operation was ongoing, this field SHOULD be set to 0x00.

Data Length (8 bits)

This field is used to advertise the length in bytes of the *Data* field.

This field MUST be in the range 0..255.

This field MUST be set to a value highest than 0 if the *Status* field is set to 0xFE or 0xFF.

Data (N bytes)

This field is used to advertise the data that was written on the Authentication media as part of the write operation.

This field MUST contain the data that was written on the media if the *Status* field is set to 0xFE or 0xFF.

Status (8 bit)

This field is used to indicate the status of the Authentication Media Write operation. This field MUST encoded according to Table 2.67.

Table 2.67: Authentication Media Write Status::Status encoding

Value	Description
0x00	**Failed: Write operation not supported** The node does not support write operations.
0x01	**Failed: Forced stop** The node has received a Authentication Media Write Stop Command before the data contents could be written on the target.
0x02	**Failed: timeout** The node has timed out and could not write the data content on a target
0x03	**Failed: Another operation is ongoing** The node has did not initiate the write operation as another operation was already ongoing.
0x04	**Failed: Aborted** The node has aborted the operation due to local input or other higher priority task.
0x05	**Failed: Buffer overflow** The node could not start or aborted the write operation as it ran into a buffer overflow while caching the data to write.
0x06	**Failed: Proprietary hex data restriction** The node has received hexadecimal data that does not comply with the required format for its Authentication Media data. The controlling node should use the "Generate-data" flag instead. This status MUST NOT be returned by a node if it does not advertise support for Data Generation in the Authentication Media Capability Report Command
0x07	**Failed: Non re-writable target** The node could not write the data on the target as it already contains some data that could not be overwritten.
0xFE	**Completed with no status:** The Write Operation was carried out, but the node has no feedback from the media writing operation and cannot confirm that the data was properly written on the target.
0xFF	**Success** The specified data was properly written on the target.

2.2.12 Barrier Operator Command Class, version 1

The Barrier Operator Command Class is used to control and query the status of motorized barriers.

2.2.12.1 Compatibility considerations

CC:0066.01.00.23.001 A supporting node MAY ignore a Barrier Operator Command if the requested operation violates operational limitations or safety regulations.

2.2.12.1.1 Node Information Frame (NIF)

CC:0066.01.00.21.001 The Barrier Operator Command Class MUST be supported only using secure communication (S0 and/or S2 Command Class).

CC:0066.01.00.21.003 A supporting node MUST NOT advertise the Barrier Operator Command Class in its NIF after network inclusion.

2.2.12.1.2 Command Class dependencies

CC:0066.01.00.21.002 A node supporting the Barrier Operator Command Class MUST support the Notification Command Class, version 4 or newer.

2.2.12.2 Barrier Operator Set Command

This command is used to initiate an unattended change in state of the barrier.

CC:0066.01.01.13.001 A supporting node MAY ignore this command if the requested operation violates operational limitations or safety regulations.

Table 2.68: Barrier Operator Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_SET (0x01)							
Target Value							

Target Value (8 bits)

CC:0066.01.01.11.001 This field MUST specify the intended state of the device.

CC:0066.01.01.11.002 The encoding of this field MUST be according to Table 2.69.

Table 2.69: Barrier Set::Target Value

Target Value		Description	Version
0x00	CLOSE	Initiate unattended close	1
0xFF	OPEN	Initiate unattended open	1

CC:0066.01.01.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

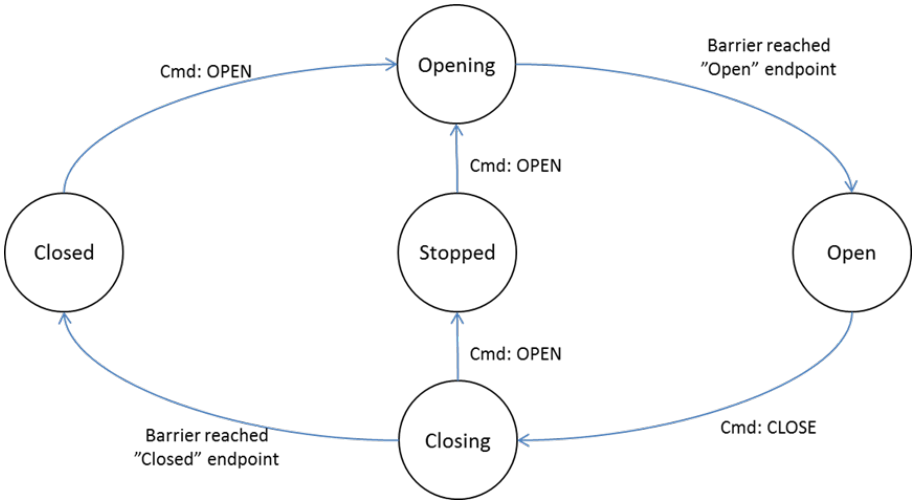


Figure 2.5: Barrier Set, state transitions

CC:0066.01.01.11.004 A receiving node MUST comply with the allowed state changes and triggering commands specified in Figure 2.5. Any state/command pair not specified in Figure 2.5 MUST NOT cause any action.

CC:0066.01.01.12.001 The receiving node SHOULD respond to OPEN and CLOSE commands. However, the receiving node
CC:0066.01.01.11.005 MUST NOT respond to OPEN or CLOSE commands if any safety issue is detected.

CC:0066.01.01.11.006 The receiving node MUST NOT respond to the CLOSE command unless it is in the “Open” state.

2.2.12.2.1 Error Handling

CC:0066.01.01.11.007 The device MUST stop if any safety issue is detected.

CC:0066.01.01.11.008 If the requested operation is overruled by a safety mechanism, the device MUST notify the requester via a Notification Report with the Notification {Access Control::Barrier unable to perform requested operation due to UL requirements}.

CC:0066.01.01.11.009 If the requested operation is prohibited by the specific device class, e.g. SPECIFIC_TYPE_SECURE_BARRIER_OPEN_ONLY or SPECIFIC_TYPE_SECURE_BARRIER_CLOSE_ONLY, the device MUST notify the requester by issuing an Application Rejected Request Command. The Application Rejected Request Command is a member of the Application Status Command Class.

2.2.12.3 Barrier Operator Get Command

This command is used to request the current state of a barrier operator device.

CC:0066.01.02.11.001 The Barrier Operator Report Command MUST be returned in response to this command.

CC:0066.01.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:0066.01.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.70: Barrier Operator Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_GET (0x02)							

2.2.12.4 Barrier Operator Report Command

This command is used to advertise the status of the barrier operator device.

Table 2.71: Barrier Operator Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_REPORT (0x03)							
State							

State (8 bits)

CC:0066.01.03.11.001 This field MUST advertise the current state of the device.

CC:0066.01.03.11.002 The encoding of this field MUST be according to Table 2.72.

Table 2.72: Barrier Report::State

Value	State	Requirement	Description	Version
0x00	Closed	REQUIRED	The barrier is in the Closed position	1
0x01 … 0x63	Stopped at exact Position	RECOMMENDED	0x01 = 1% (Near Closed) 0x63 = 99% (Near Open)	1
0xFC	Closing	REQUIRED	The barrier is closing. The current position is unknown.	1
0xFD	Stopped	REQUIRED	The barrier is stopped. The current position is unknown.	1
0xFE	Opening	REQUIRED	The barrier is opening. The current position is unknown.	1
0xFF	Open	REQUIRED	The barrier is in the Open position	1

CC:0066.01.03.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.12.5 Barrier Operator Get Signaling Capabilities Supported Command

This command is used to query a device for available subsystems which may be controlled via Z-Wave.

CC:0066.01.04.11.001 The Barrier Operator Report Signaling Capabilities Supported Command MUST be returned in response to this command.

CC:0066.01.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0066.01.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.73: Barrier Operator Get Signaling Capabilities Supported Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_SIGNAL_SUPPORTED_GET (0x04)							

2.2.12.6 Barrier Operator Report Signaling Capabilities Supported Command

This command returns a bit mask of signaling subsystem(s) supported by the sending node. The device that issuing this command may not support any signaling subsystem(s).

Table 2.74: Barrier Operator Report Signaling Capabilities Supported Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_SIGNAL_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

This field is used to advertise the event signaling capabilities supported by the sending node. If the device does not support any signaling subsystem(s), the device MAY issue this command with a bitmask felid set to 0x00.

CC:0066.01.05.11.004 It MUST be treated as a bitmask.

1. Bit 0 in Bit Mask 1 indicates if subsystem type 0x01 is supported
2. Bit 1 in Bit Mask 1 indicates if subsystem type 0x02 is supported
3. ...

CC:0066.01.05.11.001 If the subsystem type is supported, the corresponding bit MUST be set to 1.

If the subsystem type is not supported, the corresponding bit MUST be set to 0.

For the complete list of subsystem types, refer to Table 2.76.

CC:0066.01.05.11.002 The length of this field depends on the number of subsystems supported by the sending node. The length of this field MUST be in the range 1..32 bytes.

CC:0066.01.05.11.003 A sending node MUST limit the length of this field to represent only those subsystems supported and MUST NOT extend the length of the Bit Mask to pad the packet with 0x00.

2.2.12.7 Barrier Operator Event Signal Set Command

This command is used to turn on or off an event signaling subsystem that is supported by the device.

CC:0066.01.06.13.001 A supporting node MAY ignore this command if the requested operation violates safety regulations.

CC:0066.01.06.11.004 UL requirements MUST take precedence over any Set command sent to a subsystem. It is up to the discretion of the device to accept or reject the requested action. If the requested action is rejected, the

CC:0066.01.06.11.001 node MUST notify the requester via the Notification Command Class Report with the Notification {Access Control::Barrier unable to perform requested operation due to UL requirements}.

Table 2.75: Barrier Operator Event Signal Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_EVENT_SIGNAL_SET (0x06)							
Subsystem Type							
Subsystem State							

Subsystem Type (8 bits)

CC:0066.01.06.11.002 This field is used to indicate which subsystem type MUST be set at the receiving node. This field MUST comply with Table 2.76.

Table 2.76: Barrier Operator Event signal::Subsystem types encoding

Subsystem Type	Description	Version
0x00	NOT SUPPORTED - reserved	1
0x01	The Barrier Device has an Audible Notification subsystem controllable via Z-Wave (For example: Siren)	1
0x02	The Barrier Device has an Visual Notification subsystem controllable via Z-Wave (For example: Flashing Light)	1
0x03.. 0xFF	Reserved	1

Subsystem State (8 bits)

CC:0066.01.06.11.003 This field is used to indicate the state that the specified subsystem MUST assume at the receiving node. This field MUST comply with Table 2.77.

Table 2.77: Barrier Operator Event Signal::Subsystem states encoding

Subsystem State	Description	Version
0x00	Subsystem OFF	1
...	Reserved	1
0xFF	Subsystem ON	1

2.2.12.8 Barrier Operator Event Signaling Get Command

This command is used to request the state of a signaling subsystem to a supporting node.

CC:0066.01.07.11.001 The Barrier Operator Event Signaling Report Command MUST be returned in response to this command if the specified type is supported.

CC:0066.01.07.11.002 This command MUST NOT be issued via multicast addressing.

CC:0066.01.07.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.78: Barrier Operator Event Signaling Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_EVENT_SIGNALING_GET (0x07)							
Subsystem Type							

For fields' description, refer to Section 2.2.12.7 Barrier Operator Event Signal Set Command.

2.2.12.9 Barrier Operator Event Signaling Report Command

This command is used to indicate the state of a notification subsystem of a Barrier Device.

Table 2.79: Barrier Operator Event Signaling Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR (0x66)							
Command = BARRIER_OPERATOR_EVENT_SIGNALING_REPORT (0x08)							
Subsystem Type							
Subsystem State							

For fields' description, refer to Section 2.2.12.7 Barrier Operator Event Signal Set Command

2.2.13 Basic Command Class, version 1

The Basic Command Class allows a controlling device to operate the primary functionality of a supporting device without any further knowledge.

The Basic Command Class ensures a basic interoperability if no other command class is shared by two devices.

The Basic Command Class is an actuator control command class. Refer to Section 2.1.6.

2.2.13.1 Compatibility considerations

A specific device may not be able to support all Basic CC commands or parameter levels. For instance, a relay can only open fully in response to any non-zero value.

CC:0020.01.00.22.001 Any device SHOULD support the Basic Command Class.

CC:0020.01.00.21.001 A device MUST implement mappings from the Basic Command Class to specific commands according to the advertised generic and specific device class of the Node Info frame. Mappings defined by a Specific Device Class have precedence over the Generic Device Class.

CC:0020.01.00.21.002 For Z-Wave Plus devices, the Basic Command Class MUST be mapped according to the actual Z-Wave Plus Device Type. For further information, refer to [34], Section 7.

The following sections only present frame formats. For details on the mapping to other command classes, refer to [34] and Section 7.

2.2.13.1.1 Node Information Frame (NIF)

CC:0020.01.00.21.003 The Basic Command Class MUST NOT be advertised in the Node Information Frame.

CC:0020.01.00.21.004 The Basic Command Class MUST NOT be advertised in the Security Commands Supported Report (S0 as well as S2)

CC:0020.01.00.21.005 A securely included node MAY support the Basic Command Class at the highest security level but it MUST NOT support the Basic Command Class at any lower security level or non-securely.

2.2.13.2 Basic Set Command

This command is used to set a value in a supporting device.

Table 2.80: Basic Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC (0x20)							
Command = BASIC_SET (0x01)							
Value							

Value (8 bits)

CC:0020.01.01.12.001 A supporting device SHOULD support all parameter values in the range {0x00..0x63, 0xFF}.

CC:0020.01.01.11.001 A controlling device MUST NOT assume that a receiving device reacts to this command.

CC:0020.01.01.11.002 A receiving device MUST interpret Basic Set parameter values according to the requirements of the Device Class implemented by the device. Refer to [34] and Section 7.

2.2.13.3 Basic Get Command

This command is used to request the status of a supporting device.

CC:0020.01.02.11.001 The Basic Report Command MUST be returned in response to this command.

CC:0020.01.02.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.81: Basic Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC (0x20)							
Command = BASIC_GET (0x02)							

2.2.13.4 Basic Report Command

This command is used to advertise the status of the primary functionality of the device.

Table 2.82: Basic Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC (0x20)							
Command = BASIC_REPORT (0x03)							
Value							

Value (8 bits)

CC:0020.01.03.12.001 The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

For details on the mapping of values, refer to [34] and Section 7.

CC:0020.01.03.11.001 A controlling device MUST NOT assume that the Value is identical to a value previously specified in a Set command; not even when a transition has ended.

CC:0020.01.03.11.002 A receiving device MUST interpret the Value field according to Table 2.83.

Table 2.83: Basic Report::Value

Value	Level	State
0 (0x00)	0%	Off
1..99 (0x01..0x63)	1..100%	On
...	Reserved	Reserved
254 (0xFE)	Unknown	Unknown
255 (0xFF)	100%	On

CC:0020.01.03.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.14 Basic Command Class, version 2

The Basic Command Class allows a controlling device to operate the primary functionality of another device without any further knowledge.

The Basic Command Class ensures a basic interoperability if no other Command Class is shared by two devices.

The Basic Command Class is an actuator control command class. Refer to Section 2.1.6.

2.2.14.1 Compatibility considerations

CC:0020.02.00.21.001 A device supporting Basic Command Class, version 2 MUST support Basic Command Class, version 1. Commands not described in this version remain unchanged from version 1.

Version 2 adds the distinction between the Current Value and the Target State of the device.

CC:0020.02.00.21.002 Version 2 elevates the requirement level so that the Current Value field of the Basic Report command MUST advertise the current value of the device hardware; also while in transition to a new target value.

The compatibility considerations from version 1 also apply to this version. Refer to Section 2.2.13.1.

2.2.14.2 Basic Report Command

This command is used to advertise the status of the primary functionality of the device.

Table 2.84: Basic Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC (0x20)							
Command = BASIC_REPORT (0x03)							
Current Value							
Target Value							
Duration							

Current Value (8 bits)

CC:0020.02.03.11.001 The Current Value field MUST advertise the current value of the device hardware; also while in transition to a new target value.

CC:0020.02.03.11.002 The advertised values may vary depending on the device class implemented by the supporting device. A controlling device MUST interpret the value according to Table 21.For details on the mapping of values, refer to [34] and Section 7.

CC:0020.02.03.12.001 The Current Value SHOULD be identical to the Target Value when a transition has ended.

Target Value (8 bits)

CC:0020.02.03.11.003 The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition.

CC:0020.02.03.11.004 The advertised values may vary depending on the device class implemented by the supporting device. A controlling device MUST interpret the value according to Table 2.83.

For details on the mapping of values, refer [34] and Section 7.

CC:0020.02.03.11.005 If queried after receiving a Set command, the Target Value field MUST advertise the target value
CC:0020.02.03.13.001 specified in the Set command. The Target Value MAY change at a later time due to local control or a “Stop” motion control command.

CC:0020.02.03.11.006 If the device is in a motion controlled transition, the Target Value field MUST advertise the min or max value (depending on the direction) according to the command class mappings defined in [34] and Section 7.

Duration (8 bits)

CC:0020.02.03.12.002 The Duration field SHOULD advertise the time needed to reach the Target Value at the actual transition rate. The encoding of the Duration field MUST be according to Table 2.10

2.2.15 Basic Tariff Information Command Class, version 1 [NEVER CERTIFIED]

Warning: THIS COMMAND CLASS HAS NEVER BEEN CERTIFIED

This command class has never been implemented and certified by a Z-Wave product Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.

Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

This Basic Tariff Information Command Class for use with a single element or dual element meter, and for use with import (electricity received from grid) rates only. The command class is kept as simple as possible without any pricing information.

This command class supports a GET and REPORT. No Set command is supported, as it is not appropriate to set any of the parameters through Z-Wave.

2.2.15.1 Basic Tariff Information Get Command

This command is used to request current tariff information from the meter.

The Basic Tariff Information Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.85: Basic Tariff Information Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC_TARIFF_INFO (0x36)							
Command = BASIC_TARIFF_INFO_GET (0x01)							

2.2.15.2 Basic Tariff Information Report Command

This command returns information on the number of import rates supported, and current import rate information. Application can send unsolicited Basic Tariff Report commands or requested by the Basic Tariff Get Information command.

Table 2.86: Basic Tariff Information Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC_TARIFF_INFO (0x36)							
Command = BASIC_TARIFF_INFO_REPORT (0x02)							
Dual	Reserved			Total No. Import Rates			
Reserved				E1 - Current Rate in Use			
E1 - Rate Consumption Register - MSB							
E1 - Rate Consumption Register							
E1 - Rate Consumption Register							
E1 - Rate Consumption Register - LSB							
E1 - Time for Next Rate - Hours							
E1 - Time for Next Rate - Minutes							
E1 - Time for Next Rate - Seconds							
Reserved				E2 - Current Rate in Use			
E2 - Rate Consumption Register - MSB							
E2 - Rate Consumption Register							
E2 - Rate Consumption Register							
E2 - Rate Consumption Register - LSB							

Dual (1 bit)

Single Element = 0, Two Elements = 1.

If single element the E2 fields are skipped in the frame. E1 – Time for Next Rate – Seconds will be the last byte of the message and the number of data bytes will be 9.

If two elements the E2 fields are present and the number of data bytes will be 14.

Total Number of Import Rates Supported (7 bits)

Field specifies the number of import rates E1 (and E2) supported by the meter. Range of legal decimal values are 1…8. No units used. The decimal values 0 and 9…15 are reserved and MUST be ignored by receiving devices.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

E1 - Current Rate in Use (8 bits)

Field specifies the current rate in use. Range of legal decimal values are 1…8. No units used. The decimal values 0 and 9…15 are reserved and MUST be ignored by receiving devices.

E1 - Rate Consumption Register (32 bits)

The meter has a 32-bit consumption register, for the energy used in each rate. This register is the rate consumption register for the current rate in use now on element 1. Units are in Wh.

E1 - Time for Next Rate - Hours (8 bits)

Field specifies the hour value of the time that the rate is due to change on element 1. Range of legal decimal values are 0…23, or 255. The values 24…254 are reserved and MUST be ignored by receiving devices.

E1 - Time for Next Rate - Minutes (8 bits)

Field specifies the minute value of the time that the rate is due to change on element 1. Range of legal decimal values are 0…59, or 255. The decimal values 60…254 are reserved and MUST be ignored by receiving devices.

E1 - Time for Next Rate - Seconds (8 bits)

Field specifies the second value of the time that the rate is due to change on element 1. Range of legal decimal values are 0…59, or 255. The decimal values 60…254 are reserved and MUST be ignored by receiving devices.

NOTE: 255 in each field of the Time to Next Rate specifies no switching time is in use, which is appropriate for single rate meters. 255 is only a legal value if used in all three Time to Next Rate fields.

E2 - Current Rate in Use (8 bits)

Field specifies the current rate in use on element 2. Range of legal decimal values are 1…8. No units used. The decimal values 0 and 9…15 are reserved and MUST be ignored by receiving devices.

E2 - Rate Consumption Register (32 bits)

The meter has a 32-bit consumption register, for the energy used in each rate. This register is the rate consumption register for the current rate in use now on element 2. Units are in Wh.

2.2.16 Basic Window Covering Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST NOT support this command class.

New window covering device implementations SHOULD support the *Window Covering Command Class, version 1*.

This section contains Commands that may be used to control a Basic Window Covering Command Class.

2.2.16.1 Basic Window Covering Start Level Change Command

This command is used to start moving drapes, shades, blinds in a given direction. The speed of the movement is implementation specific.

Table 2.87: Basic Window Covering Start Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC_WINDOW_COVERING (0x50)							
Command = BASIC_WINDOW_COVERING_START_LEVEL_CHANGE (0x01)							
Reserved	Open / Close	Reserved					

Open/Close (1 bit) If the Open/Close bit is set to 0 the window covering SHOULD open. If field is set to 1 the window covering SHOULD close.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.16.2 Basic Window Covering Stop Level Change Command

This command is used to stop moving drapes, shades, blinds in a given direction.

Table 2.88: Basic Window Covering Stop Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC_WINDOW_COVERING (0x50)							
Command = BASIC_WINDOW_COVERING_STOP_LEVEL_CHANGE (0x02)							

2.2.17 Binary Sensor Command Class, version 1 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Notification Command Class.

If implementing this command class, it is RECOMMENDED that the Notification Command Class is also implemented.

The Binary Sensor Command Class is used to realize binary sensors, such as movement sensors.

2.2.17.1 Binary Sensor Get Command

This command is used to request the status of a sensor.

The Binary Sensor Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.89: Binary Sensor Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY (0x30)							
Command = SENSOR_BINARY_GET (0x02)							

2.2.17.2 Binary Sensor Report Command

This command is used to advertise a sensor value.

Table 2.90: Binary Sensor Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY (0x30)							
Command = SENSOR_BINARY_REPORT (0x03)							
Sensor Value							

Sensor Value (8 bits)

If the Sensor Value is 0x00 indicates that the sensor is idle and 0xFF indicates that the sensor has detected an event.

2.2.18 Binary Sensor Command Class, version 2 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the *Notification Command Class, version 3-8*.

If implementing this command class, it is RECOMMENDED that the *Notification Command Class, version 3-8* is also implemented.

The Binary Sensor Command Class is used to realize binary sensors, such as movement sensors and door/window sensors. Version 2 of this command class is extended with the following functionalities:

- A "get-supported" mechanism for the controlling device to interview the binary sensor for its supported sensor types
- A list of defined sensor types capable of reporting a binary value

NOTE: A binary sensor is defined as a sensor unit capable of providing a binary value in the report i.e. that an event was "triggered" or "not triggered" and not other intermediate values. For sensor units providing multiple values the Multilevel Sensor Command Class is more suitable. The Binary Sensor Command Class can further advance through implementation of the Notification Command Class to communicate details about the event.

2.2.18.1 Binary Sensor Get Command

This command is used to request the status of the specific sensor device.

The Binary Sensor Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.91: Binary Sensor Get Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY (0x30)							
Command = SENSOR_BINARY_GET (0x02)							
Sensor Type							

Sensor Type (8 bits)

Sensor Type specifies what type of sensor this command originates from. Refer to Table 2.92. The sensor type value 0xFF returns the first found supported sensor type in the bit mask (starting from bit 0 in Bit Mask 1) by the Binary Sensor Supported Report.

Table 2.92: Binary Sensor Get::Sensor Type encoding

Value	Sensor Type
0x00	Reserved
0x01	General purpose
0x02	Smoke
0x03	CO
0x04	CO2
0x05	Heat
0x06	Water
0x07	Freeze
0x08	Tamper
0x09	Aux
0x0A	Door/Window
0x0B	Tilt
0x0C	Motion
0x0D	Glass Break
0x0E..0xFE	Reserved
0xFF	Return 1st Sensor Type on supported list

2.2.18.2 Binary Sensor Report Command

This command is used to advertise a sensor value.

Table 2.93: Binary Sensor Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY (0x30)							
Command = SENSOR_BINARY_REPORT (0x03)							
Sensor Value							
Sensor Type							

Sensor Value (8 bits)

Sensor Value = 0x00 indicates that the sensor is idle and 0xFF indicates that the sensor has detected an event.

2.2.18.3 Binary Sensor Get Supported Sensor Command

This command is used to request the supported sensor types from the binary sensor device.

Table 2.94: Binary Sensor Get Supported Sensor Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY (0x30)							
Command = SENSOR_BINARY_SUPPORTED_GET_SENSOR (0x01)							

2.2.18.4 Binary Sensor Supported Sensor Report Command

This command must be sent as requested by a received *Binary Sensor Get Supported Sensor Command*. This command indicates the supported sensor types of the binary sensor device in a bit mask format and MUST NOT be transmitted unsolicited.

Table 2.95: Binary Sensor Supported Sensor Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY (0x30)							
Command = SENSOR_BINARY_SUPPORTED_SENSOR_REPORT (0x04)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields describe the supported sensor types by the binary sensor device and refer to the Sensor Type table of Binary Sensor Report command.

- Bit 0 in Bit Mask 1 is not allocated to any Sensor Type and must therefore be set to zero.
- Bit 1 in Bit Mask 1 indicates if Sensor Type = 1 (General Purpose) is supported.
- Bit 2 in Bit Mask 1 indicates if Sensor Type = 2 (Smoke) is supported.
- Bit 3 in Bit Mask 1 indicates if Sensor Type = 3 (CO) is supported
- ...

If the Sensor Type is supported the bit MUST be set to 1. If the Sensor Type is not supported the bit MUST be set to 0. Sensor Type = 0xFF (Return 1st Sensor Type on supported list) cannot be indicated by the Bit Masks.

Note: It is only necessary to transmit Bit Mask 1 and up to the Bit Mask N indicating the last supported sensor type. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

2.2.19 Binary Switch Command Class, version 1

The Binary Switch Command Class is used to control the On/Off state of supporting nodes. The Binary Switch Command Class is an actuator control Command Class. Refer to Section 2.1.6.

2.2.19.1 Binary Switch Set Command

This command is used to set the On/Off state at the receiving node.

CC:0025.01.01.13.001 A receiving node MAY apply a non-zero duration to the transition from one value to a new value.

Table 2.96: Binary Switch Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY (0x25)							
Command = SWITCH_BINARY_SET (0x01)							
Value							

Value (8 bits)

CC:0025.01.01.11.001 This field is used to specify the On/Off state that the receiving node MUST assume. This field MUST be encoded according to Table 2.97.

Table 2.97: Binary Switch Set - Value encoding

Value	Level	State
0 (0x00)	0%	Off
1..99 (0x01..0x63)	100%	On
255 (0xFF)	100%	On

CC:0025.01.01.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0025.01.01.11.003 If the supporting node uses this Command Class, the values MUST be interpreted as follow:

- The value 0 MUST represent the lowest possible throughput (light, water, sound, etc.)
- The values 1..99 and 255 MUST represent the highest possible throughput (light, water, sound, etc.)

The above value mapping of the Binary Switch Command Class Value allows a controlling node to control a mixed group of Binary Switch and Multilevel Switch supporting nodes via Basic Set commands. Nodes supporting the Binary Switch Command Class turn On or Off while nodes supporting the Multilevel Switch Command Class are set at the specified value level.

2.2.19.2 Binary Switch Get Command

This command is used to request the current On/Off state from a node.

CC:0025.01.02.11.001 The *Binary Switch Report Command* MUST be returned in response to this command.

CC:0025.01.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:0025.01.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.98: Binary Switch Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY (0x25)							
Command = SWITCH_BINARY_GET (0x02)							

2.2.19.3 Binary Switch Report Command

This command is used to advertise the current On/Off state at the sending node.

Table 2.99: Binary Switch Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY (0x25)							
Command = SWITCH_BINARY_REPORT (0x03)							
Value							

Value (8 bits)

CC:0025.01.03.11.001 This field is used to specify the On/Off state that the receiving node MUST assume. This field MUST be encoded according to Table 2.100.

Table 2.100: Binary Switch Report - Value encoding

Value	Level	State
0 (0x00)	0%	Off
254 (0xFE)	Unknown	Unknown
255 (0xFF)	100%	On

CC:0025.01.03.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0025.01.03.12.001 This field SHOULD advertise the current value of the node's hardware; also while in transition to a new value.

CC:0025.01.03.11.003 A controlling node MUST NOT assume that this field's value is identical to the value previously issued with a *Binary Switch Set Command*.

2.2.20 Binary Switch Command Class, version 2

The Binary Switch Command Class is used to control the On/Off state of supporting nodes.

The Binary Switch Command Class is an actuator control Command Class. Refer to Section 2.1.6.

2.2.20.1 Compatibility considerations

CC:0025.02.00.21.001 A node supporting Binary Switch Command Class, version 2 MUST support the *Binary Switch Command Class, version 1*.

CC:0025.02.00.21.002 Commands not described in this version MUST remain unchanged from version 1.

Version 2 adds duration and target value control and reporting.

2.2.20.2 Binary Switch Set Command

This command is used to set the binary state at the receiving node.

Table 2.101: Binary Switch Set Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY (0x25)							
Command = SWITCH_BINARY_SET (0x01)							
Target Value							
Duration							

Target Value (8 bits)

CC:0025.02.01.11.001 This field is used to specify the On/Off state that the receiving node MUST assume. This field MUST be encoded according to Table 2.97.

Duration (8 bits)

CC:0025.02.01.12.001 This field is used to specify the duration that the transition from the current value to the Target Value SHOULD take.

CC:0025.02.01.12.002 A supporting node SHOULD respect the specified duration value.

CC:0025.02.01.11.002 The encoding of the Duration value MUST be according to Table 2.9

2.2.20.3 Binary Switch Report Command

This command is used to advertise the current On/Off state at the sending node.

Table 2.102: Binary Switch Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY (0x25)							
Command = SWITCH_BINARY_REPORT (0x03)							
Current Value							
Target Value							
Duration							

Current Value (8 bits)

CC:0025.02.03.11.001 This field is used to advertise the current On/Off state at the sending node. This field MUST be encoded according to Table 2.100.

CC:0025.02.03.11.002 The advertised Current Value MUST NOT be updated to the Target Value before the hardware actuator has actually reached the Target Value.

Target Value (8 bits)

CC:0025.02.03.11.003 This field MUST advertise the target value of an ongoing transition or the most recent transition.

CC:0025.02.03.11.004 This field MUST be encoded according to Table 2.100.

Duration (8 bits)

CC:0025.02.03.12.001 The duration field SHOULD advertise the duration of a transition from the Current Value to the Target Value.

CC:0025.02.03.11.005 The encoding of the Duration field MUST be according to Table 2.10.

2.2.21 Binary Toggle Switch Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

A node MUST NOT implement this Command Class.

New implementations MUST use the *Binary Switch Command Class, version 1*

The Binary Toggle Switch Command Class is used for toggle-style control of binary actuator devices.

2.2.21.1 Binary Toggle Switch Set Command

This command is used to toggle a device e.g. from on to off and from off to on.

Table 2.103: Binary Toggle Switch Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_BINARY (0x28)							
Command = SWITCH_TOGGLE_BINARY_SET (0x01)							

2.2.21.2 Binary Toggle Switch Get Command

This command is used to request the state of the load controlled by the device.

The Binary Toggle Switch Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.104: Binary Toggle Switch Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_BINARY (0x28)							
Command = SWITCH_TOGGLE_BINARY_GET (0x02)							

2.2.21.3 Binary Toggle Switch Report Command

This command is used to advertise the value of a toggle switch.

Table 2.105: Binary Toggle Switch Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_BINARY (0x28)							
Command = SWITCH_TOGGLE_BINARY_GET (0x02)							
Value							

Value (8 bits)

The value MAY be 0x00 (off) or 0xFF (on).

2.2.22 Central Scene Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST support the Central Scene Command Class, version 3 or newer.

CC:005B.01.00.11.001

The Central Scene Command Class is used to communicate central scene activations to a central controller using the lifeline concept. The central scene controller only need to configure lifeline association in relevant nodes before the home control application can take action in the central scene controller. The typical application contains up to 10-15 selected nodes creating a nice "out of the box" experience with minimum effort by the customer. A scene is typically activated via a push button on the device in question.

The Central Scene Command Class can instruct the central scene controller to perform the relevant actions as shown on Figure 2.6.

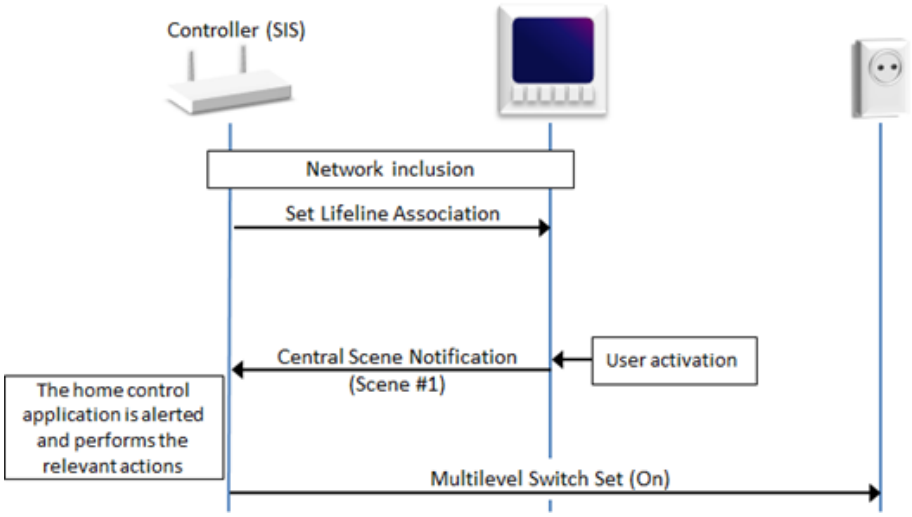


Figure 2.6: Centralized feedback and control

CC:005B.01.00.13.001 Multiple sending nodes MAY advertise the same Scene number.

Therefore, a receiving node MUST interpret the advertised Scene number in combination with the source NodeID of the sending node.

For instance, {Node A, Scene 1} may trigger changes to another scene than {Node B, Scene 1} in the receiving node

Notice that this configuration has a single point of failure in case the central scene controller is broken. The likelihood to experience the "popcorn effect" is increased because the central scene controller may be located far from the push button activated and therefore out of direct range with respect to devices to control.

To compensate for single point of failure in case the central scene controller is broken the devices could do fallback to alternative association groups to control devices directly. These association groups are activated only if the device does not receive acknowledge from configured lifeline. This will ensure some basic level of light control in the house until the central scene controller is restored. Alternatively, a backup central scene controller could be added to the system to take over in case the first one fails.

2.2.22.1 Central Scene Supported Get Command

This command is used to request the maximum number of scenes that this device supports.

CC:005B.01.01.11.001 The Central Scene Supported Report Command MUST be returned in response to this command.

CC:005B.01.01.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.106: Central Scene Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_SUPPORTED_GET (0x01)							

2.2.22.2 Central Scene Supported Report Command

This command is used to report the maximum number of scenes that the requested device supports.

Table 2.107: Central Scene Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_SUPPORTED_REPORT (0x02)							
Supported Scenes							

Supported Scenes (8 bits)

This field indicates the maximum number of scenes supported by the requested device.

CC:005B.01.02.11.001 This field MUST advertise the number of available scenes.

Scenes MUST be numbered in the range 1..[Supported Scenes].

CC:005B.01.02.13.001 A device MAY implement virtual buttons via two-button presses, touch screen swipes or other means.

CC:005B.01.02.11.002 Virtual buttons MUST be numbered after physical push buttons or graphical button objects.

CC:005B.01.02.11.003 The advertised number of Supported Scenes MUST cover physical push buttons, graphical button
CC:005B.01.02.13.002 objects as well as virtual buttons. Therefore, the advertised number of Supported Scenes MAY be larger than the number of physical push buttons or graphical button objects.

2.2.22.3 Central Scene Notification Command

This command is used to report activated scene on device in question including how notification must be interpreted.

In case a lifeline is configured in group #1 using Association Command Class, the device must use this lifeline for transmitting the Central Scene Notification Command. It is allowed to define other association groups handling Central Scene Notification Command and setup rules between the groups. However, lifeline has precedence over other groups handling Central Scene Notification Command in case lifeline is defined.

Table 2.108: Central Scene Notification Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_NOTIFICATION (0x03)							
Sequence Number							
Reserved						Key Attributes	
Scene Number							

Sequence Number (8 bits)

CC:005B.01.03.11.001 The sequence number MUST be incremented each time a Central Scene Notification Command is issued. The receiving device uses the sequence number to ignore duplicates.

Key Attributes (3 bits)

CC:005B.01.03.11.002 The key Attributes field specifies the state of the key. The field MUST be encoded according to Table 2.109

Table 2.109: Central Scene Notification::Key Attributes

Key Attribute	Description
0x00	Key Pressed
0x01	Key Released
0x02	Key Held Down

CC:005B.01.03.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:005B.01.03.11.004 Notifications carrying these Key Attributes MUST comply with Figure 2.7

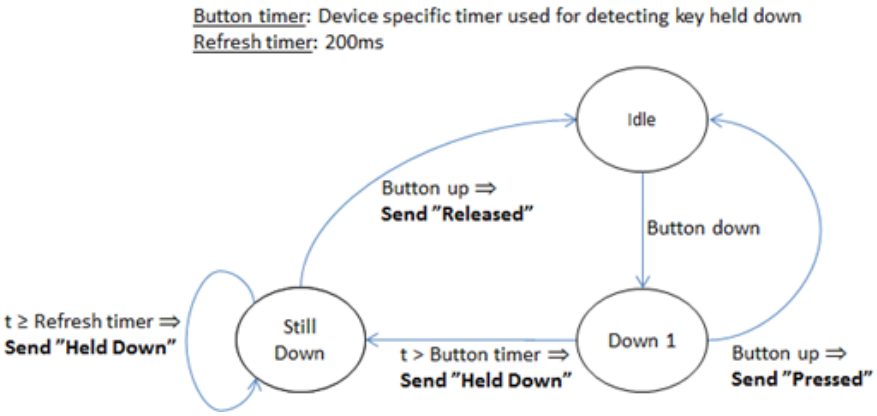


Figure 2.7: Central Scene version 1 button press decoding and timer management

Reserved

CC:005B.01.03.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Scene Number (8 bits)

CC:005B.01.03.11.006 The advertised number of Supported Scenes MUST cover physical push buttons, graphical button
CC:005B.01.03.13.001 objects as well as virtual buttons. Therefore, the advertised number of Supported Scenes MAY be larger than the number of physical push buttons or graphical button objects.

Figure 2.8: Scene number mapping to button layout

2.2.23 Central Scene Command Class, version 2 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST support the *Central Scene Command Class, version 3* or newer.

CC:005B.02.00.11.001

The Central Scene Command Class is used to communicate central scene activations to a central controller using the lifeline concept.

2.2.23.1 Compatibility considerations

Central Scene Command Class version 2 is extended on the following areas:

- The Central Scene Supported Report Command is updated to advertise Key Attributes support for each scene.
- Additional Key Attributes are defined.

Commands and paragraphs not mentioned in this version remain unchanged from version 1.

2.2.23.2 Central Scene Supported Report Command

This command is used to report the maximum number of supported scenes and the Key Attributes supported for each scene.

Table 2.110: Central Scene Supported Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_SUPPORTED_REPORT (0x02)							
Supported Scenes							
Reserved					Number of Bit Mask Bytes		Identical
Supported Key Attributes for Scene 1 - Byte 1							
...							
Supported Key Attributes for Scene 1 - Byte N							
...							
Supported Key Attributes for Scene M - Byte 1							
...							
Supported Key Attributes for Scene M - Byte N							

Fields not described below remain unchanged from version 1.

Reserved

CC:005B.02.02.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Bit Mask Bytes (2 bits)

This field advertises the size of each "Supported Key Attributes" field measured in bytes.

The value MUST be in the range 1..3.

CC:005B.02.02.11.002

Identical (1 bit)

This field indicates if all scenes are supporting the same Key Attributes:

CC:005B.02.02.11.003 The value 1 MUST indicate that all scenes support the same set of Key Attributes. In this case, the field "Supported Key Attributes for Scene 1" MUST advertise the supported Key Attributes for all scenes

CC:005B.02.02.11.004

The value 0 MUST indicate that scenes support different Key Attributes. In this case, Supported Key Attributes MUST be advertised for each individual scene.

Supported Key Attributes (N bytes)

CC:005B.02.02.11.005 This Bit Mask field advertises the attributes supported by the corresponding scene. The field MUST be encoded according to Table 2.111

Table 2.111: Central Scene Supported Report::Supported Key Attributes

Byte:Bit	Supported Key Attributes
1:0	Key Pressed 1 time
1:1	Key Released.
1:2	Key Held Down.
1:3	Key Pressed 2 times
1:4	Key Pressed 3 times
1:5	Key Pressed 4 times
1:6	Key Pressed 5 times
1:7	Reserved

CC:005B.02.02.11.006 If the Key Released attribute is supported, the Key Held Down attribute MUST also be supported.

If the Key Held Down attribute is supported, the Key Released attribute MUST also be supported.

2.2.23.3 Central Scene Notification Command

This command is used to advertise one or more key events.

If the device implements Z-Wave Plus Lifeline support is implemented, the device MUST send the Central Scene Notification Command to the actual Lifeline targets.

Table 2.112: Central Scene Notification Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_NOTIFICATION (0x03)							
Sequence Number							
Reserved					Key Attributes		
Scene Number							

Fields not described below remain unchanged from version 1.

Key Attributes (3 bits)

CC:005B.02.03.11.002 This field advertises one or more events detected by the key. The field MUST be encoded according to Table 2.113

Table 2.113: Central Scene Notification::Key Attributes

Key Attribute	Description	Version
0x00	Key Pressed	1
0x01	Key Released	1
0x02	Key Held Down	1
0x03	Key Pressed 2 times	2
0x04	Key Pressed 3 times	2
0x05	Key Pressed 4 times	2
0x06	Key Pressed 5 times	2

CC:005B.02.03.11.003

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:005B.02.03.11.004 Notification carrying these Key Attributes MUST comply with Figure 2.9

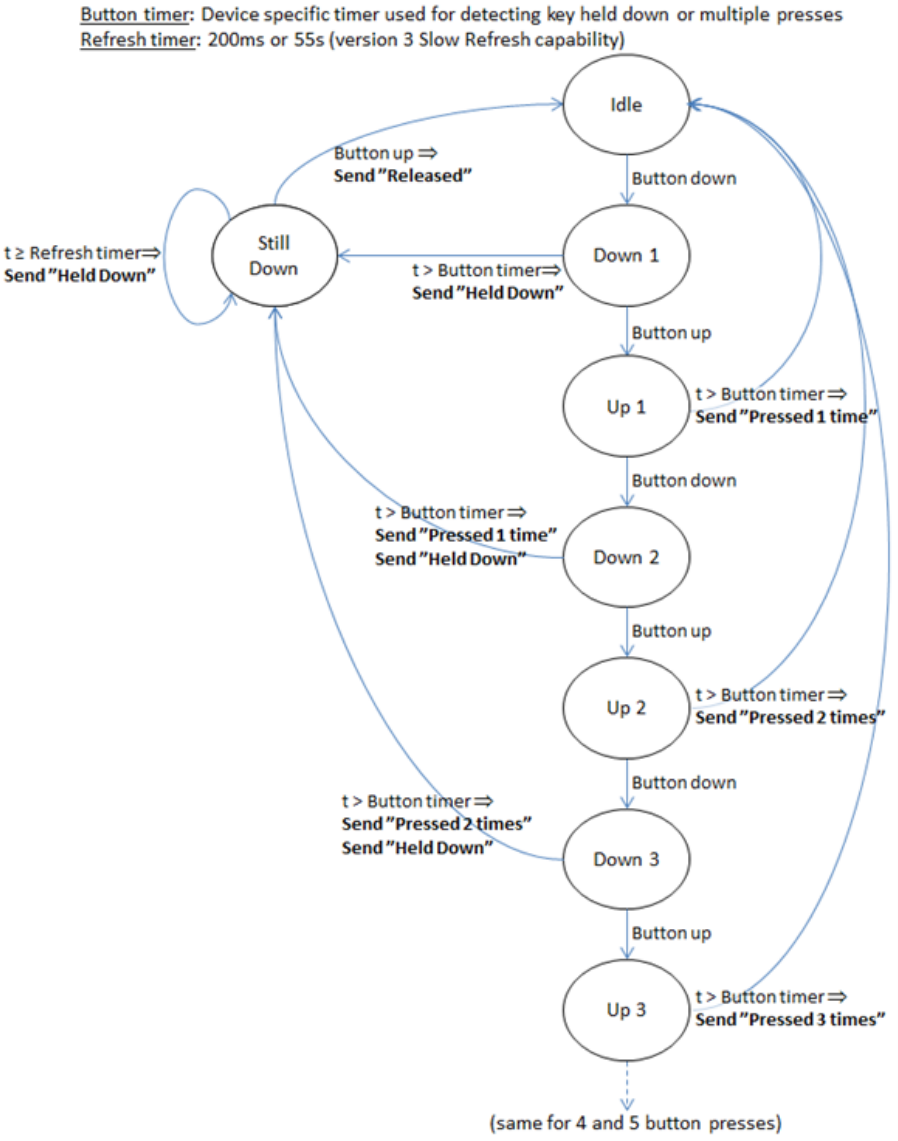


Figure 2.9: Central Scene button press decoding and timer management

2.2.24 Central Scene Command Class, version 3

The Central Scene Command Class, version 3 extends version 2 by adding a new mode for the transmission of Key Held Down notifications. This allows a node to send Key Held Down refreshes at a low rate for the duration of the key held down.

The purpose of this new feature is to provide improved compatibility with FLiRS beaming and network routing.

Central Scene Command Class, version 3 introduces three new commands to allow a controller to query and configure node capabilities.

- Central Scene Configuration Set Command
- Central Scene Configuration Get Command
- Central Scene Configuration Report Command

Central Scene Command Class, version 3 extends the following commands for a scene launching node to advertise its optional capabilities.

- Central Scene Supported Report Command
- Central Scene Notification Command

2.2.24.1 Compatibility considerations

CC:005B.03.00.22.001 A scene controller SHOULD NOT try to detect multiple key presses from the reception of successive key press notifications in any single or multiple key attribute combination. FLiRS beaming, collisions or routing may add delays between messages and render the timing between notifications unreliable.

Version 1 and version 2 required to send Key Held Down notifications commands every 200ms. It has been found that this requirement cannot be observed when the network is using FLiRS beaming or routing. Further, issuing the 200ms refreshes also degrades general network reliability and availability.

Version 3 introduces the Slow Refresh Capability, which allows a node to send refreshes every 55 seconds instead of 200ms. For details about the Slow Refresh capability, refer to Section 2.2.24.7.

CC:005B.03.00.21.001 In order to ensure a functioning network, the Slow Refresh capability MUST be enabled by default after inclusion.

CC:005B.03.00.22.002 When creating associations from a version 3 node, the node creating the association SHOULD enable the Slow Refresh capability in the node if the association destination does also support Central Scene Command Class, version 3. When using the version 3 Slow Refresh capability, the controller relies on the reception of a Key Up notification instead of the Key Held Down refreshes. The new mode still sends refreshes at a low rate, for the controller to detect if a node failed after sending a Held Down Key notification. This means that a version 2 controlling node may time out when receiving Key Held Down Notifications from a version 3 node.

However, it has been found that there exist nodes supporting version 1 or version 2 which do not send Central Scene Notification refreshes every 200ms when the Key Held Down notification is issued.

CC:005B.03.00.22.003 A controller SHOULD apply an adaptive approach based on the reception of the Key Released Notification. Initially, the controller SHOULD time out if not receiving any Key Held Down Notification refresh after 400ms and consider this to be a Key Up Notification. If, however, the controller subsequently receives a Key Released Notification, the controller SHOULD consider the sending node to be operating with the Slow Refresh capability enabled.

2.2.24.1.1 Central Scene Configuration commands support

If a node supports Slow Refresh Capability (that allows a node to send refresh every 55 seconds instead of 200 ms), may not allow users to turn off the 'Slow Refresh' functionality. If this is the case, the node MAY choose to ignore the Central Scene Configuration Set Command with 'Slow Refresh' bit set to zero.

If the node chooses to ignore the Central Scene Configuration Set Command, the node MUST return an application Reject Request Command when receiving Central Scene Configuration Set command (if it received without Supervision encapsulation).

2.2.24.1.2 Multi Channel considerations

CC:005B.03.00.22.004 Multi Channel End Points SHOULD NOT support the Central Scene Command Class.

2.2.24.2 Central Scene Configuration Set Command

This command is used to configure the use of optional node capabilities for scene notifications.

Table 2.114: Central Scene Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_CONFIGURATION_SET (0x04)							
Slow Refresh	Reserved						

Slow Refresh (1 bit)

This flag is used to configure the use of the Slow Refresh capability.

CC:005B.03.04.11.001 The value 1 MUST indicate that the scene launching node MUST use Slow Refresh.

The value 0 MUST indicate that the scene launching node MUST NOT use Slow Refresh.

Reserved

CC:005B.03.04.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.24.3 Central Scene Configuration Get Command

This command is used to query the configuration of optional node capabilities for scene notifications.

CC:005B.03.05.11.001 The Central Scene Configuration Report Command MUST be returned in response to this command.

CC:005B.03.05.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.115: Central Scene Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_CONFIGURATION_GET (0x05)							

2.2.24.4 Central Scene Configuration Report Command

This command is used to advertise the configuration of optional node capabilities for scene notifications.

Table 2.116: Central Scene Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_CONFIGURATION_REPORT (0x06)							
Slow Refresh	Reserved						

Slow Refresh (1 bit)

This flag is used to advertise the use of the Slow Refresh capability.

CC:005B.03.06.11.001 The value 1 MUST indicate that the scene launching node MUST use Slow Refresh.

The value 0 MUST indicate that the scene launching node MUST NOT use Slow Refresh capability.

Reserved

CC:005B.03.06.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.24.5 Central Scene Supported Get Command

This command is used to request the maximum number of scenes that this device supports.

CC:005B.03.01.11.001 The Central Scene Supported Report Command MUST be returned in response to this command.

CC:005B.03.01.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.117: Central Scene Supported Get Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_SUPPORTED_GET (0x01)							

2.2.24.6 Central Scene Supported Report Command

This command is used to report the maximum number of supported scenes and the Key Attributes supported for each scene.

Table 2.118: Central Scene Supported Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_SUPPORTED_REPORT (0x02)							
Supported Scenes							
Slow Refresh Support	Reserved				Number of Bit Mask Bytes		Identical
Supported Key Attributes for Scene 1 - Byte 1							
...							
Supported Key Attributes for Scene 1 - Byte N							
...							
Supported Key Attributes for Scene M - Byte 1							
...							
Supported Key Attributes for Scene M - Byte N							

All fields not described below are the same as in version 2.

Slow Refresh Support (1 bit)

This field indicates whether the node supports the Slow Refresh capability.

CC:005B.03.02.11.001 The value 1 MUST indicate that the node supports the Slow Refresh capability.

The value 0 MUST indicate that the node does not support the Slow Refresh capability.

2.2.24.7 Central Scene Notification Command

This command is used to advertise a key event.

CC:005B.03.03.11.001 If the device implements Z-Wave Plus Lifeline support, the device MUST send the Central Scene Notification Command to the actual Lifeline targets.

Table 2.119: Central Scene Notification Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE (0x5B)							
Command = CENTRAL_SCENE_NOTIFICATION (0x03)							
Sequence Number							
Slow Refresh	Reserved				Key Attribute		
Scene Number							

All fields not described below are the same as in version 2.

Slow Refresh (1 bit)

This flag is used to advertise if the node is sending Key Held Down notifications at a slow rate. A
CC:005B.03.03.11.002 sending node MUST always set this field according to the configured mode.

CC:005B.03.03.11.003 A receiving node MUST ignore this field if the command is not carrying the Key Held Down key attribute.

If the Slow Refresh field is 0:

CC:005B.03.03.12.002 - A new Key Held Down notification SHOULD be sent every 200ms until the key is released.

CC:005B.03.03.11.005 - The Sequence Number field MUST be updated at each notification transmission.

CC:005B.03.03.12.001
- If not receiving a new Key Held Down notification within 400ms, a controlling node SHOULD use an adaptive timeout approach as described in Section 2.2.24.1.

If the Slow Refresh field is 1:

CC:005B.03.03.11.006
- A new Key Held Down notification MUST be sent every 55 seconds until the key is released.

CC:005B.03.03.11.007
- The Sequence Number field MUST be updated at each notification refresh.

CC:005B.03.03.11.008
- If not receiving a new Key Held Down notification within 60 seconds after the most recent Key Held Down notification, a receiving node MUST respond as if it received a Key Release notification.

2.2.25 Climate Control Schedule Command Class, version 1 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the *Schedule Command Class, version 1*.

If implementing this command class, it is RECOMMENDED that the *Schedule Command Class, version 1* is also implemented.

The Climate Control Schedule Command Class allows devices to exchange schedules and overrides, which specify when to perform a setback on the setpoint.

Note: The setpoint is the temperature a device will try to maintain. The setback is a deviation from the setpoint. When a setback is in use the device will apply the setback to the setpoint, resulting in a different temperature. When using schedules and overrides it is possible to define several setbacks occurring at specific times.

Schedules are exchanged using the Schedule Commands.

Overrides of schedules are exchanged using the Schedule Override Commands.

Detection of updated schedules is done using the Schedule Changed Commands.

The Climate Control Schedule uses the Schedule State type to define each setback. The Schedule State type has the following format:

Table 2.120: Climate Control Schedule Command Class

7	6	5	4	3	2	1	0
Schedule State							

Schedule State (8 bits)

The values are as follows:

Table 2.121: Climate Control Schedule Command Class::Schedule State

Schedule State		Description
Hexadecimal	Decimal	
0x80	-128	The setback in 1/10 degrees (Kelvin)
...	...	Example:
0xFF	-1	0 = 0 degrees setback
0x00	0	1 = 0.1 degrees is added to the setpoint
0x01	1	2 - 0.2 degrees is added to the setpoint
...	...	-1 = 0.1 degrees is subtracted from the setpoint
0x78	120	-2 = 0.2 degrees is subtracted from the setpoint
0x79	121	Frost Protection
0x7A	122	Energy Saving Mode
0x7B - 0x7E	123 - 126	Reserved
0x7F	127	Unused State

When converting between Celsius and Fahrenheit proper rounding MUST be applied with at least two decimals in the internal calculations of a device to avoid rounding errors.

When displaying converted Fahrenheit values it is RECOMMENDED that the displayed value is rounded to nearest quarter of a degree.

2.2.25.1 Schedule Set Command

This command is used to set the climate control schedule in a device for a specific weekday. A climate control schedule defines when to use a setback on the setpoint in a device. A schedule can hold a maximum of 9 switchpoints. A switchpoint defines one setback from the current setpoint.

The entire list of switchpoints in the Command MUST be ordered by time, ascending from 00:00 towards 23:59. Switchpoints which have a Schedule State set to "Unused" MUST be placed last. Duplicates MUST NOT be allowed for Switchpoints which have a Schedule State different from "Unused".

Table 2.122: Schedule Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_SET (0x01)							
Reserved					Weekday		
Switchpoint 0 Byte 1							
Switchpoint 0 Byte 2							
Switchpoint 0 Byte 3							
...							
Switchpoint 8 Byte 1							
Switchpoint 8 Byte 2							
Switchpoint 8 Byte 3							

Weekday (3 bits)

This field MUST be encoded according to Table 2.123.

Table 2.123: Climate Control Schedule Set::Weekday encoding

Value	Description
0x00	Reserved
0x01	Monday
0x02	Tuesday
0x03	Wednesday
0x04	Thursday
0x05	Friday
0x06	Saturday
0x07	Sunday

Switchpoint (9*24 bits)

Table 2.124: Switchpoint

7	6	5	4	3	2	1	0		
Reserved			Hour						Byte 1
Reserved		Minute							Byte 2
Schedule State									Byte 3

Hour (5 bits)

This field is used to specify the hour of the of the actual switchpoint. This field MUST be in the range 0..23.

Minute (6 bits)

This field is used to specify the minute of the actual switchpoint. This field MUST be in the range 0..59.

Schedule State (8 bits)

Schedule State uses the Schedule State type format, see Section 2.2.25. If Schedule State has the value of "Unused", the Hour and Minute field MUST be ignored. Once a Schedule State of "Unused" is encountered, the parsing of switchpoints must stop.

2.2.25.2 Schedule Get Command

This command is used to request the climate control schedule in a device for a specific weekday.

The Schedule Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.125: Schedule Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_GET (0x02)							
Reserved					Weekday		

For fields' description, refer to Section 2.2.25.1 Schedule Set Command.

2.2.25.3 Schedule Report Command

This command is used to report the climate control schedule in a device for a specific weekday.

Table 2.126: Schedule Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_REPORT (0x03)							
Reserved					Weekday		
Switchpoint 0 Byte 1							
Switchpoint 0 Byte 2							
Switchpoint 0 Byte 3							
...							
Switchpoint 8 Byte 1							
Switchpoint 8 Byte 2							
Switchpoint 8 Byte 3							

For fields' description, refer to Section 2.2.25.1 Schedule Set Command

2.2.25.4 Schedule Changed Get Command

This command is used to check if the climate control schedule has changed.

The Schedule Changed Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.127: Schedule Changed Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_CHANGED_GET (0x04)							

2.2.25.5 Schedule Changed Report Command

This command is used to report if the climate control schedule has changed.

Table 2.128: Schedule Changed Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_CHANGED_REPORT (0x05)							
ChangeCounter							

ChangeCounter (8 bits)

The ChangeCounter is a timestamp for a climate control schedule and it is kept in devices which exchange climate control schedules.

One device holds a climate control schedule and other devices uses this climate control schedule. Whenever the climate control schedule changes, the device MUST update its ChangeCounter, and the other devices MUST regularly use the Schedule Changed Get Command on the device which holds the Climate Control Schedule to see if the ChangeCounter is different from last time – indicating a change in a climate control schedule.

The possible values are:

Table 2.129: Schedule Changed Report Command::ChangeCounter encoding

Value	Description
0x00	The climate control schedule change mechanism is temporarily disabled by the override function.
0x01..0xFF	The climate control schedule change mechanism is enabled. ChangeCounter is incremented by one every time climate control schedule changes. When ChangeCounter eventually reach 0xFF, the next increment MUST rollover to 0x01.

When a device is fresh and has no climate control schedule it MUST retrieve a climate control schedule using the Schedule Get Command and it MUST also use the Schedule Changed Get to get the first copy of the current ChangeCounter, thus avoiding getting the climate control schedule initially twice.

When a device is awake after sleep mode it SHOULD use this Command to detect if the schedule has been changed.

2.2.25.6 Schedule Override Set Command

This command is used to set the override in a device.

The purpose of an override is to inform a device to ignore its current climate control schedule and assume the setting provided by the Override Type and Override State fields.

Table 2.130: Schedule Override Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_OVERRIDE_SET (0x06)							
Reserved						Override Type	
Override State							

Override Type (2 bits)

This field MUST be encoded according to Table 2.131.

Table 2.131: Climate Control Schedule Override Set::Override Type encoding

Value	Description
0x00	No override
0x01	Temporary override
0x02	Permanent override
0x03	Reserved

Note: The difference between a temporary and a permanent override is that a temporary override only overrides the current switchpoint in the climate control schedule.

Both temporary and permanent overrides MAY be cancelled in the device, which receives the SCHEDULE_OVERRIDE_SET. This cancellation MUST be notified in an unsolicited SCHEDULE_OVERRIDE_REPORT as specified in the following sequence diagram:

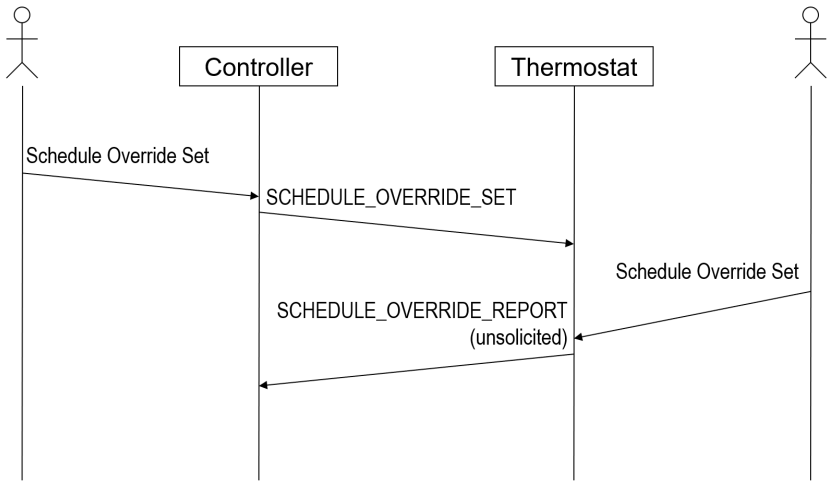


Figure 2.10: Sequence diagram for cancellation of a Schedule Override Set

Override State (8 bits)

The Override State uses the Schedule State type format, see Section 2.2.25

2.2.25.7 Schedule Override Get Command

This command is used to request the override, currently in use in a device.

The Schedule Override Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.132: Schedule Override Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_OVERRIDE_GET (0x07)							

2.2.25.8 Schedule Override Report Command

The Schedule Override Report Command is used to report the override, currently in use in a device.

Table 2.133: Schedule Override Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE (0x46)							
Command = SCHEDULE_OVERRIDE_REPORT (0x08)							
Reserved						Override Type	
Override State							

Override Type (2 bits)

This field MUST be encoded according to Table 2.131.

Both temporary and permanent overrides MAY be cancelled in the device, which receives the SCHEDULE_OVERRIDE_SET. This cancellation MUST be notified in an unsolicited SCHEDULE_OVERRIDE_REPORT as shown on figure in Section 2.2.25.6.

Override State (8 bits)

The Override State uses the Schedule State type format, see Section 2.2.25

2.2.26 Clock Command Class, version 1

The Clock Command Class is used to implement a simple clock functionality.

2.2.26.1 Interoperability considerations

CC:0081.01.00.32.001 A controlling node SHOULD configure the current time and weekday in a supporting node during node commissioning.

It has been found that some version 1 nodes issue Clock Get Command to controllers in order to learn the current time and use the returned Clock Report Command as a Clock Set Command.

CC:0081.01.00.32.002 A supporting node SHOULD NOT learn the current time with a Clock Get Command.

CC:0081.01.00.32.003 A node SHOULD control the Time Command Class to read the current time from supporting nodes (time servers) or support the Clock Command Class and wait to be set the current time by a controlling node.

CC:0081.01.00.32.004 A Z-Wave Plus node SHOULD issue a Clock Report Command via the Lifeline Association Group if they suspect to have inaccurate time and/or weekdays (e.g. after battery removal).

CC:0081.01.00.32.005 A controlling node SHOULD compare the received time and weekday with its current time and set the time again at the supporting node if a deviation is observed (e.g. different weekday or more than a minute difference)

2.2.26.2 Multi Channel Considerations

CC:0081.01.00.12.001 Multi Channel End Points SHOULD NOT support the Clock Command Class.

2.2.26.3 Clock Set Command

This command is used to set the current time in a supporting node.

Table 2.134: Clock Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLOCK (0x81)							
Command = CLOCK_SET (0x04)							
Weekday			Hour				
Minute							

Weekday (3 bits)

CC:0081.01.04.11.001 This field is used to indicate the current weekday that the receiving node MUST assume.

CC:0081.01.04.11.002 This field MUST comply with Table 2.135.

Table 2.135: Clock Set Command::Weekday encoding

Value	Description
0x00	Unused/unknown This value may be specified if the weekday is not used or not known
0x01	Monday
0x02	Tuesday
0x03	Wednesday
0x04	Thursday
0x05	Friday
0x06	Saturday
0x07	Sunday

Hour (5 bits)

CC:0081.01.04.11.003 This field is used to indicate the hour of the current time that the receiving node MUST assume.

CC:0081.01.04.11.004 This field MUST be in the range 0..23.

Minute (8 bits)

CC:0081.01.04.11.005 This field is used to indicate the minute of the current time that the receiving node MUST assume.

CC:0081.01.04.11.006 This field MUST be in the range 0..59.

CC:0081.01.04.12.001 A sending node knowing the current time with seconds precision SHOULD round its current time to the nearest minute when sending this command.

2.2.26.4 Clock Get Command

This command is used to request the current time set at a supporting node.

CC:0081.01.05.11.001 The Clock Report Command MUST be returned in response to this command.

CC:0081.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:0081.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.136: Clock Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLOCK (0x81)							
Command = CLOCK_GET (0x05)							

2.2.26.5 Clock Report Command

This command is used to advertise the current time set at the sending node.

Table 2.137: Clock Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLOCK (0x81)							
Command = CLOCK_REPORT (0x06)							
Weekday			Hour				
Minute							

For fields' description, refer to *Clock Set Command*

CC:0081.01.06.11.001 A sending node MUST comply with fields' description from Clock Set Command

2.2.27 Color Switch Command Class, version 1

The Color Switch Command Class is used to control color capable devices.

The Color Switch Command Class manipulates the color components of a device. Each color component is scaled by the brightness level previously set by a Multilevel Switch Set, Binary Switch Set or Basic Set Command.

The Color Switch Command Class is an actuator control command class. Refer to Section 2.1.6.

2.2.27.1 Compatibility considerations

The Color Switch Command Class, version 1 was previously named the Color Control Command Class, version 1. The Color Switch Command Class, version 1 is binary compatible with the Color Control Command Class, version 1.

2.2.27.1.1 Command Class dependencies

CC:0033.01.00.21.001 Nodes supporting the Color Switch Command Class, version 1 MUST support one of the following command classes in order to control the color brightness:

- Multilevel Switch Command Class, version 2
- Binary Switch Command Class, version 2.

2.2.27.2 Interoperability considerations

CC:0033.01.00.31.001 The Color Switch Command Class MUST be treated as a separate Command Class.

CC:0033.01.00.31.002 Basic and Binary/Multilevel Switch Command Class commands MUST NOT affect color component levels controlled by the Color Switch Command Class.

CC:0033.01.00.31.003 The Color Switch Command Class commands MUST NOT affect the brightness level controlled by Basic or Binary/Multilevel Switch Command Classes.

CC:0033.01.00.31.004 If the brightness level is requested via a Basic Get or Binary/Multilevel Switch Get command, the reported value MUST reflect the brightness level previously set via Basic or Multilevel Switch commands.

CC:0033.01.00.31.005 If a color component is requested via the Color Switch Get command, the reported value MUST reflect the color component level previously set via Color Switch commands.

CC:0033.01.00.31.006 An implementation MUST scale color component levels by the brightness level . Thus, to achieve a resulting light level of 100% for a given color component, the color component level must be set to 100% (via the Color Switch Command Class) and the brightness level must be set to 100% (via Basic or Multilevel Switch Command Class).

CC:0033.01.00.32.001 A controlling device SHOULD specify the highest possible color component levels of a given color tone to achieve the highest light yield when scaled by the brightness level.

CC:0033.01.00.33.001 CC:0033.01.00.31.007 The controlling device MAY however limit color component levels to achieve the same brightness for blended color tones as for pure colors. For this reason, a supporting device MUST NOT normalize color components manipulated by the Color Switch Command Class.

Likewise, a supporting device implementing indexed colors should reduce its internal color component levels for blended colors to achieve the same brightness for all color tones.

2.2.27.3 Color Switch Supported Get Command

This command is used to request the supported color components of a device.

CC:0033.01.01.11.001 The Color Switch Supported Report command MUST be returned in response to this command.

CC:0033.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:0033.01.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.138: Color Switch Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_SUPPORTED_GET (0x01)							

2.2.27.4 Color Switch Supported Report Command

This command is used to report the supported color components of a device.

Table 2.139: Color Switch Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_SUPPORTED_REPORT (0x02)							
Color Component Mask 1							
Color Component Mask 2							

Color Component Mask (variable length)

CC:0033.01.02.11.001 The Color Component Mask field MUST advertise the color components supported by the device.

- Bit 0 in Bit Mask 1 indicates if color component 0 is supported
- Bit 1 in Bit Mask 1 indicates if color component 1 is supported
- …

For the definition of Color Component IDs, refer to section Section 2.2.27.7.

2.2.27.5 Color Switch Get Command

This command is used to request the status of a specified color component.

CC:0033.01.03.11.001 The Color Switch Report Command MUST be returned in response to this command.

CC:0033.01.03.11.002 This command MUST NOT be issued via multicast addressing.

CC:0033.01.03.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.140: Color Switch Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_GET (0x03)							
Color Component ID							

Color Component ID (8 bits)

CC:0033.01.03.11.004 This field MUST specify the color component for which the status is requested.

For the definition of Color Component IDs, refer to section Section 2.2.27.7.

2.2.27.6 Color Switch Report Command

This command is used to advertise the status of a color component.

Table 2.141: Color Switch Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_REPORT (0x04)							
Color Component ID							
Value							

Color Component ID (8 bits)

CC:0033.01.04.11.001 This field MUST advertise the color component covered by this report.

For the definition of Color Component IDs, refer to section Section 2.2.27.7.

Value (8 bits)

CC:0033.01.04.11.002 The Value field MUST advertise the value of the color component identified by the Color Component ID.

CC:0033.01.04.12.001 The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

CC:0033.01.04.11.003 A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command; not even when a transition has ended.

2.2.27.7 Color Switch Set Command

This command is used to control one or more color components in a device.

CC:0033.01.05.11.001 Color component levels MUST be scaled by the brightness level. Refer to Section 2.2.27.2

Table 2.142: Color Switch Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_SET (0x05)							
Color Component ID							
Value 1							
...							
Color Component ID N							
Value N							

Reserved

CC:0033.01.05.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component Count (5 bits)

CC:0033.01.05.11.003 This field MUST specify the number of (Color Component ID, Value) datasets contained in the Color Switch Set command.

Color Component ID (8 bits)

CC:0033.01.05.11.004 This field MUST specify the color component to receive a new value. This field MUST be encoded according to Table 2.143.

Table 2.143: Color Switch Component IDs

Component ID	Label	Value range
0	Warm White	0x00..0xFF
1	Cold White	0x00..0xFF
2	Red	0x00..0xFF
3	Green	0x00..0xFF
4	Blue	0x00..0xFF
5	Amber (for 6ch Color mixing)	0x00..0xFF
6	Cyan (for 6ch Color mixing)	0x00..0xFF
7	Purple (for 6ch Color mixing)	0x00..0xFF
8	Indexed Color [OBSOLETED] A supporting node MUST NOT support indexed colors	0x00 - 0xFF: Color Index 0-255

CC:0033.01.05.11.005 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Value (8 bits)

CC:0033.01.05.11.006 This field MUST specify the value of the color component identified by the Color Component ID field.

CC:0033.01.05.11.007 Unlike other actuator control command classes, the Value used by the Color Switch Command Class spans the entire range from 0x00 to 0xFF. Thus, except for the Indexed Color, the value 0x00 MUST yield to 0% and 0xFF MUST yield to 100% of the actual color component.

2.2.27.8 Color Switch Start Level Change Command

This command is used to initiate a transition of one color component to a new level.

CC:0033.01.06.11.001 A receiving device MUST initiate the transition to a new value for the specified Color Component ID.

CC:0033.01.06.13.001 The device MAY apply a non-zero duration to the transition from one value to a new value.

Table 2.144: Color Switch Start Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_START_LEVEL_CHANGE (0x06)							
Re-served	Up / Down	Ignore Start Level	Reserved				
Color Component ID							
Start Level							

Up/Down (1 bit)

CC:0033.01.06.11.002 This field MUST specify the direction of the level change.

CC:0033.01.06.11.003 If the Up/Down bit is set to 0 the level change MUST be increasing.

CC:0033.01.06.11.004 If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

CC:0033.01.06.12.001 A receiving device SHOULD respect the Start Level if the Ignore Start Level bit is 0.

CC:0033.01.06.11.005 A receiving device MUST ignore the Start Level if the Ignore Start Level bit is 1.

CC:0033.01.06.12.002 A controlling device SHOULD set the Ignore Start Level bit to 1.

Reserved

CC:0033.01.06.11.006 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component ID (8 bits)

CC:0033.01.06.11.007 This field MUST specify the color component to start a transition.

For the definition of Color Component IDs, refer to section Section 2.2.27.7.

Start Level (8 bits)

CC:0033.01.06.11.008 The Start Level field MUST specify the initial value of the level change.

2.2.27.9 Color Switch Stop Level Change Command

This command is used to stop an ongoing transition initiated by a Color Switch Start Level Change Command.

CC:0033.01.07.11.001 A receiving device MUST stop the transition if the specified Color Component ID is currently in transition to a new value.

CC:0033.01.07.11.002 A receiving device MUST NOT stop ongoing transitions for other Color Component IDs than the one specified.

Table 2.145: Color Switch Stop Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_GET (0x03)							
Color Component ID							

Color Component ID (8 bits)

CC:0033.01.07.11.003 This field MUST specify the color component to stop a transition.

For the definition of Color Component IDs, refer to section Section 2.2.27.7.

2.2.28 Color Switch Command Class, version 2

The Color Switch Command Class, version 2 is used to control color capable devices.

The Color Switch Command Class is an actuator control command class. Refer to Section 2.1.6

2.2.28.1 Compatibility considerations

CC:0033.02.00.21.001 A device supporting Color Switch Command Class, version 2 MUST support *Color Switch Command Class, version 1*.

Version 2 adds a duration parameter to the Color Switch Set Command.

Commands not described in this version remain unchanged from version 1.

2.2.28.1.1 Command Class dependencies

CC:0033.02.00.21.002 Nodes supporting the Color Switch Command Class, version 2 MUST support one of the following command classes in order to control the color brightness:

- Multilevel Switch Command Class, version 2
- Binary Switch Command Class, version 2.

2.2.28.2 Interoperability considerations

Refer to Section 2.2.27.2.

2.2.28.3 Color Switch Set Command

This command is used to control one or more color components in a device.

CC:0033.02.05.11.001 Color component levels MUST be scaled by the brightness level. Refer to Section 2.2.27.2.

Table 2.146: Color Switch Set Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_SET (0x05)							
Reserved			Color Component Count				
Color Component ID							
Value 1							
...							
Color Component ID N							
Value N							
Duration							

Reserved

CC:0033.02.05.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component Count (5 bits)

CC:0033.02.05.11.003 This field MUST specify the number of (Color Component ID, Value) datasets contained in the Color Switch Set command.

Color Component ID (8 bits)

CC:0033.02.05.11.004 This field MUST specify the color component to receive a new value.

For the definition of Color Component IDs, refer to section Section 2.2.27.7

Value (8 bits)

CC:0033.02.05.11.005 This field MUST specify the value of the color component identified by the Component ID field.

Refer to section Table 2.143.

Duration (8 bits)

CC:0033.02.05.11.006 The Duration field MUST specify the time that the transition should take from the current value to the new target value.

CC:0033.02.05.12.001 A supporting device SHOULD respect the specified Duration value.

CC:0033.02.05.11.007 The encoding of the Duration field MUST be according to Table 2.9.

CC:0033.02.05.12.002 The factory default duration SHOULD be the same as the duration used for the Color Switch Set command, version 1.

2.2.29 Color Switch Command Class, version 3

The Color Switch Command Class is used to control color capable devices.

The Color Switch Command Class manipulates the color components of a device. Each color component is scaled by the brightness level previously set by a Multilevel Switch Set or Basic Set command.

The Color Switch Command Class is an actuator control command class. Refer to Section 2.1.6.

2.2.29.1 Compatibility considerations

CC:0033.03.00.21.001 A node supporting Color Switch Command Class, version 3 MUST support the Color Switch Command Class, version 2.

Version 3 adds duration and target value reporting to the Color Switch Report Command and adds duration control to the Color Switch Start Level Change Command.

Commands not described in this version remain unchanged from version 2.

CC:0033.03.00.23.001 A node receiving a version 1 Color Switch Set command MAY apply a factory default duration to the transition.

2.2.29.1.1 Command Class dependencies

CC:0033.03.00.21.002 Nodes supporting the Color Switch Command Class, version 3 MUST support one of the following Command Classes in order to control the color brightness:

- Multilevel Switch Command Class, version 4
- Binary Switch Command Class, version 2.

2.2.29.2 Interoperability considerations

Refer to Section 2.2.28.2

2.2.29.3 Color Switch Report Command

This command is used to advertise the status of a color component.

Table 2.147: Color Switch Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_REPORT (0x04)							
Color Component ID							
Current Value							
Target Value							
Duration							

Color Component ID (8 bits)

CC:0033.03.04.11.001 This field MUST advertise the color component covered by this report.

Current Value (8 bits)

CC:0033.03.04.11.002 The Current Value field MUST advertise the current value of the color component identified by the Color Component ID.

Refer to the Color Switch Set command for valid values.

CC:0033.03.04.12.001 The Current Value SHOULD be identical to the Target Value when a transition has ended.

Target Value (8 bits)

CC:0033.03.04.11.003 The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition for the advertised Color Component ID.

Refer to the Color Switch Set command for valid values.

CC:0033.03.04.11.004 If a transition is initiated in an interactive fashion via a local user interface or via a Start Level Change command, the advertised Target Value MUST be 0x00 or 0xFF, depending on the direction.

CC:0033.03.04.12.002 The Current Value SHOULD be identical to the Target Value when a transition has ended.

Duration (8 bits)

CC:0033.03.04.12.003 CC:0033.03.04.11.005 The Duration field SHOULD advertise the time needed to reach the Target Value at the actual transition rate. The encoding of the Duration field MUST be according to Table 2.10

2.2.29.4 Color Switch Start Level Change Command

This command is used to initiate a transition of one color component to a new value.

Table 2.148: Color Switch Start Level Change Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR (0x33)							
Command = SWITCH_COLOR_START_LEVEL_CHANGE (0x06)							
Re-served	Up / Down	Ignore Start Level	Reserved				
Color Component ID							
Start Level							
Duration							

Up/Down (1 bit)

CC:0033.03.06.11.001 This field MUST specify the direction of the level change.

CC:0033.03.06.11.002 If the Up/Down bit is set to 0 the level change MUST be increasing.

CC:0033.03.06.11.003 If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

CC:0033.03.06.12.001 A receiving device SHOULD respect the Start Level if the Ignore Start Level bit is 0.

CC:0033.03.06.11.004 A receiving device MUST ignore the Start Level if the Ignore Start Level bit is 1.

CC:0033.03.06.12.002 A controlling device SHOULD set the Ignore Start Level bit to 1.

Reserved

CC:0033.03.06.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component ID (8 bits)

Refer to Color Switch Set command.

Start Level (8 bits)

CC:0033.03.06.11.006 The Start Level field MUST specify the initial value of the level change.

Duration (8 bits)

CC:0033.03.06.11.007 The level change rate MUST be calculated to match a level change from 0x00 to 0xFF during the time specified by the Duration field.

CC:0033.03.06.12.003 A supporting device SHOULD respect the specified Duration value.

For encoding of the Duration value, refer to the Color Switch Set Command.

2.2.30 Configuration Command Class, version 1

The Configuration Command Class allows product specific configuration parameters to be changed. One example could be the default dimming rate of a light dimmer.

CC:0070.01.00.11.001 Configuration parameters MUST be specified in the product documentation. Configuration parameters accessed via this command class MUST NOT replace similar commands provided by other existing Command Classes.

CC:0070.01.00.11.002

CC:0070.01.00.12.001 A device MUST be able to operate with default factory configuration parameter values.

CC:0070.01.00.11.003 It is RECOMMENDED that configuration parameters can be manipulated via a local user interface.

CC:0070.01.00.12.002 It is RECOMMENDED that default factory configuration parameter values can be restored via a local user interface.

2.2.30.1 Compatibility considerations

2.2.30.1.1 "Default" flag

Earlier text revisions of the Configuration Command Class, versions 1-3 presented conflicting interpretations of the default field. Controllers should be aware that nodes with version 3 or less MAY reset all configuration parameters to default when receiving a Configuration Set or Configuration Bulk Set Command with the Default bit set to 1.

CC:0070.01.00.23.001

CC:0070.01.00.22.001 A controller SHOULD probe a node in order to discover what behavior it implements. The following steps are RECOMMENDED:

- If the device is version 1- 2, scan the node for available configuration parameters. Else discover the available parameters thanks to the Configuration Properties Get Command.
- Determine how the default bit works if the node implements 2 or more parameters

Step 1:

- One at a time, issue a Configuration Set for a parameter number, and check whether the value can be read back. Try each parameter for 1, 2 and 4 bytes sizes.
- Store the discovered parameter list for future use.

Step 2:

- Choose 2 parameters and issue a Configuration Set with the default bit set to 1 for each parameter.
- Read back both parameters value with a Configuration Get.
- Try to modify both parameters to non-default values (it is recommended to try the default value $\pm$ 1)
- Send a Configuration Set for one of parameters with the default bit set to 1.
- Read back both parameters and check whether one or both returned to the default value

2.2.30.2 Configuration Set Command

The Configuration Set Command is used to set the value of a configuration parameter.

Table 2.149: Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_SET (0x04)							
Parameter Number							
Default	Reserved				Size		
Configuration Value 1							
...							
Configuration Value N							

Parameter Number (8 bits)

CC:0070.01.04.13.001 This field is used to specify the actual configuration parameter. Parameter Numbers MAY be product
CC:0070.01.04.12.001 specific. Parameter Numbers SHOULD be assigned in a sequence starting from 1.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameter.

CC:0070.01.04.11.001 The value 1 MUST indicate that default factory settings must be restored for the specified Parameter
CC:0070.01.04.11.002 Number. In this case, the Configuration Value field MUST be ignored and the receiving node MUST reset the specified parameter and SHOULD NOT reset any other parameters.

CC:0070.01.04.11.003 The value 0 MUST indicate that the specified Parameter Number must assume the value specified by the Configuration Value field.

Reserved

CC:0070.01.04.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

CC:0070.01.04.11.005 This field is used to specify the size of the actual parameter. The advertised size of a given parameter MUST always be the same. The value of this field MUST comply with Table 2.150.

Table 2.150: Configuration Set::Size encoding

Size	Size of Value Field
1	8 bit
2	16 bit
4	32 bit

CC:0070.01.04.11.006 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Configuration Value (N bytes)

CC:0070.01.04.11.007 This field carries the value to be assigned. The size of the field MUST comply with the size advertised by the Size field.

CC:0070.01.04.11.008 The field MUST carry a signed value. The binary encoding of the signed value MUST comply with Table 2.12. The field Value 1 MUST be the most significant byte.

2.2.30.3 Configuration Get Command

This command is used to query the value of a configuration parameter.

CC:0070.01.05.11.001 The Configuration Report Command MUST be returned in response to this command.

CC:0070.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:0070.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.151: Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_GET (0x05)							
Parameter Number							

Parameter Number (8 bits)

CC:0070.01.05.13.001 CC:0070.01.05.12.001 This field is used to specify the requested configuration parameter. Parameter Numbers MAY be product specific. Parameter Numbers SHOULD be assigned in a sequence starting from 1.

CC:0070.01.05.12.002 A node receiving this command for an unsupported parameter SHOULD return a Report advertising the value of the first available parameter in the node.

2.2.30.4 Configuration Report Command

This command is used to advertise the actual value of the advertised parameter.

Table 2.152: Configuration Report“ Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_REPORT (0x06)							
Parameter Number							
Reserved					Size		
Configuration Value 1							
...							
Configuration Value N							

Refer to the Configuration Set Command for parameter details.

2.2.31 Configuration Command Class, version 2

The Configuration Command Class allows product specific configuration parameters to be changed. One example could be the default dimming rate of a light dimmer.

CC:0070.02.00.11.001 Configuration parameters MUST be specified in the product documentation. Configuration parameters accessed via this command class MUST NOT replace similar commands provided by other existing Command Classes.

CC:0070.02.00.11.002

CC:0070.02.00.11.003 A device MUST be able to operate with default factory configuration parameter values.

CC:0070.02.00.12.001 It is RECOMMENDED that configuration parameters can be manipulated via a local user interface.

CC:0070.02.00.12.002 It is RECOMMENDED that default factory configuration parameter values can be restored via a local user interface.

2.2.31.1 Compatibility considerations

Configuration Command Class, version 2 extends the addressing space to 65.535 product specific configuration parameters. Further, Configuration Command Class, version 2 makes it possible to set multiple configuration parameters with one Command.

CC:0070.02.00.21.001 A device supporting Configuration Command Class, version 2 MUST support *Configuration Command Class, version 1*.

2.2.31.1.1 "Default" flag

Refer to Section 2.2.30.1.1.

2.2.31.2 Interoperability considerations

Configuration Command Class, version 2 is backwards compatible with Configuration Command Class, version 1.

Configuration Command Class, version 2 does not extend any existing commands of Configuration Command Class, version 1. The 255 parameters that can be addressed by the Configuration Set Command MUST be identical to the first 255 parameters that can be addressed by the (version 2) Configuration Bulk Set Command.

CC:0070.02.00.31.001

2.2.31.3 Configuration Set Command

This Command is used to set the value of a configuration parameter.

Table 2.153: Configuration Set Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_SET (0x04)							
Parameter Number							
Default	Reserved				Size		
Configuration Value 1							
...							
Configuration Value N							

Parameter Number (8 bits)

CC:0070.02.04.13.001 This field is used to specify the actual configuration parameter. Parameter Numbers MAY be product specific. Parameter Numbers SHOULD be assigned in a sequence starting from 1.

CC:0070.02.04.12.001

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameter.

CC:0070.02.04.11.001 The value 1 MUST indicate that default factory settings must be restored for the specified Parameter
CC:0070.02.04.11.002 Number. In this case, the Configuration Value field MUST be ignored and the receiving node MUST reset the specified parameter and SHOULD NOT reset any other parameters.

CC:0070.02.04.11.003 The value 0 MUST indicate that the specified Parameter Number must assume the value specified by the Configuration Value field.

Reserved

CC:0070.02.04.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

CC:0070.02.04.11.005 This field is used to specify the size of the actual parameter. The advertised size of a given parameter MUST always be the same. The value of this field MUST comply with Table 2.150.

CC:0070.02.04.11.006 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Configuration Value (N bytes)

CC:0070.02.04.11.007 This field carries the value to be assigned. The size of the field MUST comply with the size advertised by the Size field.

CC:0070.02.04.11.008 The field MUST carry a signed value. The binary encoding of the signed value MUST comply withTable 10. The field Value 1 MUST be the most significant byte.

2.2.31.4 Configuration Bulk Set Command

This command is used to set the value of one or more configuration parameters.

Table 2.154: Configuration Bulk Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_BULK_SET (0x07)							
Parameter Offset 1 (MSB)							
Parameter Offset 2 (LSB)							
Number of Parameters							
Default	Hand- shake	Reserved			Size		
Parameter 1 - Configuration Value 1 (MSB)							
...							
Parameter 1 - Configuration Value N (LSB)							
...							
Parameter M - Configuration Value 1 (MSB)							
...							
Parameter M - Configuration Value N (LSB)							

Parameter Offset (16 bits)

CC:0070.02.07.11.001 This field is used to specify the first parameter in a range of one or more parameters. The first byte MUST carry the most significant byte of the 16 bit value.

Number of Parameters (8 bits)

This field is used to specify the number (M) of configuration parameters contained in this command.

CC:0070.02.07.11.002 The value of this field MUST be in the range 1..255.

CC:0070.02.07.11.010

The advertised parameters MUST be in a consecutive range. For example, Parameter Offset set to 4 and Number of Parameters set to 3 MUST represent parameters number 4, 5 and 6.

CC:0070.02.07.11.011 A receiving node MUST ignore values for non-existing parameters

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameters.

CC:0070.02.07.11.003 The value 1 MUST indicate that default factory settings must be restored for the Parameter Numbers defined by the Parameter Offset and Number of Parameters fields. In this case, the Configuration Value
CC:0070.02.07.11.004 field MUST be ignored and the receiving node MUST reset the specified parameters and SHOULD NOT reset any other parameters.

CC:0070.02.07.11.005 The value 0 MUST indicate that the specified Parameter Numbers must assume the value specified by the Configuration Value field.

Handshake (1 bit)

This field is used to indicate if a Configuration Bulk Report Command is to be returned when the specified configuration parameters have been stored in non-volatile memory.

CC:0070.02.07.11.006 If the Handshake bit is set to 1, a Configuration Bulk Report Command MUST be returned. The
CC:0070.02.07.11.007 command MUST echo all parameters found in this Configuration Bulk Set Command. A node return-
CC:0070.02.07.11.008 ing a Configuration Bulk Report Command with the Handshake bit set MUST be ready to receive
CC:0070.02.07.11.009 another Configuration Bulk Set Command. If the Handshake bit is set, an originating node MUST wait for the Configuration Bulk Report Command for at least one second. If not receiving a report,
CC:0070.02.07.12.001 the originating node SHOULD assume that the operation failed and resend the same Configuration Bulk Set Command one more time.

CC:0070.02.07.11.00A If the Handshake bit is set to 0, a receiving node MUST NOT return a Configuration Bulk Report in response.

Reserved

CC:0070.02.07.11.00B This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

This field is used to specify the size of the actual parameters. The advertised size of a given parameter
CC:0070.02.07.11.00C MUST always be the same. The value of this field MUST comply with Table 2.150.

CC:0070.02.07.11.00D All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Parameter - Configuration Value (M*N bytes)

CC:0070.02.07.11.00E These fields carry the values to be assigned. Each field MUST have the same size. The size of each field MUST comply with the size advertised by the Size field.

CC:0070.02.07.11.00F The field MUST carry a signed value. The binary encoding of the signed value MUST comply withTable 10. The Value 1 field MUST be the most significant byte.

2.2.31.5 Configuration Bulk Get Command

This command is used to query the value of one or more configuration parameters.

CC:0070.02.08.11.001 The Configuration Bulk Report Command MUST be returned in response to this command.

CC:0070.02.08.11.002 This command MUST NOT be issued via multicast addressing.

CC:0070.02.08.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.155: Configuration Bulk Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_BULK_GET (0x08)							
Parameter Offset 1 (MSB)							
Parameter Offset 2 (LSB)							
Number of Parameters							

Parameter Offset (16 bits)

This field is used to specify the first parameter in a range of one or more parameters. The first byte
CC:0070.02.08.11.004 MUST carry the most significant byte of the 16 bit value.

CC:0070.02.08.12.001 Parameter Numbers SHOULD be assigned in a sequence starting from 1.

Number of Parameters (8 bits)

This field is used to specify the number of requested configuration parameters.

CC:0070.02.08.11.005 The value of this field MUST be in the range 1..255.

CC:0070.02.08.11.006 The advertised parameters MUST be in a consecutive range. For example, Parameter Offset set to 4 and Number of Parameters set to 3 MUST represent parameters number 4, 5 and 6.

2.2.31.6 Configuration Bulk Report Command

This command is used to advertise the actual value of one or more advertised parameters.

Table 2.156: Configuration Bulk Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_BULK_REPORT (0x09)							
Parameter Offset 1 (MSB)							
Parameter Offset 2 (LSB)							
Number of Parameters							
Reports to follow							
Default	Hand- shake	Reserved			Size		
Parameter 1 - Configuration Value 1 (MSB)							
...							
Parameter 1 - Configuration Value N (LSB)							
...							
Parameter M - Configuration Value 1 (MSB)							
...							
Parameter M - Configuration Value N (LSB)							

Parameter Offset (16 bits)

CC:0070.02.09.11.001 This field is used to advertise the first parameter in a range of one or more parameters. The first byte MUST carry the most significant byte of the 16 bit value.

Number of Parameters (8 bits)

This field is used to advertise the number (M) of configuration parameters contained in this command.

CC:0070.02.09.11.002 The value of this field MUST be in the range 1..255.

CC:0070.02.09.11.012 The advertised parameters MUST be in a consecutive range. For example, Parameter Offset set to 4 and Number of Parameters set to 3 MUST represent parameters number 4, 5 and 6, regardless of whether they exist at the sending node.

Reports to follow (8 bits)

CC:0070.02.09.11.003

This field MUST be used to advertise the number of reports left before all requested configuration parameters values have been transferred.

CC:0070.02.09.11.004 The value 0 MUST indicate that this is the last report.

CC:0070.02.09.11.005 If the Handshake field is 1 , this field MUST be set to 0.

CC:0070.02.09.11.006 The value of this field MUST be in the range 0..255.

Default (1 bit)

CC:0070.02.09.11.007 This field MUST be used to advertise if all advertised configuration parameters have the factory default value.

CC:0070.02.09.11.008 The value 1 MUST indicate that all advertised configuration parameters have the factory default value.

CC:0070.02.09.11.009 The value 0 MUST indicate that one or more of the advertised configuration parameters do not have the factory default value.

Handshake (1 bit)

This field is used to indicate if this report is returned in response to a Configuration Bulk Set Command.

CC:0070.02.09.11.00A The value 1 MUST indicate that all configuration parameters have been stored in non-volatile memory
CC:0070.02.09.11.00B and that the sending node is ready to receive another Configuration Bulk Set Command. Except for
CC:0070.02.09.11.00C the Reports to Follow field, all other fields MUST echo the values received in the Configuration Bulk Set Command. The Reports to Follow field MUST be 0.

CC:0070.02.09.11.00D The value 0 MUST indicate that this report is returned in response to a Configuration Bulk Get Command.

Size (3 bits)

This field is used to advertise the size of the actual parameter. The advertised size of a given parameter
CC:0070.02.09.11.00E MUST always be the same. The value of this field MUST comply with Table 2.150.

CC:0070.02.09.11.00F All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Parameter - Configuration Value (M*N bytes)

These fields carry the values advertised by the Parameter Offset and Number of Parameters fields.
CC:0070.02.09.11.010 Each field MUST have the same size. The size of each field MUST comply with the size advertised by the Size field.

CC:0070.02.09.11.011 The field MUST carry a signed value. The binary encoding of the signed value MUST comply with Table 2.12 Table 10. The Value 1 field MUST be the most significant byte.

CC:0070.02.09.12.001 A sending node SHOULD set this field to 0 for non-existing parameters.

2.2.32 Configuration Command Class, version 3

The Configuration Command Class, version 3 allows product specific configuration parameters to be changed. Examples include the default dimming rate of a light dimmer and the endpoints of a window covering device.

CC:0070.03.00.11.001 Configuration parameters accessed via this command class MUST NOT replace similar commands provided by other existing application Command Classes.

CC:0070.03.00.11.002 Configuration parameter documentation MUST be provided in the product documentation.

CC:0070.03.00.12.001 Configuration parameter documentation SHOULD be provided via Z-Wave via this command class version.

CC:0070.03.00.11.003 A device MUST be able to operate with default factory configuration parameter values.

CC:0070.03.00.12.002 It is RECOMMENDED that configuration parameters can be manipulated via a local user interface.

CC:0070.03.00.12.003 It is RECOMMENDED that default factory configuration parameter values can be restored via a local user interface.

2.2.32.1 Compatibility considerations

The following additions are made to the Configuration Command Class, version 3:

- Configuration Name Get Command
- Configuration Name Report Command
- Configuration Info Get Command
- Configuration Info Report Command
- Configuration Properties Get Command
- Configuration Properties Report Command

The purpose of the above commands is to allow a manufacturer to advertise information relating to the use of a specific configuration parameter directly from the configuration interface. The advertised information should be the same as can be found in the printed documentation.

It should be emphasized that, just like versions 1 and 2, the Configuration Command Class, version 3 is intended only for access to product specific configuration parameters. The access to parameter information is only meant to facilitate access via Z-Wave to information which can also be found in the printed documentation.

Configuration Command Class versions 1 and 2 explicitly require that the binary value of all configuration parameters are represented as signed values encoded as binary 2's complement.

Configuration Command Class v3 allows for bit field, signed integer or unsigned integer representations of each individual parameter.

CC:0070.03.00.21.001 A device supporting the Configuration Command Class v3 MUST respond to a Configuration Properties Get command. If a Configuration Properties Report command is not returned in response to
CC:0070.03.00.21.002 a Configuration Properties Get command, a controlling device MUST treat the parameter value as a signed integer.

Configuration Command Class version 2 extended the possible parameter range from 1..255 to 1..65535. The first 255 parameters of the two ranges are identical. Parameters in the range 1..255 can be addressed using either the Bulk Set/Get Commands or Set/Get Commands while parameters in the range 256..65535 can only be addressed using the Bulk Set/Get Commands.

2.2.32.1.1 "Default" flag

Refer to Section 2.2.30.1.1.

2.2.32.1.2 Configuration Properties Report

Earlier text revisions of the Configuration Command Class, version 3 did not explicitly specify that
CC:0070.03.00.21.003 the "Min Value", "Max Value" and "Default Value" fields MUST be omitted if the advertised size is 0 in the Configuration Properties Report Command.

CC:0070.03.00.22.001 Controlling nodes SHOULD be aware that some legacy nodes supporting version 3 could by error include the "Min Value", "Max Value" and "Default Value" fields and set them to 0x00 with an
CC:0070.03.00.22.002 arbitrary size. If a controlling node receives a Report in which the "Next Parameter Number" field seems to be set to 0x0000 when requesting parameter number 0, the controlling node SHOULD inspect the last 2 bytes of the command frame in order to find out what is the Next Parameter Number.

2.2.32.2 Interoperability considerations

CC:0070.03.00.32.001 It is RECOMMENDED that the name and information stored in a supporting device are in English.

CC:0070.03.00.32.002 It is RECOMMENDED that the last line of the information field presents an Internet URL which points to updated information for the actual configuration parameter as well as other language variants.

2.2.32.3 Configuration Name Get Command

This command is used to request the name of a configuration parameter.

CC:0070.03.0A.11.001 The Configuration Name Report Command MUST be returned in response to this command.

CC:0070.03.0A.11.002 This command MUST NOT be issued via multicast addressing.

CC:0070.03.0A.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.157: Configuration Name Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_NAME_GET (0x0A)							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							

Parameter Number (16 bits)

This field is used to specify the requested configuration parameter.

CC:0070.03.0A.11.004 The first byte MUST carry the most significant byte of the 16 bit value.

2.2.32.4 Configuration Name Report Command

This command is used to advertise the name of a parameter.

Table 2.158: Configuration Name Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_NAME_REPORT (0x0B)							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							
Reports to follow							
Name 1							
...							
Name N							

Parameter Number (16 bits)

This field is used to advertise the actual configuration parameter.

CC:0070.03.0B.11.001 The first byte MUST carry the most significant byte of the 16 bit value.

CC:0070.03.0B.13.001 If a non-existing parameter is specified in a Configuration Name Get Command, a receiving node MAY return a zero-length Name field in the Configuration Name Report Command. The receiving
CC:0070.03.0B.12.001 node SHOULD however return an error message e.g. “Unassigned parameter”.

Parameter numbers above 255 can only be addressed using the Bulk Set/Get Commands. Parameter numbers in the range 1..255 can be addressed using either the Bulk Set/Get Commands or the Set/Get Commands.

Reports to follow (8 bits)

This field is used to advertise the number of reports left before all parts of the command have been transferred.

CC:0070.03.0B.11.002 The value 0 MUST indicate that this is the last report.

Name (N bytes)

CC:0070.03.0B.12.002 This field is used to advertise the name of the parameter. It is RECOMMENDED that the Name field advertises the name found in the documentation.

CC:0070.03.0B.11.003 The field MUST carry a byte string with no zero termination. The Name field MAY span multiple reports. The number of Name bytes transmitted MUST be determined from the length field in the frame.

CC:0070.03.0B.11.004 Bytes MUST be encoded in UTF-8 format.

CC:0070.03.0B.12.003 It is RECOMMENDED that the name stored in a supporting device is in English.

2.2.32.5 Configuration Info Get Command

This command is used to request usage information for a configuration parameter.

CC:0070.03.0C.11.001 The Configuration Info Report Command MUST be returned in response to this command.

CC:0070.03.0C.11.002 This command MUST NOT be issued via multicast addressing.

CC:0070.03.0C.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.159: Configuration Info Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_INFO_GET (0x0C)							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							

Parameter Number (16 bits)

This field is used to specify the requested configuration parameter.

CC:0070.03.0C.11.004 The first byte MUST carry the most significant byte of the 16 bit value.

2.2.32.6 Configuration Info Report Command

This command is used to advertise usage information for a configuration parameter.

Table 2.160: Configuration Info Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_INFO_REPORT (0x0D)							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							
Reports to follow							
Info 1							
...							
Info N							

Parameter Number (16 bits)

This field is used to advertise the actual configuration parameter.

CC:0070.03.0D.11.001 The first byte MUST carry the most significant byte of the 16 bit value.

CC:0070.03.0D.13.001 CC:0070.03.0D.12.001 If a non-existing parameter is specified in a Configuration Info Get Command, a receiving node MAY return a zero-length Info field in the Configuration Info Report Command. The receiving node SHOULD however return an error message, e.g. “Unassigned parameter”.

Parameter numbers above 255 can only be addressed using the Bulk Set/Get Commands. Parameter numbers in the range 1..255 can be addressed using either the Bulk Set/Get Commands or the Set/Get Commands

Reports to follow (8 bits)

This field is used to advertise the number of reports left before all parts of the command have been transferred.

CC:0070.03.0D.11.002 The value 0 MUST indicate that this is the last report.

Info (N bytes)

CC:0070.03.0D.12.002 This field is used to advertise the detailed information available for the parameter. It is RECOMMENDED that the Info field carries all information found in the documentation.

CC:0070.03.0D.11.003 The field MUST carry a byte string with no zero termination.

CC:0070.03.0D.11.004 The number of Info bytes transmitted MUST be determined from the length field in the frame.

CC:0070.03.0D.13.002 The Info field MAY span multiple reports. The number of Info bytes MAY be zero.

CC:0070.03.0D.11.005 Bytes MUST be encoded in UTF-8 format.

CC:0070.03.0D.12.003 It is RECOMMENDED that the information stored in a supporting device is in English.

CC:0070.03.0D.12.004

It is RECOMMENDED that the last line of the information field presents an Internet URL which points to updated information for the actual configuration parameter as well as other language variants.

2.2.32.7 Configuration Properties Get Command

This command is used to request the properties of a configuration parameter.

The Configuration Properties Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.161: Configuration Properties Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_PROPERTIES_GET (0x0E)							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							

Parameter Number (16 bits)

This field is used to specify the requested configuration parameter.

CC:0070.03.0E.11.001 The first byte MUST carry the most significant byte of the 16 bit value.

CC:0070.03.0E.11.002 CC:0070.03.0E.11.003 If a non-existing parameter is specified in this command, a receiving node MUST advertise zero values in the Format and Size fields. The Next Parameter Number field MUST advertise the next available configuration parameter.

CC:0070.03.0E.12.001 It is RECOMMENDED that a controlling device initiates probing of supported configuration parameters by issuing this command for parameter number 0.

CC:0070.03.0E.12.002 If a Size field value of zero is returned, a controlling device SHOULD issue a Configuration Properties Get Command for the parameter advertised in the Next Parameter Number field.

2.2.32.8 Configuration Properties Report Command

This command is used to advertise the properties of a configuration parameter.

Table 2.162: Configuration Properties Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_PROPERTIES_REPORT (0x0F)							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							
Reserved			Format			Size	
Min Value 1 (MSB)							
...							
Min Value N (LSB)							
Max Value 1 (MSB)							
...							
Max Value N (LSB)							
Default Value 1 (MSB)							
...							
Default Value N (LSB)							
Next Parameter Number 1 (MSB)							
Next Parameter Number 2 (LSB)							

Parameter Number (16 bits)

This field is used to advertise the actual configuration parameter.

CC:0070.03.0F.11.001 The first byte MUST carry the most significant byte of the 16 bit value.

CC:0070.03.0F.11.002 CC:0070.03.0F.11.003 If this field advertises a non-existing parameter , a receiving node MUST advertise zero values in the Format and Size fields. The Next Parameter Number field MUST advertise the next available configuration parameter.

Parameter numbers above 255 can only be addressed using the Bulk Set/Get Commands. Parameter numbers in the range 1..255 can be addressed using either the Bulk Set/Get Commands or the Set/Get Commands.

Reserved

CC:0070.03.0F.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Format (3 bits)

This field is used to advertise the format of the parameter.

CC:0070.03.0F.11.005 The value of this field MUST comply with Table 2.163.

Table 2.163: Configuration Properties Report::Format encoding

Value	Parameter format and presentation
0x00	Signed Integer
0x01	Unsigned Integer
0x02	Enumerated (Radio buttons)
0x03	Bit field (Checkboxes)

CC:0070.03.0F.11.006 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0070.03.0F.11.007 If the parameter format is “Unsigned integer”, normal binary integer encoding MUST be used.

CC:0070.03.0F.11.008 If the parameter format is “Signed integer”, the binary encoding MUST comply with Table 2.12.

CC:0070.03.0F.11.009 If the parameter format is “Enumerated”, the parameter MUST be treated as an unsigned integer.

CC:0070.03.0F.12.001

A graphical configuration tool SHOULD present this parameter as a series of radio buttons [21].

CC:0070.03.0F.11.00A If the parameter format is "Bit field" the parameter MUST be treated as a bit field where each individual bit can be set or reset.

CC:0070.03.0F.12.002 A graphical configuration tool SHOULD present this parameter as a series of checkboxes [21].

Size (3 bits)

This field is used to advertise the size of the actual parameter.

CC:0070.03.0F.11.00B The advertised size MUST also apply to the fields "Min Value", "Max Value", "Default Value" carried in this command.

CC:0070.03.0F.11.00C The value of this field MUST comply with Table 2.164.

Table 2.164: Configuration Properties Report::Size encoding

Size	Size of parameter
0	0 bit (Unassigned parameter, Value fields omitted)
1	8 bit
2	16 bit
4	32 bit

CC:0070.03.0F.11.00D All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Min Value (N bytes)

CC:0070.03.0F.11.00E This field MUST advertise the minimum value that the actual parameter can assume.

CC:0070.03.0F.11.00F If the parameter is "Bit field", this field MUST be set to 0.

CC:0070.03.0F.11.010 The size of the field MUST comply with the size advertised by the Size field. This field MUST be omitted if the Size field is set to 0.

CC:0070.03.0F.11.011 The sign encoding MUST comply with the mode advertised by the Signed field.

Max Value (N bytes)

CC:0070.03.0F.11.012 This field MUST advertise the maximum value that the actual parameter can assume.

CC:0070.03.0F.11.013 If the parameter is "Bit field", each individual supported bit MUST be set to '1', while each un-supported bit of MUST be set to '0'.

CC:0070.03.0F.12.003 A graphical configuration tool SHOULD NOT present checkboxes for un-supported bits.

CC:0070.03.0F.11.014 The size of the field MUST comply with the size advertised by the Size field. This field MUST be omitted if the Size field is set to 0.

CC:0070.03.0F.11.015 The sign encoding MUST comply with the mode advertised by the Signed field.

Default Value (N bytes)

CC:0070.03.0F.11.016 This field MUST advertise the default value of the actual parameter.

CC:0070.03.0F.11.017 The size of the field MUST comply with the size advertised by the Size field. This field MUST be omitted if the Size field is set to 0.

CC:0070.03.0F.11.018 The sign encoding MUST comply with the mode advertised by the Signed field.

Next Parameter Number (16 bits)

Configuration parameter identifiers may be assigned in a non-sequential order.

CC:0070.03.0F.11.019 This field MUST advertise the next available configuration parameter.

The value 0x0000 MUST indicate that this is the last available configuration parameter.

CC:0070.03.0F.11.01A The first byte MUST carry the most significant byte of the 16 bit value.

2.2.33 Configuration Command Class, version 4

The Configuration Command Class, version 4 introduces the support of read-only parameters, parameters requiring network re-inclusion and advanced parameters as well as allows optional support for Bulk Commands.

A read-only parameter can be accessed with Get commands but cannot be modified with Set Commands. A parameter requiring re-inclusion is typically a parameter altering the node capabilities, which requires performing the inclusion again in order to have the controller to learn about the node capabilities once more. A parameter may be advertised as "advanced" to indicate to the controlling application that the actual parameter is intended only for advanced use, e.g. calibration.

The following commands are modified in version 4:

- Configuration Set Command
- Configuration Bulk Set Command
- Configuration Properties Report Command

The following new command is introduced:

- Configuration Default Reset Command

All commands not mentioned in this version remain unchanged from previous versions.

2.2.33.1 Compatibility considerations

Configuration Command Class, version 4 is backwards compatible with the previous versions of Configuration Command Class.

CC:0070.04.00.21.001 A device supporting Configuration Command Class, version 4 MUST support Configuration Command Class, version 3.

2.2.33.1.1 Multi Channel Consideration

CC:0070.04.00.12.001 Multi Channel End Points SHOULD NOT support the Configuration Command Class.

2.2.33.1.2 "Default" flag

Refer to Section 2.2.30.1.1.

CC:0070.04.00.21.002 From version 4 and onwards, a node MUST NOT reset all Configuration Parameters when receiving a Set or Bulk Set Command with the default flag set to 1.

2.2.33.1.3 Configuration Properties Report

Refer to Section 2.2.32.1.2.

The recommendation given in version 3 does not apply to version 4 supporting nodes as new fields are appended at the end of the Configuration Properties Report Command in version 4.

2.2.33.1.4 "Altering capabilities" flag

Configuration parameters modifying a node's and/or (non-dynamic) Multi Channel End Point capa-
CC:0070.04.00.21.003 bilities MUST be advertised as "Altering capabilities".

CC:0070.04.00.21.004 When such a parameter is modified:

- A Z-Wave or Z-Wave Plus node MUST NOT advertise the new and/or Multi Channel End Point capabilities before being excluded from its current network.
- A Z-Wave Plus version 2 node MUST advertise its new capabilities immediately.

2.2.33.1.5 "Advanced" flag

The difference between normal and advanced parameters lies on the controlling node side.

CC:0070.04.00.23.001 A parameter MAY be advertised as "advanced" in order to simplify the configuration of a node by normally not showing such a parameter to the end user.

CC:0070.04.00.22.001 Advanced parameters SHOULD be presented to the user only when an [Advanced] option is selected in the controller user interface.

2.2.33.1.6 Parameters value and network inclusion/exclusion

CC:0070.04.00.21.005 From version 4 and onwards, a node MUST NOT modify or reset any configuration parameter when being included or excluded of a Z-Wave network.

CC:0070.04.00.21.006 A node MUST reset all its configuration parameters if either:

- It is manually reset to factory default
- It receives a Configuration Default Reset Command.

CC:0070.04.00.21.007 A node MUST NOT reset all its configuration parameters in any other case.

2.2.33.1.7 Bulk commands support

CC:0070.04.00.22.002 A node supporting Configuration Command Class, version 4 MAY elect to ignore the following Commands:

- Configuration Bulk Set Command
- Configuration Bulk Get Command
- Configuration Bulk Report Command

CC:0070.04.00.21.008 If it is the case, the node MUST:

- Return an Application Rejected Request Command when receiving one of the ignored commands (if received without Supervision encapsulation)
- Advertise that it ignores the Bulk Commands in the Configuration Properties Report

CC:0070.04.00.21.009 If Bulk Commands are supported, they MUST be supported for all parameters. If Bulk Commands are ignored, they MUST be ignored for all parameters.

2.2.33.2 Configuration Set Command

This command is used to set the value of a configuration parameter.

Table 2.165: Configuration Set Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_SET (0x04)							
Parameter Number							
Default	Reserved				Size		
Configuration Value 1							
...							
Configuration Value N							

CC:0070.04.04.11.001 A receiving node MUST ignore this command if the specified parameter is advertised as a Read-only parameter.

All fields not specified below are unchanged from version 2 Configuration Set Command.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameter.

CC:0070.04.04.11.002 The value 1 MUST indicate that default factory settings must be restored for the specified Parameter
CC:0070.04.04.11.003 Number. In this case, the Configuration value field MUST be ignored and any other parameters MUST NOT be reset by a receiving node.

CC:0070.04.04.11.004 The value 0 MUST indicate that the specified Parameter Number MUST assume the value specified by the Configuration Value field.

Configuration Value (N bytes)

CC:0070.04.04.11.005 This field carries the value to be assigned. The size of the field MUST comply with the size advertised by the Size field.

CC:0070.04.04.11.006 The configuration value MUST be encoded according to the Format field advertised in the Configuration Properties Report Command for the parameter number.

2.2.33.3 Configuration Bulk Set Command

This command is used to set the value of one or more configuration parameters.

Table 2.166: Configuration Bulk Set Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_BULK_SET (0x07)							
Parameter Offset 1 (MSB)							
Parameter Offset 2 (LSB)							
Number of Parameters							
Default	Hand- shake	Reserved			Size		
Parameter 1 - Configuration Value 1 (MSB)							
...							
Parameter 1 - Configuration Value N (LSB)							
...							
Parameter M - Configuration Value 1 (MSB)							
...							
Parameter M - Configuration Value N (LSB)							

CC:0070.04.07.11.001

A receiving node MUST ignore this command for parameters that are advertised as Read-only parameters.

All fields not specified below are unchanged from version 2 Configuration Bulk Set Command.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameters.

CC:0070.04.07.11.002 The value 1 MUST indicate that default factory settings must be restored for all the Parameter Numbers defined by the Parameter Offset and Number of Parameters fields. In this case, the Configuration
CC:0070.04.07.11.003 Value field MUST be ignored and any other parameters MUST NOT be reset by a receiving node.

CC:0070.04.07.11.004 The value 0 MUST indicate that the specified Parameter Numbers MUST assume the value specified by the Configuration Value field.

Parameter -Configuration Value (M*N bytes)

CC:0070.04.07.11.005 These fields carry the parameter values to be assigned. Each parameter value field MUST have the same size. The size of each field MUST comply with the size advertised by the Size field.

CC:0070.04.07.11.006 The values MUST be encoded according to the Format field advertised in the Configuration Properties Report Command for each parameter number.

2.2.33.4 Configuration Properties Report Command

This command is used to advertise the properties of a configuration parameter.

Table 2.167: Configuration Properties Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_PROPERTIES_REPORT (0x0F)							
Parameter Number 1 (MSB)							
Parameter Number 2 (MSB)							
Altering capabilities	Read-only	Format			Size		
Min Value 1 (MSB)							
...							
Min Value N (LSB)							
Max Value 1 (MSB)							
...							
Max Value N (LSB)							
Default Value 1 (MSB)							
...							
Default Value N (LSB)							
Next Parameter Number 1 (MSB)							
Next Parameter Number 2 (LSB)							
Reserved						No Bulk support	Advanced

All fields not specified below are unchanged from version 3 Configuration Properties Report Command.

Read-only (1 bit)

This field is used to indicate if the parameter is read-only.

CC:0070.04.0F.11.001 The value 1 MUST indicate that the advertised parameter is read-only.

CC:0070.04.0F.11.002 The value 0 MUST indicate that the advertised parameter is not read-only.

Altering capabilities (1 bit)

This field is used to indicate if the advertised parameter triggers a change in the node's capabilities.

CC:0070.04.0F.11.003 The value 1 MUST indicate that the advertised parameter alters the node capabilities when being changed.

CC:0070.04.0F.11.004 The value 0 MUST indicate that the advertised parameter does not alter the node capabilities when being changed.

Refer to Section 2.2.33.1.4 "Altering capabilities" flag.

Advanced (1 bit)

This field is used to indicate if the advertised parameter is to be presented in the "Advanced" parameter section in the controller GUI.

CC:0070.04.0F.11.005 The value 1 MUST indicate that the advertised parameter is an Advanced parameter.

CC:0070.04.0F.11.006 The value 0 MUST indicate that the advertised parameter is not an Advanced parameter.

No Bulk support (1 bit)

This field is used to advertise if the sending node supports Bulk Commands.

CC:0070.04.0F.11.007 The value 1 MUST indicate that the Bulk Commands will be ignored by the sending node.

CC:0070.04.0F.11.008 The value 0 MUST indicate that the Bulk Commands are supported by the sending node

CC:0070.04.0F.11.009 A sending node MUST always advertise the same value in this field, regardless of the parameter number.

2.2.33.5 Configuration Default Reset Command

This command is used to reset all configuration parameters to their default value.

Table 2.168: Configuration Default Reset Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION (0x70)							
Command = CONFIGURATION_DEFAULT_RESET (0x01)							

CC:0070.04.01.11.001 A node receiving this command MUST reset all its Configuration Parameters to their default value.

2.2.34 Controller Replication Command Class, version 1

The Controller Replication Command Class is used to copy scene and group data to another controlling node. The Command Class may be used in conjunction with a controller shift or when including a new controller to the network. It is OPTIONAL to use this command class during a controller shift or when including a new controller to the network.

2.2.34.1 Transfer group command

This command is used to replicate mappings between Group ID and Node ID.

Table 2.169: Transfer Group Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION (0x21)							
Command = CTRL_REPLICATION_TRANSFER_GROUP (0x31)							
Sequence Number							
Group ID							
Node ID							

Sequence Number (8 bits)

Sequence Number of this particular command.

Group ID (8 bits)

Group ID of the group that the node is member of.

NodeID (8 bits)

NodeID that belongs to the specified group.

2.2.34.2 Transfer Group Name Command

This command is used to replicate group names.

Table 2.170: Transfer Group Name Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION (0x21)							
Command = CTRL_REPLICATION_TRANSFER_GROUP_NAME (0x32)							
Sequence Number							
Group ID							
Group Name 1							
...							
Group Name N							

Sequence Number (8 bits)

Sequence Number of this particular command.

Group ID (8 bits)

Group ID associated with a specific group.

Group Name (N bytes)

The Group Name fields contain the assign group name in ASCII characters.

2.2.34.3 Transfer scene command

This command is used to replicate mappings between Scene ID and Node ID.

Table 2.171: Transfer Scene Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION (0x21)							
Command = CTRL_REPLICATION_TRANSFER_SCENE (0x33)							
Sequence Number							
Scene ID							
Node ID							
Level							

Sequence Number (8 bits)

Sequence Number of this particular command.

Scene ID (8 bits)

The scene ID is the parameter used to link together the different devices that takes part of a scene.

Node ID (8 bits)

The Node ID for a device that is part of the scene.

Level (8 bits)

The level is the parameter used for the specified scene.

2.2.34.4 Transfer Scene Name Command

This command is used to replicate scene names.

Table 2.172: Transfer Scene Name Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION (0x21)							
Command = CTRL_REPLICATION_TRANSFER_SCENE_NAME (0x34)							
Sequence Number							
Scene ID							
Scene Name 1							
...							
Scene Name N							

Sequence Number (8 bits)

Sequence Number of this particular command.

Scene ID (8 bits)

Scene ID associated with a specific scene.

Scene Name (N bytes)

The Scene Name fields contain the assign scene name in ASCII characters.

2.2.35 Demand Control Plan Configuration Command Class, version 1

The Demand Control Plan Configuration Command Class allows Utility Suppliers to issue and manage a list of Demand Control Plan (DCP) events to the end consumer.

The DCP Configuration commands are separated for the DCP monitoring commands in the Demand Control Plan Monitor Command Class, allowing the classes to be optionally supported at different security levels. (E.g. DCP monitoring commands could be supported using non-secure communication, while enabling strict secure-only communication for the DCP Configuration Command Class). Refer to the Security and Security 2 command classes for more details.

A DCP event contains information regarding criticality, products involved, requested reduction, time duration and if a certain rate (identified by a Demand Control Plan Rate ID, refer to the Rate Table Configuration Command Class) is associated with the event. When a DCP event is outdated, it is removed from the list. It is the utility supplier responsibility to prevent overflow of the list by query the number of free positions in the list before submitting a new DCP event to the list. Each DCP event is uniquely identified by a timestamp issued the Utility Supplier.

A DCP event may also include information, which enables devices not supporting this class to use in the Demand Control Plan through the Start & Stop Association Group functionality. The installer, the end user or Utility Supplier (remote management) configures the Association groups. During the configuration process the devices are selected and the association entries are created. These associations can additionally be configured with the specific Z-Wave commands (through the Association Command Configuration Command Class). If no Z-Wave commands are specified in the Associations groups, it is the responsibility of the device to issue the relevant commands based on Utility Supplier specific algorithms.

2.2.35.1 DCP list supported get command

This command is used to request the total size of the DCP list along with the number of free entries in the list.

The DCP List Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.173: DCP List Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_CONFIG (0x3A)							
Command = DCP_LIST_SUPPORTED_GET (0x01)							

2.2.35.2 DCP list supported report command

This command is used to provide the total size of the DCP list along with the number of free entries in the list.

Table 2.174: DCP List Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_CONFIG (0x3A)							
Command = DCP_LIST_SUPPORTED_REPORT (0x02)							
DCP List Size							
Free DCP List entries							

DCP List Size (8 bits)

This value specifies the DCP list size. 0x00 is reserved

Free DCP List entries (8 bits)

This value specifics the number of free entries for new DCP events. The value 0x00 specifies a full list.

2.2.35.3 DCP list set command

This command is used to place a new DCP event in the DCP list. Each DCP event is time stamped for future reference.

Table 2.175: DCP List Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_CONFIG (0x3A)							
Command = DCP_LIST_SET (0x03)							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							
DCP Rate ID							
Reserved							Number of DC
Generic Device Class 1							
Specific Device Class 1							
...							
Generic Device Class N							
Specific Device Class N							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Start Second Local Time							
Duration Hour Time							
Duration Minute Time							
Duration Second Time							
Event Priority							
Load shedding							
Start Association Group							
Stop Association Group							
Randomization interval							

Timestamp -Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Timestamp -Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Timestamp -Day (8 bits)

Specify the day of the month between 01 and 31.

Timestamp -Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Timestamp -Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Timestamp -Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

DCP Rate ID (8 bits)

Specify if a specific rate is applicable when participating in the DCP event. If an entry in the Rata table contains a matching DCP Rate ID this rate will be active for the duration of the event regardless of other parameters in the rate is not met.

DCP Rate ID usage Example

Prior to the DCP events a given rate table is configured defining when rates are active during a day. The Table contains two entries with DCP Rate IDs allowing the Utility Supplier to activate the rates outside of the time defined in the Table when a DCP event is placed in the DCP list with the corresponding DCP rate ID.

Prior to Jul14 2008 two DCP events are placed in the List by the Utility Supplier, which changes the Rate profile of Jul14 compared to an 'standard' day.

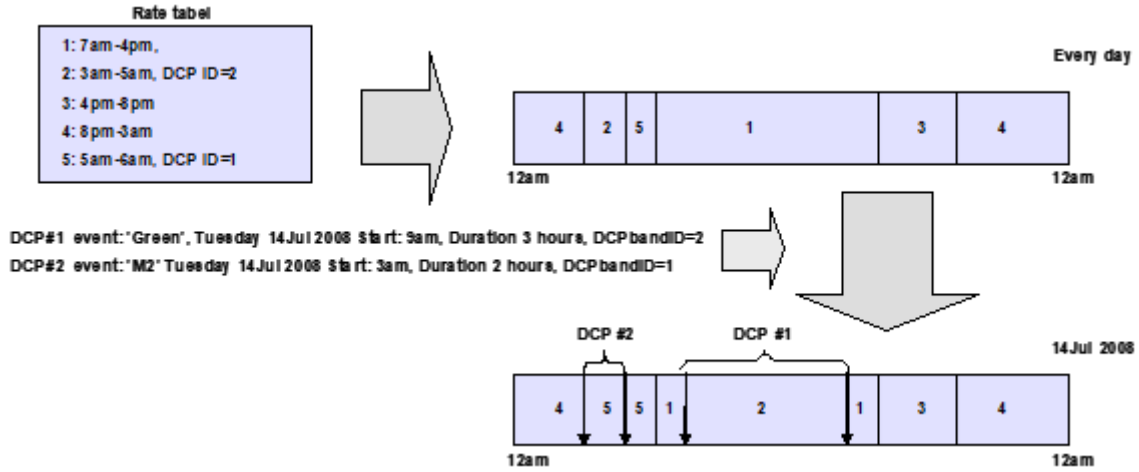


Figure 2.11: DCP Rate ID usage Example

Randomization interval (8 bits)

Specify the randomization interval in units of seconds, which must be applied as an offset to the start and stopping of the events as requested in the DCP event.

E.g. A value of 0x10 specifies that every device should randomly select a start and stop time offset between 0 and 16 seconds. This offset SHOULD be applied to the start and duration fields in the DCP event.

Number of DC (2 bits)

Specify the number of Generic/Specific Device Classes, which are requested to participate in the DCP event

Generic Device Class (8bits)

Specify the Generic Device Class identifier. Refer to [34] and Section 7.

Specific Device Class (8 bits)

Specify the Specific Device Class identifier. Refer to [34] and Section 7.

Start Year (16 bits)

Specify the year in the usual Gregorian calendar for the start of the event. The first byte (Year 1) is the most significant byte.

Start Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December) for the start of the event.

Start Day (8 bits)

Specify the day of the month for the start of the event between 01 and 31.

Start Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time for the start of the event.

Start Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time for the start of the event.

Start Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Duration Hour Time (8 bits)

Specify the number of complete hours of the event.

Duration Minute Time (8 bits)

Specify the number of complete minutes of the event.

Duration Second Time (8 bits)

Specify the number of complete seconds of the event.

Event Priority (8 bits)

The parameter specifies the priority of the DCP event. The High priority is used by the utility Supplier to mandate device participation and lower priorities are used by devices to voluntary to participate in the event and to which degree.

Table 2.176: Event Priority

Event Priority	Description	Device participation
0x00	Reserved	Voluntary
0x01	V1 - Green Energy	Voluntary
0x02	V2	Voluntary
0x03	V3	Voluntary
0x04	V4	Voluntary
0x05	V5	Voluntary
0x06	V6	Voluntary
0x07	V7	Voluntary
0x08	M1 - Emergency	Mandatory
0x09	M2	Mandatory
0x0A	M3	Mandatory
0x0B	M4	Mandatory
0x0C	Utility defined	Utility defined
0x0D	Utility defined	Utility defined
0x0E	Utility defined	Utility defined
0x0F	Utility defined	Utility defined

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Load shedding (8 bits)

Specify the load shedding in percentage of average consumption requested for the event. This load shedding will be applied to the devices with the device classes specified in the command.

This field MUST be in the range 0x01..0x64, which represents values in the range 1%..100%. All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Start Association group (8 bits)

Specify which association group (see the Association Command Class) should be activated when the event is started. The value 0x00 specifies no group should be activated.

If the device also supports the Association Command Configuration Command Class it possible to dynamically and remotely (from the Energy Supplier or others) to specify precisely which current and future Z-Wave commands should be send to which Z-Wave nodes.

The configuration of the Start Association group is done either manually by the installer or automatically when the device detect other relevant devices

Stop Association group (8 bits)

Specify which association group (see the Association Command Class) should be activated when the event is stopped. The value 0x00 specifies no group should be activated.

The configuration of the Stop Association group is done either manually by the installer or automatically when the device detect other relevant devices

Start Associating group and Stop Associating Group usage example

A small energy control system consisting of a device supporting the DCP command class and 3 Z-Wave devices which can participate in the application. NodeId 1: Setback Thermostat device, NodeId 8: Simple Thermostat device, NodeId 5: Multilevel Power Switch device.

Through the use of the Association Command Class and the Association Command Configuration Command Class the following Associations has been established in the device.

Group 1

Node1, Thermostat_Setback_Set(permanent override,energy saving mode)

Node8, Thermostat_Setpoint_Set(Heating setpoint#1, 19,5oC)

Node5, Multilevel_Switch:Set (Dimlevel =0x20)

Group 2

Node1, Thermostat_Setback_Set(No override,Setback = 0oC)

Node8, Thermostat_Setpoint_Set(Heating setpoint#1, 21,5oC)

Node5, Multilevel_Switch:Set (Dimlevel =0x40)

A DCP event can now specifically invoke Group1 when starting the event and invoking group2 when stopping the event by specifying Start Association Group=0x01 and Stop Association Group=0x02.

2.2.35.4 DCP list remove

This command is used to remove a DCP event from the DCP list.

Table 2.177: DCP List Remove

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_CONFIG (0x3A)							
Command = DCP_LIST_REMOVE (0x04)							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							

Refer to *DCP list report command* for description of fields

2.2.36 Demand Control Plan Monitor Command Class, version 1

The Demand Control Plan Monitor Command Class allows devices to monitor the list of Demand Control Plan (DCP). A DCP event contains information regarding criticality, products involved, requested reduction, time duration and if a certain rate (identified by a Demand Control Plan Rate ID, refer to the Rate Table Configuration Command Class) is associated with the event. When a DCP event is outdated, it is removed from the list. Each DCP event is uniquely identified by a timestamp issued the Utility Supplier.

2.2.36.1 DCP list get command

This command is used to request the pending DCP event in a device.

The DCP List Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.178: DCP List Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR (0x3B)							
Command = DCP_LIST_GET (0x01)							

2.2.36.2 DCP list report command

This command reports the pending DCP event in a device. If more than one DCP event is pending - the reports will be submitted in chronically order as to when the events was placed on the list. Newest entry will be reported first.

Table 2.179: DCP List Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR (0x3B)							
Command = DCP_LIST_REPORT (0x02)							
Reports to Follow							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							
DCP ID							
Reserved							Number of DC
Generic Device Class 1							
Specific Device Class 1							
...							
Generic Device Class N							
Specific Device Class N							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							

continues on next page

Table 2.179 – continued from previous page

7	6	5	4	3	2	1	0
Start Minute Local Time							
Start Second Local Time							
Duration Hour Time							
Duration Minute Time							
Duration Second Time							
Event Priority							
Load shedding							
Start Association Group							
Stop Association Group							
Randomization interval							

Reports to Follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Refer to DCP List Set command class (*DCP list set command*) for detailed description of the fields.

2.2.36.3 DCP event status get

This command is used to query the status of a specific DCP event in the DCP list.

The DCP Event Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.180: DCP Event Status Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR (0x3B)							
Command = DCP_EVENT_STATUS_GET (0x03)							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							

Refer to DCP List Set command class (*DCP list set command*) for detailed description of the fields.

2.2.36.4 DCP event status report

This command is used to provide the status of a specific DCP event in the DCP list.

Table 2.181: DCP Event Status Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR (0x3B)							
Command = DCP_EVENT_STATUS_GET (0x03)							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							
Event status							

Refer to DCP List report command class (*DCP list report command*) for description of fields

Event Status (8 bits)

The field contains the status of the event.

Table 2.182: DCP Event Status Report::Event Status encoding

Event status	Description
0x01	Event Started
0x02	Event Completed
0x03	Event Rejected by the user
0x04	Event not Applicable

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.37 Door Lock Command Class, version 1-2

The Door Lock Command Class is used to operate and configure a door lock device.

The Door Lock Command Class is an actuator control command class. Refer to Section 2.1.6.

2.2.37.1 Compatibility considerations

A device supporting Door Lock CC, Version 2 MUST support Door Lock CC, version 1.

CC:0062.02.00.21.001 The Door Lock Command Class, version 2 adds the "unknown" state to the Door Lock Operation Report Command.

CC:0062.01.00.21.001 A supporting node MAY implement a subset of the features represented by the Door Lock Mode, Door Handles Mode and Door Condition fields which are provided by the commands of this Command Class.

2.2.37.2 Door Lock Operation Set Command

This command is used to set the operation mode of a supporting door lock device.

Table 2.183: Door Lock Operation Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_OPERATION_SET (0x01)							
Door Lock Mode							

Door Lock Mode (8 bits)

The Door Lock Mode field is used to specify the operation mode of the door lock device.

CC:0062.01.01.11.001 The encoding of this field MUST be according to Table 2.184.

Table 2.184: Door Lock Operation Set::Mode

Mode	Description	Version
0x00	Door Unsecured 1)	1
0x01	Door Unsecured with timeout 2)	1
0x10	Door Unsecured for inside Door Handles 1)	1
0x11	Door Unsecured for inside Door Handles with timeout 2)	1
0x20	Door Unsecured for outside Door Handles 1)	1
0x21	Door Unsecured for outside Door Handles with timeout 2)	1
0xFF	Door Secured	1

1) Constant mode. Door will be unsecured until set to secured mode by another command.

2) Timeout mode. Fallback to secured mode after timeout has expired.

CC:0062.01.01.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.01.01.11.003 A controlling node MUST NOT specify modes using timeout-based fall back if the Operation Type of the supporting device is set to Constant operation (set by Door Lock Configuration Set).

CC:0062.01.01.11.004 A supporting node MUST ignore modes using timeout-based fall back if Operation Type is set to Constant operation.

CC:0062.01.01.11.006 A supporting node MUST apply the constant modes without any timeout even if it is set to Timed Operation Type.

CC:0062.01.01.13.001 A supporting node MAY implement a subset of the Door Lock Modes defined by Table 36.

CC:0062.01.01.11.005 A supporting node MUST accept the Door Lock Mode values 0x00 and 0xFF.

2.2.37.3 Door Lock Operation Get Command

This command is used to request the status of a door lock device.

CC:0062.01.02.11.001 The Door Lock Operation Report command MUST be returned in response to this command.

CC:0062.01.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:0062.01.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.185: Door Lock Operation Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_OPERATION_GET (0x02)							

2.2.37.4 Door Lock Operation Report Command

This command is used to advertise the status of a door lock device.

Table 2.186: Door Lock Operation Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_OPERATION_REPORT (0x03)							
Door Lock Mode							
Outside Door Handles Mode				Inside Door Handles Mode			
Door Condition							
Remaining Lock Time Minutes							
Remaining Lock Time Seconds							

Door Lock Mode (8 bits)

CC:0062.01.03.11.001 The Door Lock Mode field MUST advertise the mode of the door lock device.

CC:0062.01.03.11.002 The encoding of this field MUST be according to Table 2.187.

Table 2.187: Door Lock Operation Report::Door Lock Mode

Mode	Description	Version
0x00	Door Unsecured 1)	1
0x01	Door Unsecured with timeout 2)	1
0x10	Door Unsecured for inside Door Handles 1)	1
0x11	Door Unsecured for inside Door Handles with timeout 2)	1
0x20	Door Unsecured for outside Door Handles 1)	1
0x21	Door Unsecured for outside Door Handles with timeout 2)	1
0xFE	Door/Lock State Unknown 3)	2
0xFF	Door Secured	1

1) Constant mode. Door will be unsecured until set back to secured mode by command

2) Timeout mode. Fallback to secured mode after timeout has expired

3) Bolt is not fully retracted/engaged

CC:0062.01.03.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.01.03.13.001 A supporing node MAY advertise a subset of the available Door Lock Mode values.

CC:0062.01.03.11.004

A controlling node MUST accept all Door Lock Mode values.

CC:0062.01.03.12.001 The Door Lock Mode field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

Outside Door Handles Mode (4 bits)

CC:0062.01.03.11.005 This field MUST advertise the status of each individual outside door handle.

CC:0062.01.03.11.006 The encoding of the Outside Door Handles Mode bitmask field MUST be according to Table 2.188 and Table 2.189.

Table 2.188: Door Handles Mode bitmask

Bit 3	Bit 2	Bit 1	Bit 0
Handle 4	Handle 3	Handle 2	Handle 1

Table 2.189: Door Handles Mode bit encoding

Bit value	Description
'0'	Disabled
'1'	Enabled

CC:0062.01.03.11.007 The value '0' MUST signify that the actual handle cannot open the door locally.

CC:0062.01.03.11.008 The value '1' MUST signify that the actual handle can open the door locally.

CC:0062.01.03.13.002 A supporting node MAY advertise just a subset of the available Door Handles Mode values.

CC:0062.01.03.11.009 A controlling node MUST accept all Door Handles Mode values.

Inside Door Handles Mode (4 bits)

CC:0062.01.03.11.00A This field MUST advertise the status of each individual inside door handle.

CC:0062.01.03.11.00B The encoding of the Inside Door Handles Mode bitmask field MUST be according to Table 2.188 and Table 2.189.

CC:0062.01.03.13.003 A supporting node MAY advertise just a subset of the available Door Handles Mode values.

CC:0062.01.03.11.00C A controlling node MUST accept all Door Handles Mode values.

Door Condition (8 bits)

CC:0062.01.03.11.00D The Door Condition field MUST advertise the status of the door lock components.

CC:0062.01.03.11.00E The encoding of the Door Condition bitmask field MUST be according to Table 2.203.

CC:0062.01.03.13.004 A supporting node MAY advertise just a subset of the available Door Condition values.

CC:0062.01.03.11.0010 A controlling node MUST accept all Door Condition values.

Remaining Lock Time Minutes (8 bits)

CC:0062.01.03.11.011 This field MUST advertise the remaining time before the door lock will automatically be locked again.

CC:0062.01.03.11.012 The encoding of the *Remaining Lock Time Minutes* field MUST be according to Table 2.190. The time the supporting node stays unlocked MUST be determined by combining the *Remaining Lock Time Minutes* and *Remaining Lock Time Seconds* fields.

Table 2.190: Door Lock Operation Report::Remaining Lock Time Minutes

Value	Operation
0x00..0xFD	Unlocked 0 .. 253 minutes (Operation Type = Timed Operation)
0xFE	No unlocked period (Operation Type = Constant Operation)

CC:0062.01.03.11.013

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.01.03.13.005 A supporting node MAY advertise just a subset of the available Remaining Lock Time values.

CC:0062.01.03.11.014 A controlling node MUST accept all Remaining Lock Time values.

Remaining Lock Time Seconds (8 bits)

CC:0062.01.03.11.015 This field MUST advertise the remaining time before the door lock will automatically be locked again.

CC:0062.01.03.11.016 The encoding of the Remaining Lock Time Seconds field MUST be according to Table 2.191. The time to stay unlocked MUST be determined by combining the *Remaining Lock Time Minutes* and *Remaining Lock Time Seconds* fields.

Table 2.191: Door Lock Operation Report::Remaining Lock Time Seconds

Value	Operation
0..59 (0x00..0x3B)	Unlocked 0 .. 59 seconds (Operation Type = Timed Operation)
254 (0xFE)	No unlocked period (Operation Type = Constant Operation)

CC:0062.01.03.11.017 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.01.03.13.006 A supporting node MAY advertise just a subset of the available Lock Timeout values.

CC:0062.01.03.11.018 A controlling de MUST accept all Lock Timeout values.

2.2.37.5 Door Lock Configuration Set Command

This command is used to set the configuration of a supporting door lock device.

CC:0062.01.04.11.001 A door lock device MUST be able to operate with the factory default settings.

Table 2.192: Door Lock Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CONFIGURATION_SET (0x04)							
Operation Type							
Outside Door Handles Enabled				Inside Door Handles Enabled			
Lock Timeout Minutes							
Lock Timeout Seconds							

Operation Type (1 byte)

CC:0062.01.04.11.002 The Operation Type field MUST be set according to Table 2.193. When timed operation is specified, the Lock Timeout Minutes and Lock Timeout Seconds fields MUST be set to valid values.

Table 2.193: Door Lock Operation Type

Operation Type	Description	Valid Lock Timeout values
0x01	Constant operation	Minutes = 0xFE Seconds= 0xFE
0x02	Timed operation	Minutes = 0x00..0xFD Seconds= 0x00..0x3B

CC:0062.01.04.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.01.04.13.001 A supporting node MAY accept just a subset of the available Operation Type values.

CC:0062.01.04.11.004

A supporting node MUST accept the Operation Type value 0x01.

Outside Door Handles Enabled (4 bits)

This field is used to advertise the status of each individual outside door handle.

CC:0062.01.04.11.005 The encoding of the Outside Door Handles Mode bitmask field MUST be according to Table 2.188 and Table 2.189.

CC:0062.01.04.13.002 A supporting node MAY ignore the Door Handles Mode value.

Inside Door Handles Enabled (4 bits)

CC:0062.01.04.11.006 This field MUST advertise the status of each individual inside door handle.

CC:0062.01.04.11.007 The encoding of the Inside Door Handles Mode bitmask field MUST be according to Table 2.188 and Table 2.189.

CC:0062.01.04.13.003 A supporting node MAY ignore the Door Handles Mode value.

Lock Timeout Minutes (1 byte)

CC:0062.01.04.11.008 This field MUST specify the time that a door lock must wait before automatically being locked again.

CC:0062.01.04.11.009 The encoding of the Lock Timeout Minutes field MUST be according to Table 2.190. The time to stay unlocked MUST be determined by combining the *Lock Timeout Minutes* and *Lock Timeout Seconds* fields.

CC:0062.01.04.13.004 A supporting node MAY ignore the Lock Timeout values if it implements only the Operation Type = 0x01 (Constant operation).

Lock Timeout Seconds (1 byte)

CC:0062.01.04.11.00A This field MUST specify the time that a door lock must wait before automatically being locked again.

CC:0062.01.04.11.00B The encoding of the Lock Timeout Seconds field MUST be according to Table 2.191. The time to stay unlocked MUST be determined by combining the *Lock Timeout Minutes* and *Lock Timeout Seconds* fields.

CC:0062.01.04.13.005 A supporting node MAY ignore the Lock Timeout values if it implements only the Operation Type = 0x01 (Constant operation).

2.2.37.6 Door Lock Configuration Get Command

This command is used to request the configuration parameters of a door lock device.

CC:0062.01.05.11.001 The Door Lock Configuration Report command MUST be returned in response to this command.

CC:0062.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:0062.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.194: Door Lock Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CONFIGURATION_GET (0x05)							

2.2.37.7 Door Lock Configuration Report Command

This command is used to advertise the configuration parameters of a door lock device.

Table 2.195: Door Lock Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CONFIGURATION_REPORT (0x06)							
Operation Type							
Outside Door Handles Enabled				Inside Door Handles Enabled			
Lock Timeout Minutes							
Lock Timeout Seconds							

For fields' description, refer to Section 2.2.37.5 *Door Lock Configuration Set Command*.

2.2.38 Door Lock Command Class, version 3

The Door Lock Command Class is used to operate and configure a door lock device.

The Door Lock Command Class is an actuator control command class. Refer to Section 2.1.6.

Commands and fields not described in version 3 remain unchanged from version 2.

2.2.38.1 Compatibility considerations

CC:0062.03.00.21.001 A node supporting the Door Lock Command Class, version 3 MUST support the Door Lock Command Class, version 2.

Version 3 adds duration and target value reporting to the Door Lock Operation Report Command.

2.2.38.2 Door Lock Operation Report Command

This command is used to advertise the status of a door lock device.

Table 2.196: Door Lock Operation Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_OPERATION_REPORT (0x03)							
Current Door Lock Mode							
Outside Door Handles Enabled				Inside Door Handles Enabled			
Door Condition							
Remaining Lock Time Minutes							
Remaining Lock Time Seconds							
Target Door Lock Mode							
Duration							

Fields not described below remain unchanged from previous versions.

Current Door Lock Mode (8 bits)

CC:0062.03.03.11.001 The Current Door Lock Mode field MUST advertise the current mode of the door lock device.

CC:0062.03.03.11.002 The encoding of this field MUST be according to Table 2.187.

CC:0062.03.03.12.001 An Operation Set Command may be used to initiate a transition to a new target mode. The device may be queried for its current mode while in transition to a new mode. The response to an Operation Get SHOULD be the current mode of the device hardware, e.g. Door/Lock State Unknown. (applies to Version 2 and newer).

Target Door Lock Mode (8 bits)

CC:0062.03.03.11.003 The Target Door Lock Mode field MUST advertise the target mode of an ongoing transition or the most recent transition.

CC:0062.03.03.11.004 The encoding of this field MUST be according to Table 2.187.

Duration (8 bits)

CC:0062.03.03.12.002 The Duration field SHOULD advertise the remaining time before the target mode is reached.

CC:0062.03.03.11.005 The encoding of this field MUST be according to Table 2.8.

2.2.39 Door Lock Command Class, version 4

2.2.39.1 Terminology

A door lock supports at least 2 **modes**: secured (it cannot be opened with handles) and unsecured (it can be opened with handles). It may support more modes where the node is for example secured only for the inside handles.

A door lock may have several components such as a **latch**, a **bolt** and **handles**.

A latch, which can also be called a latch bolt or spring bolt, usually locks the door automatically when closing it. The latch bolt is disengaged (retracted) typically when the user turns the door handle, which via the lockset's mechanism, manually retracts the latch bolt, allowing the door to open.

Handles allow to retract the door's latch.

A bolt, also named dead bolt and hook bolt, allows to secure the door. When the bolt is out or extended, the door lock is usually secured (locked) but there are locks that are unlocked even then the bolt is out which will withdraw the bolt when user turns the door handle (which also retract the latch). The position of the bolt is not directly related to the Door Lock Mode.

A supporting door lock runs in **constant operation** by default. In constant operation, the door lock stays in the same mode until receiving a new command or being locally actuated. A door lock may support **timed operation**. Timed operation has its own subset of modes, which after being activated will automatically return to the secured mode after the configured Lock Timeout.

An **auto-relock** functionality can be supported for returning automatically to secured mode some given time after any operation mode change, including constant operation or changes initiated physically by a user (unlocking with key, thumb turn, etc). Auto-relock should not have any effect when the lock runs in Timed Operation

A door lock node may have a **twist assist** mechanism. When a user starts rotating a key or the thumb turn to unlock the door, a motor starts rotating to help the user completing the movement. It can be helpful for disabled users.

A node may support a **hold and release** functionality for spring latches. This is useful for locks having a handle that cannot retract the door spring latch on one side. This functionality defines how long the spring latch must be retracted after the door mode has been changed to unsecured.

A **block-to-block** functionality can be activated for locks having separate lock block on each side of the door and cannot sense local changes on one side. Enabling the block-to-block feature will make the door lock try to execute a lock or unlock command on both sides regardless of the last known lock mode (representing the side that can be sensed). When this feature is disabled, the lock will ignore a command indicating a mode if it is identical to the last known lock mode.

2.2.39.2 Compatibility considerations

CC:0062.04.00.21.001 The Door Lock Command Class, version 4 is backwards compatible with the Door Lock Command Class, version 3. Fields and commands not mentioned in this version MUST remain unchanged from the Door Lock Command Class, version 3.

Commands for advertising node capabilities are introduced in this version:

- Door Lock Capabilities Get Command
- Door Lock Capabilities Report Command

These commands are extended in order to provide the auto-relock, twist assist and block-to-block functionalities.

- Door Lock Configuration Set Command
- Door Lock Configuration Report Command

2.2.39.3 Door Lock Operation Set Command

This command is used to set the mode of a supporting door lock node.

Table 2.197: Door Lock Operation Set Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_OPERATION_SET (0x01)							
Door Lock Mode							

Door Lock Mode (8 bits)

CC:0062.04.01.11.001 This field is used to specify the operation mode that the supporting node MUST assume. This field MUST be encoded according to Table 2.198.

Table 2.198: Door Lock Operation Set::Mode encoding

Value	Mode Description	Operation Type	Version
0x00	Door Unsecured	Constant	1
0x01	Door Unsecured with timeout	Timed	1
0x10	Door Unsecured for inside Door Handles	Constant	1
0x11	Door Unsecured for inside Door Handles with timeout	Timed	1
0x20	Door Unsecured for outside Door Handles	Constant	1
0x21	Door Unsecured for outside Door Handles with timeout	Timed	1
0xFF	Door Secured	Constant	1

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.04.01.51.001 A controlling node MUST NOT specify modes using timeout-based fall back if the Operation Type of the supporting device is set to Constant operation (set by Door Lock Configuration Set).

CC:0062.04.01.11.002 A supporting node MUST ignore modes using timeout-based fall back if Operation Type is set to Constant operation.

CC:0062.04.01.11.003 A supporting node MUST apply the constant modes without any timeout even if it is configured to run in Timed Operation.

CC:0062.04.01.11.004 A supporting node MUST ignore the command if the value specified in this field is not advertised as supported in the Door Lock Capabilities Report Command.

CC:0062.04.01.11.005 A supporting node MUST support the door lock mode values 0x00 and 0xFF.

2.2.39.4 Door Lock Operation Get Command

This command is used to request the mode of a supporting door lock node.

CC:0062.04.02.11.001 The Door Lock Operation Report Command MUST be returned in response to this command.

CC:0062.04.02.51.001 This command MUST NOT be issued via multicast addressing.

CC:0062.04.02.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.199: Door Lock Operation Get Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_OPERATION_GET (0x02)							

2.2.39.5 Door Lock Operation Report Command

This command is used to advertise the mode of a supporting door lock node.

Table 2.200: Door Lock Operation Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_OPERATION_REPORT (0x03)							
Current Door Lock Mode							
Outside Door Handles Mode				Inside Door Handles Mode			
Door Condition							
Remaining Lock Time Minutes							
Remaining Lock Time Seconds							
Target Door Lock Mode							
Duration							

Current Door Lock Mode (8 bits)

CC:0062.04.03.11.001 This field MUST advertise the current door lock mode of the sending node. This field MUST be encoded according to Table 2.201.

Table 2.201: Door Lock Operation Report::Mode encoding

Value	Mode Description	Operation Type	Version
0x00	Door Unsecured	Constant (or timed)	1
0x01	Door Unsecured with timeout	Timed	1
0x10	Door Unsecured for inside Door Handles	Constant (or timed)	1
0x11	Door Unsecured for inside Door Handles with timeout	Timed	1
0x20	Door Unsecured for outside Door Handles	Constant (or timed)	1
0x21	Door Unsecured for outside Door Handles with timeout	Timed	1
0xFE	Door mode unknown (e.g. bolt not fully retracted)	N/A	2
0xFF	Door Secured	Constant (or timed)	1

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.04.03.12.001 A sending node SHOULD set this field to 0xFE if sending this command while performing a transition between 2 modes.

Outside Door Handles Mode (4 bits)

This field is used to advertise the status of each individual outside door handle.

CC:0062.04.03.11.002 This field MUST be treated as a bitmask field and MUST be encoded according to Table 2.202.

Table 2.202: Door Lock Operation Report::Door Handles Mode bitmask encoding

Bit 3	Bit 2	Bit 1	Bit 0
Handle 4	Handle 3	Handle 2	Handle 1

CC:0062.04.03.11.003 The value 0 MUST indicate that the actual handle cannot open the door locally.

The value 1 MUST indicate that the actual handle can open the door locally.

Inside Door Handles Mode (4 bits)

This field is used to advertise the status of each individual inside door handle.

CC:0062.04.03.11.004

This field MUST be treated as a bitmask field and MUST be encoded according to Table 2.202.

CC:0062.04.03.11.005 The value 0 MUST indicate that the actual handle cannot open the door locally.

The value 1 MUST indicate that the actual handle can open the door locally.

Door Condition (8 bits)

This field is used to advertise the state of the door lock components.

CC:0062.04.03.11.006 This field MUST be encoded according to Table 2.203.

Table 2.203: Door Lock Operation Report::Door Condition bitmask encoding

Value	Bit 7..3	Bit 2	Bit 1	Bit 0
bit set to '0'	*Reserved*	Latch Open (out, engaged, extended)	Bolt Locked (out, extended)	Door Open
bit set to '1'	*Reserved*	Latch Closed (in, disengaged, retracted)	Bolt Unlocked (in, retracted)	Door Closed

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

CC:0062.04.03.11.007 If a door component is not advertised as supported in the Door Lock Capabilities Report Command, the corresponding bit MUST be set to 0 and MUST be ignored by a controlling node.

Remaining Lock Time Minutes (8 bits)

This field is used to advertise the remaining time before the door lock mode will automatically be changed to "secured" if operating in timed operation with a timed operation mode.

CC:0062.04.03.11.008 This field MUST be encoded according to Table 2.204. The remaining time that the sending node will stay in the current mode MUST be determined by combining this field and the *Remaining Lock Time Seconds* field.

Table 2.204: Door Lock Operation Report::Remaining Lock Time Minutes encoding

Value	Description
0..253 (0x00..0xFD)	0..253 minutes (Operation Type and Current mode = Timed Operation)
254 (0xFE)	Undefined

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.04.03.11.009 A node currently set in a constant operation type mode MUST set this field to 0xFE. However, if unsecured with the auto-relock functionality active, it MAY advertise the time remaining before auto-relock secures the lock again.

Remaining Lock Time Seconds (8 bits)

This field is used to advertise the remaining time before the door lock mode will automatically be changed to "secured" if operating in timed operation with a timed operation mode.

CC:0062.04.03.11.00A This field MUST be encoded according to Table 2.205. The remaining time that the sending node will stay in the current mode MUST be determined by combining this field and the *Remaining Lock Time Minutes* field.

Table 2.205: Door Lock Operation Report::Remaining Lock Time Seconds encoding

Value	Description
0..59 (0x00..0x3B)	0..59 seconds (Operation Type and mode= Timed Operation)
254 (0xFE)	Undefined

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0062.04.03.11.00B A node currently set in a constant operation type mode MUST set this field to 0xFE. However, if unsecured with the auto-relock functionality active, it MAY advertise the time remaining before auto-relock secures the lock again.

Target Door Lock Mode (8 bits)

This field is used to advertise the target mode of an ongoing transition or of the most recent transition.

CC:0062.04.03.11.00C This field MUST be encoded according to Table 2.201. This field MUST be set to the mode specified in the most-recent received Door Lock Operation Set Command.

CC:0062.04.03.11.00D If the node is going to change mode due to the auto-relock functionality or a timed operation, this field MUST NOT advertise the future mode and MUST be set to the same value as the *Current Door Lock Mode* field.

CC:0062.04.03.11.00E If no transition is ongoing, this field MUST be set to the same value as the *Current Door Lock Mode* field.

CC:0062.04.03.11.00F If the *Current Door Lock Mode* is set to 0xFE, this field MUST indicate the mode is it going to assume when the ongoing transition has been completed.

Duration (8 bits)

CC:0062.04.03.12.002 This field SHOULD advertise the estimated remaining time before the mode indicated in the *Target Door Lock Mode* field is reached.

CC:0062.04.03.11.010 This field MUST be set to 0 if the *Target Door Lock Mode* and *Current Door Lock Mode* fields are set to the same value.

CC:0062.04.03.11.011 The encoding of this field MUST be according to Table 2.10.

2.2.39.6 Door Lock Configuration Set Command

This command is used to set the configuration of a supporting door lock node.

CC:0062.04.04.11.001 A supporting node MUST be able to operate with the factory default configuration.

Table 2.206: Door Lock Configuration Set Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CONFIGURATION_SET (0x04)							
Operation Type							
Outside Door Handles Enabled				Inside Door Handles Enabled			
Lock Timeout Minutes							
Lock Timeout Seconds							
Auto-relock time 1 (MSB)							
Auto-relock time 2 (LSB)							
Hold and release time 1 (MSB)							
Hold and release time 2 (LSB)							
Reserved						BTB	TA

Operation Type (1 byte)

CC:0062.04.04.11.002

This field is used to set the operation type at the supporting node. This field MUST be encoded according to Table 2.207.

Table 2.207: Door Lock Configuration Set::Operation Type encoding

Value	Description	Version
0x01	Constant operation	1
0x02	Timed operation	1

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

If this field is set to timed operation (0x02), the *Lock Timeout Minutes* and *Lock Timeout Seconds* fields MUST be set to values different than 0xFE. CC:0062.04.04.11.003

A supporting node MUST ignore this field if it is set to an operation type that is not advertised as supported in the Door Lock Capabilities Report Command. CC:0062.04.04.11.004

Outside Door Handles Enabled (4 bits)

This field is used to enable or disable each individual outside door handle at the supporting node. This field MUST be treated as a bitmask and MUST be encoded according to Table 2.202. CC:0062.04.04.11.005

The value 0 MUST indicate that the actual handle cannot open the door locally. CC:0062.04.04.11.006

The value 1 MUST indicate that the actual handle can open the door locally.

A supporting node MUST ignore an outside door handle mode value if it advertises no support for the corresponding handle in the Door Lock Capabilities Report Command. CC:0062.04.04.11.007

Inside Door Handles Enabled (4 bits)

This field is used to enable or disable each individual inside door handle at the supporting node. This field MUST be treated as a bitmask and MUST be encoded according to Table 2.202. CC:0062.04.04.11.008

The value 0 MUST indicate that the actual handle cannot open the door locally. CC:0062.04.04.11.009

The value 1 MUST indicate that the actual handle can open the door locally.

A supporting node MUST ignore an inside door handle mode value if it advertises no support for the corresponding handle in the Door Lock Capabilities Report Command. CC:0062.04.04.11.00A

Lock Timeout Minutes (8 bits)

This field is used to specify the time that the supporting node must wait before returning to the secured mode when receiving timed operation modes in a Door Lock Operation Set Command.

Values in the range 0x00..0xFD MUST indicate the actual number of minutes. CC:0062.04.04.11.00B

The value 0xFE MUST indicate that this field is undefined.

This field MUST be set to 0xFE if the *Operation Type* field is set to 0x01 (constant operation) CC:0062.04.04.11.00C

This field MUST be in the range 0x00..0xFD if the *Operation Type* field is set to 0x02 (timed operation)

The time to stay unlocked MUST be determined by combining this field and the Lock Timeout Seconds fields. CC:0062.04.04.11.00D

A supporting node MUST ignore this field if it is set to the timed operation type is not advertised as supported in the Door Lock Capabilities Report Command. CC:0062.04.04.11.00E

Lock Timeout Seconds (8 bits)

This field is used to specify the time that the supporting node must wait before returning to the secured mode when receiving timed operation modes in a Door Lock Operation Set Command.

Values in the range 0x00..0x3B MUST indicate the actual number of seconds. CC:0062.04.04.11.00F

The value 0xFE MUST indicate that this field is undefined.

CC:0062.04.04.11.010 This field MUST be set to 0xFE if the *Operation Type* field is set to 0x01 (constant operation)

This field MUST be in the range 0x00..0x3B if the *Operation Type* field is set to 0x02 (timed operation)

CC:0062.04.04.11.011 The time to stay unlocked MUST be determined by combining this field and the *Lock Timeout Minutes* field.

CC:0062.04.04.11.012 A supporting node MUST ignore this field if it is set to the timed operation type is not advertised as supported in the Door Lock Capabilities Report Command.

Auto-relock time (16 bits)

This field is used to specify the time setting in seconds for auto-relock functionality.

CC:0062.04.04.11.013 This field MUST be ignored by a node advertising no support for the auto-relock functionality in the Door Lock Capabilities Report Command.

CC:0062.04.04.11.014 The value 0 MUST indicate that the auto-relock functionality is disabled and the door lock MUST NOT return to secured mode automatically. (unless using Timed Operation)

CC:0062.04.04.11.015 Values in the range 1..65535 MUST indicate that the auto-relock functionality is enabled and the supporting node MUST return to secured mode after the time in seconds indicated by this field.

Hold and release time (16 bits)

This field is used to specify the time setting in seconds for letting the latch retracted after the supporting node's mode has been changed to unsecured.

CC:0062.04.04.11.016 This field MUST be ignored by a node advertising no support for the hold and release functionality in the Door Lock Capabilities Report Command.

CC:0062.04.04.11.017 The value 0 MUST indicate that the hold and release functionality is disabled and the supporting node MUST NOT keep the latch retracted when the door lock mode becomes unsecured.

CC:0062.04.04.11.018 Values in the range 1..65535 MUST indicate that the hold and release functionality is enabled and the door lock latch MUST keep open according to the time in seconds indicated by this field.

TA (Twist assist) (1 bit)

CC:0062.04.04.11.019 This field MUST indicate if the twist assist functionality is enabled.

CC:0062.04.04.11.01A This field MUST be ignored by a node advertising no support for the twist assist functionality in the Door Lock Capabilities Report Command.

CC:0062.04.04.11.01B The value 0 MUST indicate that the twist assist functionality MUST be disabled at the supporting node.

The value 1 MUST indicate that the twist assist functionality MUST be enabled at the supporting node.

BTB (Block-to-block) (1 bit)

This field is used to indicate if the block-to-block functionality is enabled.

CC:0062.04.04.11.01C This field MUST be ignored by a node advertising no support for the block-to-block functionality in the Door Lock Capabilities Report Command.

CC:0062.04.04.11.01D The value 0 MUST indicate that the block-to-block functionality is disabled and the supporting node MAY ignore Door Lock Operation Set Commands if it detects to be already in the specified mode.

CC:0062.04.04.11.01E The value 1 MUST indicate that the block-to-block functionality is enabled and the supporting node MUST activate its motors until blocked to try to reach the mode indicated by Door Lock Operation Set Commands, even if it detects to be already in the specified mode.

2.2.39.7 Door Lock Configuration Get Command

This command is used to request the configuration parameters of a door lock device.

CC:0062.04.05.11.001 The Door Lock Configuration Report Command MUST be returned in response to this command.

CC:0062.04.05.51.001 This command MUST NOT be issued via multicast addressing.

CC:0062.04.05.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.208: Door Lock Configuration Get Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CONFIGURATION_GET (0x05)							

2.2.39.8 Door Lock Configuration Report Command

This command is used to advertise the configuration parameters of a door lock device.

Table 2.209: Door Lock Configuration Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CONFIGURATION_REPORT (0x06)							
Operation Type							
Outside Door Handles Enabled				Inside Door Handles Enabled			
Lock Timeout Minutes							
Lock Timeout Seconds							
Auto-relock time 1 (MSB)							
Auto-relock time 2 (LSB)							
Hold and release time 1 (MSB)							
Hold and release time 2 (LSB)							
Reserved						BTB	TA

Fields not described below MUST be according to descriptions in Section 2.2.39.6 *Door Lock Configuration Set Command*.

Outside Door Handles Enabled (4 bits)

This field is used advertise if each individual outside door handle is enabled at the supporting node. This field MUST be treated as a bitmask and MUST be encoded according to Table 2.202.

Inside Door Handles Enabled (4 bits)

This field is used advertise if each individual inside door handle is enabled at the supporting node. This field MUST be treated as a bitmask and MUST be encoded according to Table 2.202.

2.2.39.9 Door Lock Capabilities Get Command

This command is used to request the Door Lock capabilities of a supporting node.

CC:0062.04.07.11.001 The Door Lock Capabilities Report Command MUST be returned in response to this command.

CC:0062.04.07.51.001 This command MUST NOT be issued via multicast addressing.

CC:0062.04.07.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.210: Door Lock Capabilities Get Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CAPABILITIES_GET (0x07)							

2.2.39.10 Door Lock Capabilities Report Command

This command is used to advertise the Door Lock capabilities supported by the sending node.

Table 2.211: Door Lock Capabilities Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK (0x62)							
Command = DOOR_LOCK_CAPABILITIES_REPORT (0x08)							
Reserved			Supported Operation type Bit Mask Length				
Supported Operation Type Bit Mask 1							
...							
Supported Operation Type Bit Mask N							
Supported Door Lock Mode List Length							
Supported Door Lock Mode 1							
...							
Supported Door Lock Mode M							
Supported Outside Handle Modes Bitmask				Supported Inside Handle Modes Bitmask			
Supported door components							
Reserved				ARS	HRS	TAS	BTBS

Reserved

CC:0062.04.08.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Supported Operation Type Bit Mask Length (5 bits)

This field is used to advertise the length in bytes of the *Supported Operation Type Bit Mask* field carried in the command.

CC:0062.04.08.11.002 This field MUST be set to the minimum value which allows to advertise all supported Operation types.

Supported Operation Type Bit Mask (N bytes)

CC:0062.04.08.11.003 This field is used to advertise the supported Door Lock Operation Types values. The length of this field in bytes MUST match the value advertised in the Supported Operation Type Bit Mask Length field.

CC:0062.04.08.11.004 This field MUST be encoded as follows:

- Bit 0 in Bit Mask 1 represents Operation Type 0x00 (Reserved)
- Bit 1 in Bit Mask 1 represents Operation Type 0x01 (Constant operation)
- Bit 2 in Bit Mask 1 represents Operation Type 0x02 (Timed operation)

- ...

Valid Operation Type values are defined in Table 2.207.

CC:0062.04.08.11.005 If an Operation Type is supported, the corresponding bit MUST be set to '1'.

If an Operation Type is not supported, the corresponding bit MUST be set to '0'.

CC:0062.04.08.11.006 All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

CC:0062.04.08.11.007 A node MUST support the constant operation type 0x01.

Supported Door Lock Mode List Length (8 bits)

CC:0062.04.08.11.008 This field MUST advertise the length in bytes of the *Supported Door Lock Mode* field carried in the command.

Supported Door Lock Mode (M bytes)

This field is used to advertise the supported Door Lock Modes values. The length of this field in bytes CC:0062.04.08.11.009 MUST match the value advertised in the *Supported Door Lock Mode List* Length field.

CC:0062.04.08.11.00A Each byte MUST indicate the value of a supported Door Lock Mode that can be specified in a Door Lock Operation Set Command. Each byte MUST be encoded according to Table 2.198.

CC:0062.04.08.11.00B A sending node MUST advertise support for the Door Lock modes 0x00 and 0xFF.

Supported Outside Handle Modes Bitmask (4 bits)

This field is used to advertise which outside handles can be enabled and disabled using the Door Lock Configuration Set Command.

CC:0062.04.08.11.00C This field MUST be treated as a bitmask and represent outside door handles as described in Table 2.202.

CC:0062.04.08.11.00D If the corresponding handle can be enabled and disabled, the corresponding bit MUST be set to 1.

If the corresponding handle cannot be enabled or disabled, the corresponding bit MUST be set to 0.

Supported Inside Handle Modes Bitmask (4 bits)

This field is used to advertise which inside handles can be enabled and disabled using the Door Lock Configuration Set Command.

CC:0062.04.08.11.00E This field MUST be treated as a bitmask and represent inside door handles as described in Table 2.202.

CC:0062.04.08.11.00F If the corresponding handle can be enabled, the corresponding bit MUST be set to 1.

If the corresponding handle cannot be enabled, the corresponding bit MUST be set to 0.

Supported Door components (8 bits)

CC:0062.04.08.11.010 This field MUST advertise the supported door lock component that can be reported in the Door Lock condition field of the Door Lock Operation Report Command.

CC:0062.04.08.11.011 This field MUST be treated as a bitmask and represent inside door handles as described in Table 2.212.

Table 2.212: Door Lock Capabilities Report::Supported Door components encoding

Bit 7..3	Bit 2	Bit 1	Bit 0
Reserved	Latch	Bolt	Door

CC:0062.04.08.11.012 If the corresponding component is supported, the corresponding bit MUST be set to '1'.

If the corresponding component is not supported, the corresponding bit MUST be set to '0'.

ARS (Auto-Relock support) (1 bit)

This field is used to advertise if the sending node supports the auto-relock functionality.

CC:0062.04.08.11.013 The value 1 MUST indicate that the auto-relock functionality is supported.

The value 0 MUST indicate that the auto-relock functionality is not supported.

HRS (Hold and release support) (1 bit)

CC:0062.04.08.11.014 This field MUST advertise if the hold and release functionality is supported.

CC:0062.04.08.11.015 The value 0 MUST indicate that the hold and release functionality is not supported

The value 1 MUST indicate that the hold and release functionality is supported.

TAS (Twist assist support) (1 bit)

CC:0062.04.08.11.016 This field MUST advertise if the Twist Assist functionality is supported.

CC:0062.04.08.11.017 The value 0 MUST indicate that the twist assist functionality is not supported

The value 1 MUST indicate that the twist assist functionality is supported.

BTBS (Block-to-block support) (1 bit)

This field is used to advertise if the block-to-block functionality is supported.

CC:0062.04.08.11.018 The value 0 MUST indicate that the block-to-block functionality is not supported

The value 1 MUST indicate that the block-to-block functionality is supported.

2.2.40 Door Lock Logging Command Class, version 1

This Door Lock Logging Command Class provides an audit trail in an access control application. Each time an event takes place at the door lock, the system logs the user's ID, date, time etc.

2.2.40.1 Door Lock Logging Records Supported Get Command

This command is used to request the number of records that the audit trail supports.

The Door Lock Logging Records Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.213: Door Lock Logging Records Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING (0x4C)							
Command = DOOR_LOCK_LOGGING_RECORDS_SUPPORTED_GET (0x01)							

2.2.40.2 Door Lock Logging Records Supported Report Command

This command is used to report the maximum number of reports the audit trail supports.

Table 2.214: Door Lock Logging Records Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING (0x4C)							
Command = DOOR_LOCK_LOGGING_RECORDS_SUPPORTED_REPORT (0x02)							
Max records stored							

Max records stored (8 bits)

The number of records the audit trail supports stored in a queue.

2.2.40.3 Door Lock Logging Record Get Command

This command is used to request the audit trail.

The Door Lock Logging Record Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.215: Door Lock Logging Record Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING (0x4C)							
Command = RECORD_GET (0x03)							
Record number							

Record number (8 bits)

The Record number field indicates the record to be requested.

A value 0 - *Max records* stored is acceptable with a value of 0 being the most recent entry. When requesting with a value of 0, the report will contain the record number so the latest record is known.

2.2.40.4 Door Lock Logging Record Report Command

This command returns records from the audit trail.

To provide flexibility the user associated with the record may be identified by one of two methods: the user identifier or the user code.

Table 2.216: Door Lock Logging Record Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING (0x4C)							
Command = RECORD_REPORT (0x04)							
Record number							
Timestamp - Year 1							
Timestamp - Year 2							
Timestamp - Month							
Timestamp - Day							
Record status			Timestamp - Hour Local Time				
Timestamp - Minute Local Time							
Timestamp - Second Local Time							
Event Type							
User Identifier							
User Code Length							
USER_CODE 1							
...							
USER_CODE N							

Record number (8 bits)

Record number requested (1- 255).

Timestamp - Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Timestamp - Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Timestamp - Day (8 bits)

Specify the day of the month between 01 and 31.

Record status (3-bits)

The Record Status field is used to indicate if legal data is stored in the record.

Table 2.217: Record Status Field

Record State	Description
0	Requested record is empty.
1	Requested record holds legal data.

Timestamp - Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Timestamp - Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Timestamp - Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Note: If RTC (Real Time Clock) all values in Time Stamp record SHOULD be set to 0. Most recent Record MUST be stored under Record Number 1.

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. A User Identifier of 0 is acceptable when the record does not need to identify a user or if the User Code is provided in this report.

User Code Length (8 bits)

The User Code Length field indicates the number of bytes used to hold the User Code. A length of 0 is acceptable when the record does not need to identify a User Code or when the User Identifier field is non-zero.

USER_CODE (N bytes)

These fields contain the user code. Minimum code length is 4 and maximum 10 ASCII digits. For further details about the user code, refer to the User Code Command Class.

Event Type (8 bits)

This field MUST be encoded according to Table 2.218.

Table 2.218: Door Lock Logging Record Report::Event Type encoding

Event Type	Description
1	Lock Command: Keypad access code verified lock command
2	Unlock Command: Keypad access code verified unlock command
3	Lock Command: Keypad lock button pressed
4	Unlock command: Keypad unlock button pressed
5	Lock Command: Keypad access code out of schedule
6	Unlock Command: Keypad access code out of schedule
7	Keypad illegal access code entered
8	Key or latch operation locked (manual)
9	Key or latch operation unlocked (manual)
10	Auto lock operation
11	Auto unlock operation
12	Lock Command: Z-Wave access code verified
13	Unlock Command: Z-Wave access code verified
14	Lock Command: Z-Wave (no code)
15	Unlock Command: Z-Wave (no code)
16	Lock Command: Z-Wave access code out of schedule
17	Unlock Command Z-Wave access code out of schedule
18	Z-Wave illegal access code entered
19	Key or latch operation locked (manual)
20	Key or latch operation unlocked (manual)
21	Lock secured
22	Lock unsecured
23	User code added
24	User code deleted
25	All user codes deleted
26	Admin code changed
27	User code changed
28	Lock reset
29	Configuration changed

continues on next page

Table 2.218 – continued from previous page

Event Type	Description
30	Low battery
31	New Battery installed

Not all events types are supported and it is up to manufacturer to decide which ones are to be supported.

2.2.41 Energy Production Command Class, version 1 [NEVER CERTIFIED]

Warning: THIS COMMAND CLASS HAS NEVER BEEN CERTIFIED

This command class has never been implemented and certified by a Z-Wave product Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.

Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

The Energy Production Command Class is used to retrieve various production data from the device.

2.2.41.1 Energy Production Get Command

This command is used to request various production data from the device.

The Energy Production Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.219: Energy Production Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENERGY_PRODUCTION (0x90)							
Command = ENERGY_PRODUCTION_GET (0x02)							
Parameter Number							

Parameter Number (8 bits)

The parameter number specifies the kind of production data to retrieve. Currently the following parameter numbers are defined:

Table 2.220: Energy Production Parameter Number

Parameter Number	Description
0x00	Instant energy production
0x01	Total energy production
0x02	Energy production today
0x03	Total production time

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.41.2 Energy Production Report Command

This command is used to retrieve various production data from the device.

Table 2.221: Energy Production Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENERGY_PRODUCTION (0x90)							
Command = ENERGY_PRODUCTION_REPORT (0x03)							
Parameter Number							
Precision			Scale		Size		
Value 1							
...							
Value N							

Parameter Number (8 bits)

Refer to description under the Energy Production Get Command.

Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Scale (2 bits)

The Scale field indicates the scale used for the specified parameter number:

Table 2.222: Energy Production Scale

Parameter Number	Scale	Description
0x00	0x00	W
0x01	0x00	Wh
0x02	0x00	Wh
0x03	0x00	Seconds
0x03	0x01	Hours

Size (3 bits)

The size field indicates the number of bytes used for the value. This field can take values from 1 (001b), 2 (010b) or 4 (100b).

Value (N bytes)

The value is MUST be treated as a signed field (Table 2.12). The field MAY be 1, 2 or 4 bytes in size. The first byte is the most significant byte.

2.2.42 Entry Control Command Class, version 1

The Entry Control Command Class defines a method for advertising user input to a central Entry Control application and for the discovery of capabilities. User input may be button presses, RFID tags or other means.

It is RECOMMENDED that this command class is only supported via secure communication. This recommendation may be raised to a stronger requirement at the Device Class or Device Type level.

The Entry Control Command Class provides the following commands:

- Key Supported Get/Report - Used by the controller to request which ASCII keys are present on the Entry Control device.
- Event Supported Get/Report - Used by the controller to request which Event Types are supported by the Entry Control device.
- Configuration Set/Get/Report - Used by the controller to configure the Entry Control device
- Notification Command - Used by the Entry Control device to notify the controller of inputs.

2.2.42.1 Interoperability Considerations

It is RECOMMENDED that an Entry Control input device that implements the Entry Control Command Class also implements the Indicator Command Class, version 2.

If implementing the Indicator Command Class, version 2, it is RECOMMENDED that all indicator resources are addressable via the Indicator Command Class, version 2.

The input device MAY implement local control of indicator resources. If the Indicator Command Class, version 2 is supported, the first received Indicator Set command MUST disable all local control of indicator resources – except for interface feedback functionality like a short beep on each button press.

Transmissions may fail due to central control application fault or due to RF jamming. It is RECOMMENDED that an Entry Control input device provides local user feedback if the transmission of notifications to the central control application fails.

If the device only features light indicators, it is RECOMMENDED that all light flashes at 2Hz for at least 5 seconds to indicate transmission error.

If the device features a buzzer, it is RECOMMENDED that the buzzer generates sound pulses at a rate of 2Hz for at least 5 seconds to indicate transmission error.

If implementing the indicator Command Class, version 2, in devices based on Role Type RSEN (using Wake Up Command Class), the control of indicators must be synchronized with the Wake Up Notification. The controller must therefore wait for the Wake Up Notification, before it can control the indictors. The device implementing the indicator Command Class must therefore send the Wake Up Notification after sending the Entry Control Notification Command, in case the device expects to be controlled.

2.2.42.2 Security Considerations

An attacker could theoretically determine the length of manually entered user credentials even if they are transmitted via encryption. It is therefore MANDATORY to add padding bytes to ASCII strings so that all transferred ASCII strings are structured as one or more blocks of 16 characters. The ASCII code 0xFF MUST be reserved for padding purposes and it MUST NOT be used for any other purposes.

2.2.42.3 Handling user supplied data

The Entry Control device must cache the user input before sending the full entry in one Notification Frame. A user input MUST therefore have a termination, which MAY be determined by:

- The Key Cache Size is exceeded
- The Key Cache Timeout is exceeded
- A Command Button is pressed
- User data is received by other means, e.g. from an RFID tag

Key Cached Size:

The Key Cached Size is configured to specify the number of user inputs before the Notification Frame is sent. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

The Key Cached Size May be configured to 1, in which case a Notification Frame is send for each user input.

It is RECOMMENDED to have a default Key Cached Size of 4, in which case the Notification Frame will be send after 4 entries.

Key Timeout:

The Key Timeout is configured to specify the maximum time between user inputs. If the time between user inputs exceeds the Timeout, the cached user inputs will be send in a Notification Frame. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

Based on Command Button:

A user input may be terminated by the user pressing on of the Command Buttons like ENTER or ARM_ALL. The cached entry will be sent in a Notification Frame immediately after the user presses the Command Button. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

Based on RFID:

When presenting an RFID tag, the ID will be read from the tag, and this terminates the user input. So the cached ID ‘s from the RFID tag data can immediately be sent in a Notification Frame. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

Presenting an RFID tag may also terminate an ongoing user input. For instance, a user may enter four characters and subsequently present an RFID tag. In that case, the four entries must first be sent in one Notification Frame, and subsequently the ID ‘s from the RFID tag data is sent in a second Notification Frame.

2.2.42.4 Handling Incorrect Entry

A user may do an incorrect entry e.g. the credential is 1234 but the user enters 1734. In this case the terminal may provide a “delete” option, to allow the user to fix or re-enter the credential. If a “delete” option is not provided, the invalid code must first be send, followed by a new (correct) entry.

The terminal must not assume that the receiving controller does error handling like 1+7+Delete+2+3+4 = 1234.

2.2.42.5 Entry Control Notification Command

This command is used to advertise user input.

Depending on the Event Type, this command MAY carry manually entered user credentials.

Table 2.223: Entry Control Notification Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_NOTIFICATION (0x01)							
Sequence Number							
Reserved						Data Type	
Event Type							
Event Data Length							
Event Data 1 (optional)							
...							
Event Data N (optional)							

Sequence Number (8 bit)

The sequence number MUST be incremented each time a new command of this type is issued.

The value MUST be in the range 0..255. The initial value MAY be any value in the range 0..255.

A receiving device MUST use the Sequence Number to detect and ignore duplicates.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Data Type (2 bits)

This field is used to advertise the type of data (if any) that is appended to this command.

Table 2.224: Entry Control Notification::Event Data Type

Data Type		Description
0x00	NA	No data included
0x01	RAW	1 to 32 bytes of arbitrary binary data
0x02	ASCII	1 to 32 ASCII encoded characters. ASCII codes MUST be in the value range 0x00-0xF7. The string MUST be padded with the value 0xFF to fit 16 byte blocks when sent in a notification.
0x03	MD5	16 byte binary data encoded as a MD5 hash value.

All other values are reserved and MUST NOT be used by a sending device. Reserved values MUST be ignored by a receiving device.

Event Type (4 bit) This field is used to advertise the actual Event Type.

The field MUST be encoded according to Section 2.2.42.13.

Event Data Length (8 bit)

This field MUST advertise the length of the Event Data field in bytes.

The value MUST be in the range 0..32.

If no data bytes are included, this field MUST be set to 0.

Event Data (n bytes)

This field is used to carry data related to the event, e.g. received from an RFID tag.

The length of this field MUST comply with the length advertised by the Event Data Length field.

The format of this field MUST comply with the data format advertised by the Data Type field.

2.2.42.6 Entry Control Key Supported Get Command

This command is used to query the keys that a device implements for entry of user credentials.

The Entry Control Key Supported Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.225: Entry Control Key Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_KEY_SUPPORTED_GET (0x02)							

2.2.42.7 Entry Control Key Supported Report Command

This command is used to advertise the keys that a device implements for entry of user credentials.

A management interface may determine the available keys for credential entry from this command.

The range of available Command Keys may be determined via the Entry Control Event Supported Report Command.

Table 2.226: Entry Control Key Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_KEY_SUPPORTED_REPORT (0x03)							
Key Supported Bit Mask Length							
Key Supported Bit Mask 1							
...							
Key Supported Bit Mask N							

Key Supported Bit Mask Length

This field is used to advertise the number of Key Supported Bit Mask Bytes to follow.

Only the Key Supported Bit Mask bytes until the last supported ASCII key SHOULD be included.

Key Supported Bit Mask (Variable length)

This field is used to advertise the keys that a device implements for entry of user credentials.

The Key Supported Bit Mask field MUST advertise ASCII codes that represent the supported keys.

- Bit 0 in Bit Mask 1 indicates that the supporting device may issue ASCII code 0
- Bit 1 in Bit Mask 1 indicates that the supporting device may issue ASCII code 1
- ...
- Bit 7 in Bit Mask 16 indicates that the supporting device may issue ASCII code 127

2.2.42.8 Entry Control Event Supported Get Command

This command is used to request the supported Events of a device.

The Entry Control Event Supported Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.227: Entry Control Event Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_EVENT_SUPPORTED_GET (0x04)							

2.2.42.9 Entry Control Event Supported Report Command

Table 2.228: Entry Control Event Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_EVENT_SUPPORTED_REPORT (0x05)							
Reserved						Data Type Supported Bit Mask Length	
Data Type Supported Bit Mask 1							
...							
Data Type Supported Bit Mask M							
Reserved			Event Supported Bit Mask Length				
Event Type Supported Bit Mask 1							
...							
Event Type Supported Bit Mask N							
Key Cached Size supported Minimum							
Key Cached Size supported Maximum							
Key Cached Timeout supported Minimum							
Key Cached Timeout supported Maximum							

Data Type Supported Bit Mask Length (2 bits)

This field is used to advertise the number of Data Type Supported Bit Mask Bytes to follow.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Data Type Supported Bit Mask (Variable length)

This field is used to advertise the supported Data Types.

- Bit 0 in Bit Mask 1 indicates if Data Type 0 is supported
- Bit 1 in Bit Mask 1 indicates if Data Type 1 is supported
- ...

For the definition of Data Type IDs, refer to Table 2.224.

Event Type Supported Bit Mask Length (1 byte)

This field is used to advertise the number of Event Type Supported Bit Mask Bytes to follow.

Event Type Supported Bit Mask (variable length)

This field is used to advertise the supported Event Type.

- Bit 0 in Bit Mask 1 indicates if Event Type 0 is supported
- Bit 1 in Bit Mask 1 indicates if Event Type 1 is supported
- ...

For the definition of Event Type IDs, refer to Table 2.232.

Key Cached Size Supported Minimum

The minimum configurable number of key entries before the CACHED_KEYS notification is sent.

Refer to Section 2.2.42.10.

Key Cached Size Supported Maximum

The maximum configurable number of key entries before the CACHED_KEYS notification is sent.

Refer to Section 2.2.42.10.

Key Cached Timeout Supported Minimum

The minimum configurable timeout before the CACHED_KEYS notification is sent.

Refer to Section 2.2.42.10.

Key Cached Timeout Supported Maximum

The maximum configurable timeout before the CACHED_KEYS notification is sent.

Refer to Section 2.2.42.10.

2.2.42.10 Entry Control Configuration Set Command

This command is used to configure Event Type specific parameters.

Table 2.229: Entry Control Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_CONFIGURATION_SET (0x06)							
Key Cache Size							
Key Cache Timeout							

Key Cache Size (1 byte) This field specifies the number of characters the key cache MUST store before sending data to the central control application. Data MUST be sent when the number of characters in the cache matches or exceeds the value of this field.

The value MUST be in the range 1..32. The default value of this field SHOULD be 4.

In deployments where characters are entered first and a command button is pressed subsequently, it is RECOMMENDED that this field is set to 32 and that the Key Cache Timeout is set to 2 seconds.

Key Cache Timeout (1 byte)

This field specifies the number of seconds the key cache MUST wait for additional characters before sending data to the central control application. Data MUST be sent if a Key Cache Timeout occurs.

The Key Cache Timeout MUST be measured from the most recent reception of a character.

The value SHOULD be in the range 1..10 seconds. The default value SHOULD be 2 seconds.

2.2.42.11 Entry Control Configuration Get Command

This command is used to request the operational mode of a device.

The Entry Control Configuration Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.230: Entry Control Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_CONFIGURATION_GET (0x07)							

2.2.42.12 Entry Control Configuration Report Command

This command is used to advertise the current operational mode of a device.

Table 2.231: Entry Control Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL (0x6F)							
Command = ENTRY_CONTROL_CONFIGURATION_REPORT (0x08)							
Key Cache Size							
Key Cache Timeout							

Key Cache Size (1 byte)

Refer to Section 2.2.42.10 Entry Control Configuration Set Command.

Key Cache Timeout (1 byte)

Refer to Section 2.2.42.10 Entry Control Configuration Set Command.

2.2.42.13 Event Types

The Event Types MUST be encoded according to Table 2.232.

Table 2.232: Event Types

Event Type		Recommended Button Label	Event Data (optional)
0x00	CACHING	-	-
0x01	CACHED_KEYS	-	ASCII bytes
0x02	ENTER	Enter	ASCII bytes
0x03	DISARM_ALL	Disarm	ASCII bytes
0x04	ARM_ALL	Arm	ASCII bytes
0x05	ARM_AWAY	Away	ASCII bytes
0x06	ARM_HOME	Home	ASCII bytes
0x07	EXIT_DELAY	Arm_Delay	ASCII bytes
0x08	ARM_1	Arm zone 1	ASCII bytes
0x09	ARM_2	Arm zone 2	ASCII bytes
0x0A	ARM_3	Arm zone 3	ASCII bytes
0x0B	ARM_4	Arm zone 4	ASCII bytes
0x0C	ARM_5	Arm zone 5	ASCII bytes
0x0D	ARM_6	Arm zone 6	ASCII bytes
0x0E	RFID	-	As advertised by the Data Type field
0x0F	BELL	-	-
0x10	FIRE	Fire	-
0x11	POLICE	Police	-
0x12	ALERT_PANIC	-	-
0x13	ALERT_MEDICAL	-	-
0x14	GATE_OPEN	Open, Up. 'O' or similar	ASCII bytes
0x15	GATE_CLOSE	Close, Down, 'C' or similar	ASCII bytes
0x16	LOCK	-	ASCII bytes
0x17	UNLOCK	-	ASCII bytes
0x18	TEST	Test	ASCII bytes
0x19	CANCEL	Cancel	ASCII bytes

All other values are reserved and MUST NOT be used by a sending device. Reserved values MUST be ignored by a receiving device.

Event Type CACHING

The CACHING Event Type is used to indicate to the central controller that the user has started entering credentials, and that caching is initiated. This allows the central controller to change the indications on the Entry Control device through the indicator Command Class, or to change the status of the central controller user interface.

If Key Cached Size it set to 1, the CACHING event notification MUST NOT be sent. Instead each individual credential byte MUST be sent in its own CACHED_KEYS event notification.

Event Type CACHED_KEYS

The CACHED_KEYS Event Type is used to send user inputs in a Notification Frame. The CACHED_KEYS is sent when the user input is terminated by one of the following reasons:

- The Key Cache Size is exceeded
- The Key Cache Timeout is exceeded
- A command button is pressed
- User data is received by other means, e.g. from an RFID tag

2.2.43 Generic Schedule Command Class, version 1 [NEVER CERTIFIED]

Warning: THIS COMMAND CLASS HAS NEVER BEEN CERTIFIED

This command class has never been implemented and certified by a Z-Wave product Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.

Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

The Generic Schedule Command Class is used to define schedules.

Schedules defined in this command class are essentially time ranges or intervals. Each schedule is associated to a Schedule ID.

The Schedules defined in this Command Class can be used in other Command Classes, e.g. to schedule when user credentials are valid or when to apply a particular configuration.

2.2.43.1 Terminology

Schedules are composed by one or more **Time Range(s)**. A time range is a composed of one or several **scheduling parameters**, such as weekdays, day of the month, hour of the day, etc.

Time ranges are defined individually with scheduling parameters. For example, it could be Time Range 1 representing weekdays during work hours and Time Range 2 representing public holidays at fixed dates. (e.g. 25th of December and 31st of October)

A schedule can then be set **including** or **excluding** each of its composing time ranges. Schedule ID 1 could include weekdays work hours (Time Range 1) and exclude public holidays (Time Range 2). It means that Schedule 1 must be active only on weekdays work hours when it is not a public holiday.

At least one of the included Time Ranges conditions must be fulfilled and none of the excluded Time Ranges must be fulfilled for a Schedule to be active.

2.2.43.2 Interoperability considerations

This Command Class relies on time and date synchronization between controlling and supporting nodes.

Nodes supporting this command class should also have correct time or date settings depending on their capabilities. Supporting nodes may support the Clock Command Class for this purpose. The Time Command Class may also be used for reading the current date and time at the supporting node.

2.2.43.3 Generic Schedule Capabilities Get Command

This command is used to request the scheduling capabilities of a supporting node.

The Generic Schedule Capabilities Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.233: Generic Schedule Capabilities Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_CAPABILITIES_GET (0x01)							

2.2.43.4 Generic Schedule Capabilities Report Command

This command is used to advertise the number of supported Schedule Slot IDs for each Schedule Type/User Identifier by the sending node.

Table 2.234: Generic Schedule Capabilities Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_CAPABILITIES_REPORT (0x02)							
Number of supported Schedule IDs (MSB)							
Number of supported Schedule IDs (LSB)							
Res	Number of supported Time Range IDs (MSB)						
Number of supported Time Range IDs (LSB)							
Number of supported Time Ranges per Schedule							
Reserved					Week-days	Date	Hour & Minute

Number of supported Schedule IDs (16 bits)

This field is used to indicate how many Schedule IDs are supported by a supporting node.

This field MUST be in the range 1..65535.

Number of supported Time Range IDs (15 bits)

This field is used to indicate how many Time Ranges IDs are supported by a supporting node.

This field MUST be in the range 1.. 32767.

The number of supported Time Ranges IDs should be equal or greater than the (*Number of supported Schedule IDs*) x (*Number of supported Time Ranges per Schedule*)

Number of supported Time Ranges per Schedule (8 bits)

This field is used to indicate how many time ranges can be used as part of 1 schedule.

This field MUST be in the range 1..255.

Weekdays (1 bit)

This field is used to indicate if the supporting node can accept weekdays in the time ranges.

The value 0 MUST indicate that weekdays are not supported in time ranges and will be ignored.

The value 1 MUST indicate that weekdays are supported in time ranges.

Date (1 bit)

This field is used to indicate if the supporting node can accept years, month and day parameters in the time ranges.

The value 0 MUST indicate that start/stop dates are not supported in time ranges and will be ignored.

The value 1 MUST indicate that start/stop dates are supported in time ranges.

Hour & Minute (1 bit)

This field is used to indicate if the supporting node can accept start and stop hours/minutes in the time ranges.

The value 0 MUST indicate that start/stop hours/minutes are not supported in time ranges and will be ignored.

The value 1 MUST indicate that start/stop hours/minutes are supported in time ranges.

A supporting node MUST advertise support for at least 1 scheduling parameter (Weekdays, Date, Hour & Minute).

2.2.43.5 Generic Schedule Time Range Set Command

This command is used to configure a Time Range for a supporting node.

A Time Range MUST be active when all the conditions indicated in its configuration are fulfilled. Some examples are provided in Section 2.2.43.5.1.

Table 2.235: Generic Schedule Time Range Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_TIME_RANGE_SET (0x03)							
Res	Time Range ID 1 (MSB)						
Time Range ID 2 (LSB)							
In use	Weekday bitmask						
In use	Start Year (MSB)						
Start Year (LSB)							
In use	Stop Year (MSB)						
Stop Year (LSB)							
In use	Reserved			Start Month			
In use	Reserved			Stop Month			
In use	Reserved		Start Day				
In use	Reserved		Stop Day				
In use	Reserved		Start Hour				
In use	Reserved		Stop Hour				
In use	Res	Start Minute					
In use	Res	Stop Minute					
In use	Reserved		Daily Start Hour				
In use	Reserved		Daily Stop Hour				
In use	Res	Daily Start Minute					
In use	Res	Daily Stop Minute					

Res / Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Time Range ID (15 bits)

This field is used to indicate which Time Range ID to use for the Time Range being set.

This field MUST be in the range 0..[Number of supported Time Range IDs].

The value 0 MUST indicate that the supporting node MUST erase all configured Time Ranges. In this case, all other fields MUST be set to 0 and ignored by a receiving node.

Values in the range 1..[Number of supported Time Range IDs] MUST indicate the actual Time Range ID to configure.

If a non-supported Time Range ID is specified in this command, a receiving node MUST ignore the command.

In use (15 x 1 bit)

This field is used to indicate if the corresponding scheduling parameter must be used in the Time Range being set.

This field MUST apply to the scheduling parameter being part of the same byte. E.g. the "in use" field part of the byte comprising the *Weekday bitmask* field MUST indicate if the *Weekday bitmask* parameter MUST be used and applied to the current Time Range.

The value 0 MUST indicate that the corresponding field is set to 0 and MUST be ignored.

The value 1 MUST indicate that the corresponding field is set to a valid value and MUST be applied for the actual Time Range ID.

All "in use" fields set to 0 MUST indicate to erase a Time Range configuration.

Weekday bitmask (7 bits)

This field is used to indicate on which days the actual Time Range ID MUST be active.

This field MUST be treated as a bitmask and encoded according to Table 2.236.

Table 2.236: Generic Schedule Set::Weekday bitmask encoding

Bit #	Weekday
Bit 0	Sunday
Bit 1	Monday
Bit 2	Tuesday
Bit 3	Wednesday
Bit 4	Thursday
Bit 5	Friday
Bit 6	Saturday

If the corresponding *In use* field is set to 1:

- The value 0 MUST indicate that the Time Range MUST NOT be active during the corresponding weekday.
- The value 1 MUST indicate that the Time Range MUST be active during the corresponding weekday.

If the corresponding *In use* field is set to 0, the Time Range MUST be active regardless of the current weekday.

A supporting node MUST ignore this field if it advertises no support for the weekday scheduling parameter in the Generic Schedule Capabilities Report Command

Start Year (15 bits)

This field is used to indicate the start year for the actual Time Range ID.

If the corresponding *In use* field is set to 1, the fields MUST indicate that the Time Range ID MUST be use this field as parameter. Refer to Section 2.2.43.5.1 for details. This field MUST be encoded using unsigned representation.

If the corresponding *In use* field is set to 0, the Time Range MUST ignore this parameter.

A supporting node MUST ignore this field if it advertises no support for the Date scheduling parameters in the Generic Schedule Capabilities Report Command

Stop Year (15 bits)

This field is used to indicate the stop year for the actual Time Range ID.

If the corresponding *In use* field is set to 1, the fields MUST indicate that the Time Range ID MUST be use this field as parameter. Refer to Section 2.2.43.5.1 for details. This field MUST be encoded using unsigned representation.

If the corresponding *In use* field is set to 0, the Time Range MUST ignore this parameter.

A supporting node MUST ignore this field if it advertises no support for the Date scheduling parameters in the Generic Schedule Capabilities Report Command

Start Month (4 bits)

This field is used to indicate the start month for the actual Time Range ID.

If the corresponding *In use* field is set to 1, the fields MUST indicate that the Time Range ID MUST be use this field as parameter. Refer to Section 2.2.43.5.1 for details. The values 1..12 MUST represent respectively January..December.

If the corresponding *In use* field is set to 0, the Time Range MUST ignore this parameter.

A supporting node MUST ignore this field if it advertises no support for the Date scheduling parameters in the Generic Schedule Capabilities Report Command

Stop Month (4 bits)

This field is used to indicate the stop month for the actual Time Range ID.

If the corresponding *In use* field is set to 1, the fields MUST indicate that the Time Range ID MUST be use this field as parameter. Refer to Section 2.2.43.5.1 for details. The values 1..12 MUST represent respectively January..December.

If the corresponding *In use* field is set to 0, the Time Range MUST ignore this parameter.

A supporting node MUST ignore this field if it advertises no support for the Date scheduling parameters in the Generic Schedule Capabilities Report Command

Start Day (5 bits)

This field is used to indicate the start day of the month for the actual Time Range ID.

If the corresponding *In use* field is set to 1, the fields MUST indicate that the Time Range ID MUST be use this field as parameter. Refer to Section 2.2.43.5.1 for details. The field MUST be in the range 1..31.

If the corresponding *In use* field is set to 0, the Time Range MUST ignore this parameter.

A supporting node MUST ignore this field if it advertises no support for the Date scheduling parameters in the Generic Schedule Capabilities Report Command

Stop Day (5 bits)

This field is used to indicate the stop day of the month for the actual Time Range ID.

If the corresponding *In use* field is set to 1, the fields MUST indicate that the Time Range ID MUST be use this field as parameter. Refer to Section 2.2.43.5.1 for details. The field MUST be in the range 1..31.

If the corresponding *In use* field is set to 0, the Time Range MUST ignore this parameter.

A supporting node MUST ignore this field if it advertises no support for the Date scheduling parameters in the Generic Schedule Capabilities Report Command

Start Hour /Stop Hour (5 bits) & Start Minute / Stop Minute (6 bits)

These fields are used to indicate from and until which times the actual Time Range ID MUST be active.

If the corresponding *In use* field is set to 1:

- The *Start Hour / Stop Hour* fields MUST be in the range 0..23.
- The *Start Minute / Stop Minute* fields MUST be in the range 0..59.

If the corresponding In use field is set to 0, the corresponding field MUST be ignored by the Time Range.

A supporting node MUST ignore this field if it advertises no support for the Hour & Minute scheduling parameter in the Generic Schedule Capabilities Report Command

Daily Start Hour /Stop Hour (5 bits) & Daily Start Minute / Stop Minute (6 bits)

These fields are used to indicate in which time range the actual Time Range ID MUST be active.

If the corresponding *In use* field is set to 1:

- The fields MUST indicate that the Time Range ID MUST be active daily between the 2 specified hour and minutes times. (e.g. from the 08:00 to 16:30)
- The Start Hour / Stop Hour fields MUST be in the range 0..23.
- The Start Minute / Stop Minute fields MUST be in the range 0..59.

If the corresponding *In use* field is set to 0, the corresponding field MUST be ignored by the Time Range.

A supporting node MUST ignore this field if it advertises no support for the Hour & Minute scheduling parameter in the Generic Schedule Capabilities Report Command

2.2.43.5.1 Time Range parameters and rules

Time Ranges MUST be active when all their scheduling parameters are matching the conditions. We define the parameter groups shown in Table 2.237.

Table 2.237: Generic Schedule Set Command::Parameter groups

Parameter Group	Fields
Daily Start time	Daily Start Hour Daily Start Minute
Daily Stop time	Daily Stop Hour Daily Stop Minute
Start Datetime	Start Year Start Month Start Day Start Hour Start Minute
Stop Datetime	Stop Year Stop Month Stop Day Stop Hour Stop Minute
Weekday	Weekday bitmask

2.2.43.5.2 All parameters specified

If all Scheduling parameters are specified in a TimeRange, the Time Range MUST be active when the following condition is fulfilled:

- Daily Start Time <= Current time < Daily Stop Time
- Start Datetime <= Current datetime <= Stop Datetime
- Current weekday in (weekday bitmask)

For example, if Daily Start Time is set to 08:30, Daily Stop Time is set to 17:00, Start Datetime is set to 10th of January 2019 at 12:00, Stop Datetime is set to 5th of February 2019 at 12:00, Weekday bitmask is set to Monday to Friday, then the Time Range will be active:

- On the 10th of January 2019 from 12:00 to 17:00
- On subsequent weekdays until the 4th of February 2019, between 08:30 and 17:00
- On the 5th of February 2019 from 08:30 to 12:00

2.2.43.5.3 Some parameters undefined in parameter groups

If some of the parameters within a group are undefined, they are to be interpreted as wildcard and excluded from the comparison for the parameter group. For example, if Start Datetime has no Start Year and Start Month defined, but only Start Day set to 15, the time range MUST start on the 15th of every month also regardless of the year.

If Start Day is set to 15 and Stop Minute is set to 30, then the Time Range MUST be active every hour during 30 minutes from the 15th until the end of each month.

2.2.43.5.4 All parameters missing from a parameter group

If all fields are undefined (not in use) for a parameter group, the corresponding parameter group MUST NOT be used for the Time Range activation. For example, if none of the Stop Datetime parameters are defined, then the Time Range MUST be active when:

- Daily Start Time <= Current time < Daily Stop Time
- Start Datetime <= Current datetime
- Current weekday in (weekday bitmask)

2.2.43.6 Generic Schedule Time Range Get Command

This command is used to request the configuration of a Time Range ID for a supporting node.

The Generic Schedule Time Range Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.238: Generic Schedule Time Range Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_TIME_RANGE_GET (0x04)							
Res	Time Range ID 1 (MSB)						
Time Range ID 2 (LSB)							

Time Range ID (16 bits)

This field is used to specify the requested Time Range ID.

The first byte MUST carry the most significant byte of the 15 bits.

A supporting node MUST return this value in the *Time Range ID* field in the returned Generic Schedule Report Command.

2.2.43.7 Generic Schedule Time Range Report Command

This command is used to advertise the configuration of a Time Range ID for a supporting node.

Table 2.239: Generic Schedule Time Range Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_TIME_RANGE_REPORT (0x05)							
Res	Time Range ID 1 (MSB)						
Time Range ID 2 (LSB)							
In use	Weekday bitmask						
In use	Start Year (MSB)						
Start Year (LSB)							
In use	Stop Year (MSB)						
Stop Year (LSB)							
In use	Reserved			Start Month			
In use	Reserved			Stop Month			
In use	Reserved		Start Day				
In use	Reserved		Stop Day				
In use	Reserved		Start Hour				
In use	Reserved		Stop Hour				
In use	Res	Start Minute					
In use	Res	Stop Minute					
In use	Reserved		Daily Start Hour				
In use	Reserved		Daily Stop Hour				
In use	Res	Daily Start Minute					
In use	Res	Daily Stop Minute					
Res	Next Time Range ID						
Next Time Range ID							

Fields not described below MUST remain identical to the description in Section 2.2.43.5 Generic Schedule Time Range Set Command.

Res / Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Time Range ID (15 bits)

This field is used to indicate which Time Range ID configuration is being advertised.

If this field is set to a non-supported or non-configured Time Range ID, all scheduling parameters fields MUST be set to 0.

Next Time Range ID (15 bits)

This field is used to indicate the next non-empty Time Range ID.

This field MUST indicate the next non-empty Time Range ID. The value 0 MUST indicate that there is no following Time Range ID configured at the receiving node.

Non-empty Time Range slot ID means that at least 1 *In use* flag is set to 1.

2.2.43.8 Generic Schedule Set Command

This command is used to configure a Schedule for a supporting node.

A schedule MUST be active when:

- At least one of the included Time Ranges is active
- None of the excluded Time Ranges are active.

Refer to Section 2.2.43.8.1 for details.

Table 2.240: Generic Schedule Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_SET (0x06)							
Schedule ID 1 (MSB)							
Schedule ID 2 (LSB)							
Number of Time Range IDs							
Include/ exclude	Time Range ID 1 (MSB)						
Time Range ID 1 (LSB)							
...							
Include/ exclude	Time Range ID N (MSB)						
Time Range ID N (LSB)							

Res / Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Schedule ID (16 bits)

This field is used to indicate which Schedule slot ID to use for the schedule being set.

This field MUST be in the range 0..[Number of supported Schedule IDs].

The value 0 MUST indicate that the supporting node MUST erase all configured schedules. In this case, the *Number of Time Range IDs* field MUST be set to 0 by a sending node.

Values in the range 1..[Number of supported Schedule IDs] MUST indicate the actual Schedule ID to configure.

If a non-supported Schedule ID is specified in this command, a receiving node MUST ignore the command.

Number of Time Range IDs (8 bits)

This field is used to indicate the number of Time Range IDs included as part of the Schedule.

This field MUST be in the range 0..[Number of supported Time Range IDs per Schedule].

A supporting node MUST ignore this command if it is set to a value higher than the reported *Number of supported Time Ranges per Schedule* field in the Generic Schedule Capabilities Report Command

The number of *Include/exclude* and *Time Range ID* field blocks in this command MUST be according to this field.

Include/exclude (N x 1 bit)

This field is used to indicate if the Schedule ID must include or exclude the corresponding Time Range ID.

The value 0 MUST indicate that the corresponding Time Range is excluded and the schedule MUST be active outside of the Time Range.

The value 1 MUST indicate that the corresponding Time Range is included and the schedule MUST be active during the defined Time Range.

Time Range ID (N x 15 bits)

This field is used to indicate which Time Range ID MUST be part of the Schedule ID.

Undefined Time Range IDs (all zeroed out) may be part of Schedules.

2.2.43.8.1 Schedules and Time Ranges examples

2.2.43.8.2 Example 1

- Time Range 1 is from 24th to 26th of December
- Time Range 2 is from 17:00 to 08:00 every day
- Time Range 3 is weekend days.

If a schedule is set to:

- Include Time Range 1
- Exclude Time Range 2
- Include Time Range 3

Then it MUST be active:

- every week-end from 08:00 to 17:00
- on the 24th,25th and 26th of December from 08:00 to 17:00 regardless of the current weekday.

2.2.43.8.3 Example 2

- Time Range 3 is undefined (all unused value).

If a schedule is set to:

- Include Time Range 1
- Include Time Range 2
- Include Time Range 3

Then the schedule MUST be active at all times.

2.2.43.8.4 Example 3

- Time Range 3 is undefined (all unused value).

If a schedule is set to:

- Include Time Range 1
- Exclude Time Range 2
- Exclude Time Range 3

Then the schedule MUST never be active.

2.2.43.8.5 Example 4

- Time Range 1 has no start time and finishes on the 30th of June
- Time Range 2 is from 08:00 to 17:00 on week days
- Time Range 3 is 10:00 to 15:00 on weekend days

If a schedule is set to:

- Include Time Range 1
- Include Time Range 2
- Include Time Range 3

Then the schedule MUST be active:

- All the time from the 1st of January until the 30th of June
- From 08:00 to 17:00 on weekdays between the 1st of July and the 31st of December every year
- From 10:00 to 15:00 on weekends between the 1st of July and the 31st of December every year

2.2.43.9 Generic Schedule Get Command

This command is used to request the configuration of a Schedule ID for a supporting node.

The Generic Schedule Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.241: Generic Schedule Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_GET (0x07)							
Schedule ID 1 (MSB)							
Schedule ID 2 (LSB)							

Schedule ID (16 bits)

This field is used to specify the requested Schedule ID.

The first byte MUST carry the most significant byte of the 16 bits.

A supporting node MUST return this value in the *Schedule ID* field in the returned Generic Schedule Report Command.

2.2.43.10 Generic Schedule Report Command

This command is used to advertise the contents of a Schedule for a supporting node.

Table 2.242: Generic Schedule Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GENERIC_SCHEDULE (0xA3)							
Command = GENERIC_SCHEDULE_REPORT (0x08)							
Schedule ID 1 (MSB)							
Schedule ID 2 (LSB)							
Number of Time Range IDs							
Include/ exclude	Time Range ID 1 (MSB)						
Time Range ID 1 (LSB)							
...							
Include/ exclude	Time Range ID N (MSB)						
Time Range ID N (LSB)							
Next Schedule ID 1 (MSB)							
Next Schedule ID 2 (LSB)							

Fields not described below MUST remain identical to the description in Section 2.2.43.10.

Res / Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Schedule ID (16 bits)

This field is used to indicate which Schedule ID configuration is being advertised.

If this field is set to a non-supported or non-configured Schedule ID, all the *Number of Time Range IDs* field MUST be set to 0.

Number of Time Range IDs (8 bits)

This field is used to indicate the number of Time Range IDs included as part of the Schedule.

This field MUST be in the range 0..[Number of supported Time Range IDs per Schedule].

The number of *Include/exclude* and *Time Range ID* field blocks in this command MUST be according to this field.

Include/exclude (N x 1 bit)

This field is used to indicate if the Schedule ID must include or exclude the corresponding Time Range ID.

The value 0 MUST indicate that the corresponding Time Range is excluded and the schedule MUST be active outside of the Time Range.

The value 1 MUST indicate that the corresponding Time Range is included and the schedule MUST be active during the defined Time Range.

Time Range ID (N x 15 bits)

This field is used to indicate which Time Range ID MUST be part of the Schedule ID.

Undefined Time Range IDs (all zeroed out) may be part of Schedules.

Next Schedule ID (16 bits)

This field is used to indicate the next non-empty Schedule ID.

This field MUST indicate the next non-empty Schedule slot ID. The value 0 MUST indicate that there is no following Schedule Slot ID configured at the supporting node.

2.2.44 Geographic LocationCommand Class, version 1

The Geographic Location Command Class is used to read latitude and longitude from another device. The latitude and longitude may also be set according to the geographic location in question. Date and geographic location may be used to calculate sunrise and sunset for e.g. automatic lighting control.

2.2.44.1 Multi Channel Considerations

Multi Channel End Points SHOULD NOT support the Geographic Location Command Class.

2.2.44.2 Geographic Location Set Command

This command is used to set latitude and longitude.

Table 2.243: Geographic Location Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GEOGRAPHIC_LOCATION (0x8C)							
Command = GEOGRAPHIC_LOCATION_SET (0x01)							
Longitude Degrees							
Long. Sign	Longitude Minutes						
Latitude Degrees							
Lat. Sign	Latitude Minutes						

Longitude (16 bits)

The longitude determines one's location on the earth's surface, East or West of the Greenwich Meridian. The Greenwich Meridian is located at the Greenwich observatory, in Greenwich, England to be the geographic point for where East and West meet. Therefore, Greenwich Meridian is indicated as 0° longitude. Longitude values for points East of the Meridian are always positive, while points West of the Meridian are always negative. Valid ranges are for degrees (from -180 to 180) and minutes (0-59). Other values will be interpreted as 0.

Latitude (16 bits)

The latitude determines one's location on the earth's surface, North or South of the Equator. Latitude is measured between -90° South, and +90° North of the Equator point (0°). Valid ranges are for degrees (from -90 to 90) and minutes (0-59). Other values will be interpreted as 0.

2.2.44.3 Geographic Location Get Command

This command is used to request latitude and longitude from a device.

The Geographic Location Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.244: Geographic Location Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GEOGRAPHIC_LOCATION (0x8C)							
Command = GEOGRAPHIC_LOCATION_GET (0x02)							

2.2.44.4 Geographic Location Report Command

This command returns latitude and longitude from a device in a Z-Wave network.

Table 2.245: Geographic Location Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GEOGRAPHIC_LOCATION (0x8C)							
Command = GEOGRAPHIC_LOCATION_REPORT (0x03)							
Longitude Degrees							
Long. Sign	Longitude Minutes						
Latitude Degrees							
Lat. Sign	Latitude Minutes						

Refer to description under the *Geographic Location Set Command*.

2.2.45 HRV Status Command Class, version 1

The residential Heat Recovery Ventilation (HRV) Status Command Class is used to read out a number of parameters in the ventilation system.

2.2.45.1 HRV Status Get Command

This command is used to request specific parameters from the ventilation system.

The HRV Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.246: HRV Status Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS (0x37)							
Command = HRV_STATUS_GET (0x01)							
Status Parameter							

Status Parameter (8 bits)

The status parameter used to indicate which status parameter that is requested.

Table 2.247: HRV Status Parameter

Value	Status Parameter
0x00	Outdoor Air temperature
0x01	Supply Air (to room) temperature
0x02	Exhaust Air (from room) temperature
0x03	Discharge Air temperature
0x04	Room temperature
0x05	Relative Humidity in room
0x06	Remaining filter life

2.2.45.2 HRV Status Report Command

This command is used to report a specific status parameter in response to a HRV Status Get.

Table 2.248: HRV Status Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS (0x37)							
Command = HRV_STATUS_REPORT (0x02)							
Status Parameter							
Precision			Scale		Size		
Value 1							
...							
Value N							

Status Parameter (8 bits)

The status parameter used to indicate which status parameter that is reported. Refer to Section 2.2.45.1 HRV Status Get for possible values.

Precision (3 bits)

The precision field describes what the precision of the setpoint value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Scale (2 bits)

The scale field indicates the scale used the list of possible scales are given below:

Table 2.249: HRV Scale Parameter

Status Parameter	Scale	Value
Outdoor Air temperature	Celsius (C)	0
	Fahrenheit (F)	1
Supply Air (to room) temperature	Celsius (C)	0
	Fahrenheit (F)	1
Exhaust Air (from room) temperature	Celsius (C)	0
	Fahrenheit (F)	1
Discharge Air temperature	Celsius (C)	0
	Fahrenheit (F)	1
Room temperature	Celsius (C)	0
	Fahrenheit (F)	1
Relative Humidity in room	Percentage (%)	0
Remaining filter life	Percentage (%)	0

Size (3 bits)

The size field indicates the number of bytes used for the sensor value. This field can take values from 1 (001b), 2 (010b) or 4 (100b).

Value (N bytes)

The value MUST be treated as a signed field (Table 2.12). The value MAY be 1, 2 or 4 bytes in size. This first byte is the most significant byte.

2.2.45.3 HRV Status Supported Get Command

This command is used to request a bitmap of the supported status parameters.

The HRV Status Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.250: HRV Status Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS (0x37)							
Command = HRV_STATUS_SUPPORTED_GET (0x03)							

2.2.45.4 HRV Status Supported Report Command

This command is used to report a bitmap indicating the supported status parameters.

Table 2.251: HRV Status Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS (0x37)							
Command = HRV_STATUS_SUPPORTED_REPORT (0x04)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields describe the supported status parameters from the ventilation system. Bit 0 in the Bit Mask 1 field used to indicate if the status parameter "Outdoor Air Temperature" is supported, 0 indicating not supported and 1 indicating supported. Bit 1 in the Bit Mask 1 field used to indicate if the status parameter "Supply Air Temperature" is supported and so forth. All available status parameters are given in Section 2.2.45.1 HRV Status Get.

It is only necessary to send the Bit Mask fields 1 and up to the one indicating the last support status parameter. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

2.2.46 HRV Control Command Class, version 1

The Heat Recovery Ventilation (HRV) Control Command Class is introducing control of Heat Recovery Ventilation systems via the Z-Wave interface.

2.2.46.1 HRV Mode Set Command

This command is used to set the desired mode in the device.

Table 2.252: HRV Mode Set

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_MODE_SET (0x01)							
Reserved			Mode				

Mode (5 bits)

The mode identifier MAY be set to the following values:

Table 2.253: HRV Mode

Mode	Name	Description
0	Off	The HRV system is in the off state, frost protection can occur.
1	Demand / Automatic	The HRV system is controlled based on sensor input.
2	Schedule	The HRV system is controlled based on predefined input from the factory and/or user/installer.
3	Energy Savings Mode	The HRV system will be put into a reduced heat / ventilation mode.
4	Manual	The HRV system is in manual mode. The command HRV Manual Control Set may be used to manually control the device.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.46.2 HRV Mode Get Command

This command is used to request the current mode from the device.

The HRV Mode Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.254: HRV Mode Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_MODE_GET (0x02)							

2.2.46.3 HRV Mode Report Command

The HRV Mode Report Command is used to report the mode from the device.

Table 2.255: HRV Mode Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_MODE_REPORT (0x03)							
Reserved				Mode			

Mode (8 bits)

Refer to description under the Section 2.2.46.1 HRV Mode Set command.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.46.4 HRV Bypass Set Command

This command is used to set the bypass mode when the ventilation system is set to manual mode. If the system is not in manual mode while receiving this command it MUST be ignored.

Table 2.256: HRV Bypass Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_BYPASS_SET (0x04)							
Bypass							

Bypass (8 bits)

The value MAY be 0x00 (close) or 0xFF (open).

Furthermore, the field MAY carry a percentage value between 1 to 99 (0x01 - 0x63). If the ventilation system supports modulated bypass, the percentage value will represent the aperture of the bypass. If the system does not support the full range of aperture steps, the values SHOULD be mapped linearly over the entire scale.

If ventilation system does support modulated bypass the values from 1 to 99 MUST be interpreted as fully open.

The value 254 (0xFE) MAY be used to set the bypass into automatic mode.

2.2.46.5 HRV Bypass Get Command

This command is used to request the current bypass setting.

The HRV Bypass Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.257: HRV Bypass Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_BYPASS_GET (0x05)							

2.2.46.6 HRV Bypass Report Command

This command is used to report the current bypass parameters.

Table 2.258: HRV Bypass Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_BYPASS_REPORT (0x06)							
Bypass							

Bypass (8 bits)

See description in Section 2.2.46.4 HRV Bypass Set command.

2.2.46.7 HRV Ventilation Rate Set Command

This command is used to set the ventilation rate when the ventilation system is set to manual mode. If the system is not in manual mode while receiving this command it MUST be ignored.

Table 2.259: HRV Ventilation Rate Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_VENTILATION_RATE_SET (0x07)							
Ventilation Rate							

Ventilation Rate (8 bits)

The value MAY be 0x00 (off) or 0xFF (on).

The field MAY carry a percentage value between 1 to 99 (0x01 - 0x63). A ventilation system MAY map the values 1..99 to less than 99 steps.

2.2.46.8 HRV Ventilation Rate Get Command

This command is used to request the current ventilation rate setting.

The HRV Ventilation Rate Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.260: HRV Ventilation Rate Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_VENTILATION_RATE_GET (0x08)							

2.2.46.9 HRV Ventilation Rate Report Command

This command is used to report the current ventilation rate setting.

Table 2.261: HRV Ventilation Rate Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_VENTILATION_RATE_SET (0x07)							
Ventilation Rate							

Ventilation Rate (8 bits)

See description under Section 2.2.46.7 HRV Ventilation Rate Set Command.

2.2.46.10 HRV Mode Supported Get Command

This command is used to request the supported modes from the device.

The HRV Mode Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.262: HRV Mode Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_MODE_SUPPORTED_GET (0x0A)							

2.2.46.11 HRV Mode Supported Report Command

This command is used to report the supported modes from the device.

Table 2.263: HRV Mode Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL (0x39)							
Command = HRV_CONTROL_MODE_SUPPORTED_REPORT (0x0B)							
Reserved				Manual Control Supported			
Bit Mask 1							
...							
Bit Mask N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Manual Control Supported (3 bits)

The manual control supported bits describes the supported manual control modes of the ventilation system. The mode is supported if the bit is 1. If the bit is 0, the mode is not supported.

The bits are mapped to the following controls:

Table 2.264: HRV Manual Control Support

Bit	Name
0	Bypass Open / Close
1	Bypass Auto
2	Modulated Bypass
3	Ventilation Rate

E.g. a ventilation system supporting only open and close would report Manual Control Supported = 0x01.

Bit Mask (N bytes)

The Bit Mask fields describe the supported modes by the device.

- Bit 0 in Bit Mask 1 field indicates if Mode = 0 (Off) is supported.
- Bit 1 in Bit Mask 1 field indicates if Mode = 1 (Demand / Automatic) is supported.
- …

If the Mode is supported the bit MUST be set to 1. If the Mode is not supported the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported mode. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

All available modes are given in Section 2.2.46.1 HRV Mode Set.

2.2.47 Humidity Control Mode Command Class, version 1

The Humidity Control Mode Command Class is used to control a humidity control device.

2.2.47.1 Humidity Control Mode Set Command

This command is used to set the humidity control mode in the device.

Table 2.265: Humidity Control Mode Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE (0x6D)							
Command = HUMIDITY_CONTROL_MODE_SET (0x01)							
Reserved				Mode			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Mode (4 bits)

The encoding of the humidity control mode field MUST be according to Table 2.266.

Table 2.266: Humidity Control Mode

Mode	Description	Version
0	Off - Humidity control system is off.	1
1	Humidify - The system will attempt to raise humidity to the humidifier setpoint.	1
2	De-humidify - The system will attempt to lower the humidity to the de-humidifier setpoint.	1
3	Auto - The system will automatically switch between humidifying and de-humidifying in order to satisfy the humidify and de-humidify setpoints	2

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

2.2.47.2 Humidity Control Mode Get Command

This command is used to request the supported humidity control modes from the device.

The Humidity Control Mode Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.267: Humidity Control Mode Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE (0x6D)							
Command = HUMIDITY_CONTROL_MODE_GET (0x02)							

2.2.47.3 Humidity Control Mode Report Command

This command is used to report the humidity control mode from the device.

Table 2.268: Humidity Control Mode Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE (0x6D)							
Command = HUMIDITY_CONTROL_MODE_REPORT (0x03)							
Reserved				Mode			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Mode (4 bits)

The encoding of the humidity control mode field MUST be according to Table 2.266.

2.2.47.4 Humidity Control Mode Supported Get Command

This command is used to request the supported modes from the device.

The Humidity Control Mode Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.269: Humidity Control Mode Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE (0x6D)							
Command = HUMIDITY_CONTROL_MODE_SUPPORTED_GET (0x04)							

2.2.47.5 Humidity Control Mode Supported Report Command

This command is used to report the supported humidity control modes from the device.

Table 2.270: Humidity Control Mode Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE (0x6D)							
Command = HUMIDITY_CONTROL_MODE_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields MUST advertise the supported humidity control modes by the device. The encoding of the Size field MUST be according to Table 2.271.

Table 2.271: Supported Humidity Control Mode

Field	Bit	Support for Control Mode	Version
Bit Mask 1	0	*Reserved*	1
Bit Mask 1	1	1 (Humidifier)	1
Bit Mask 1	2	2 (De-humidifer)	1
Bit Mask 1	3	3 (Auto)	2

All Bit Mask fields and bits not specified above are reserved. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

For each individual bit, The value '0' MUST signify that the actual Mode is not supported.

The value '1' MUST signify that the actual Mode is supported.

A sending node MAY omit trailing Bit Mask fields if they are not needed. The number of Bit Mask fields MUST be determined from the length field in the frame.

2.2.48 Humidity Control Operating State Class, version 1

The Humidity Control Operating State Command Class is used to obtain the operating state of the humidity control device.

2.2.48.1 Humidity Control Operating State Get Command

This command is used to request the operating state of the humidity control device.

The Humidity Control Operating State Report Command MUST be returned in response to thiscommand.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.272: Humidity Control Operating State Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_OPERATING_STATE (0x6E)							
Command = HUMIDITY_CONTROL_OPERATING_STATE_GET (0x01)							

2.2.48.2 Humidity Control Operating State Report Command

This command is used to report the operating state of the humidity control device.

Table 2.273: Humidity Control Operating State Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_OPERATING_STATE (0x6E)							
Command = HUMIDITY_CONTROL_OPERATING_STATE_REPORT (0x02)							
Reserved				Operating State			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Operating State (4 bits)

The Operating State field MUST advertise the current operating state.

The encoding of the Operating State field MUST be according to Table 2.274.

Table 2.274: Operating State

Operating State	Description	Version
0	Idle	1
1	Humidifying	1
2	De-humidifying	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

2.2.49 Humidity Control Setpoint Command Class, version 1-2

The Humidity Control Setpoint Command Class is used for humidity control setpoint handling.

2.2.49.1 Humidity Control Setpoint Set Command

This command is used to set the humidity control setpoint in the device.

Table 2.275: Humidity Control Setpoint Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_SET (0x01)							
Reserved				Setpoint Type			
Precision			Scale		Size		
Value 1							
...							
Value N							

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

The Setpoint Type field MUST specify the humidity control setpoint to be set in the humidity control device. The encoding of the Setpoint Type field MUST be according to Table 2.276.

Table 2.276: Setpoint Type

Setpoint Type	Description	Version
1	Humidifier	1
2	De-humidifier	1
3	Auto	2

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Precision (3 bits)

The precision field MUST specify the precision of the setpoint value. The number indicates the number of decimals. As an example, the decimal value 1025 with precision two (2) is equal to 10.25.

Scale (2 bits)

The Scale field MUST specify the humidity scale used. The encoding of the Scale field MUST be according to Table 2.277.

Table 2.277: Scale Values

Scale	Description	Version
0	Percentage value	1
1	Absolute humidity (g/m3)	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

The Size field MUST specify the number of bytes used for the Value field. The encoding of the Size field MUST be according to Table 2.278.

Table 2.278: Size Values

Size	Description	Version
1	Value field is 1 byte long	1
2	Value field is 2 bytes long	1
4	Value field is 4 bytes long	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Value (N bytes)

The value is a signed field. The field MAY be one, two, or four bytes in size as specified by the Size field. Value 1 is the most significant byte.

2.2.49.2 Humidity Control Setpoint Get Command

This command is used to request the given humidity control setpoint type in a device.

The Humidity Control Setpoint Report MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.279: Humidity Control Setpoint Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_GET (0x02)							
Reserved				Setpoint Type			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field MUST indicate which humidity control setpoint MUST be returned by the supporting node. The encoding of the Setpoint Type field MUST be according to Table 2.276.

2.2.49.3 Humidity Control Setpoint Report Command

This command is used to report the value of the humidity control setpoint type in the device.

Table 2.280: Humidity Control Setpoint Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_REPORT (0x03)							
Reserved				Setpoint Type			
Precision			Scale		Size		
Value 1							
...							
Value N							

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

For fields' description, refer to Section 2.2.49.1 Humidity Control Setpoint Set Command

2.2.49.4 Humidity Control Setpoint Supported Get Command

This command is used to request the humidity control setpoint types supported by the device.

The Humidity Control Setpoint Supported Report Command MUST be returned in response to a this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.281: Humidity Control Setpoint Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_SUPPORTED_GET (0x04)							

2.2.49.5 Humidity Control Setpoint Supported Report Command

This command is used to report the humidity control setpoint types supported by the device.

Table 2.282: Humidity Control Setpoint Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields MUST advertise the humidity control setpoint types supported by the device.The encoding of the Size field MUST be according to Table 2.283.

Table 2.283: Humidity Control Setpoint Supported Report values

Field	Bit	Support for Setpoint Type	Version
Bit Mask 1	0	*Reserved*	1
Bit Mask 1	1	1 (Humidifier)	1
Bit Mask 1	2	2 (De-humidifier)	1
Bit Mask 1	3	3 (Auto)	2

All Bit Mask fields and bits not specified above are reserved. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

For each individual bit, The value '0' MUST signify that the actual Setpoint Type is not supported.

The value '1' MUST signify that the actual Setpoint Type is supported.

A sending node MAY omit trailing Bit Mask fields if they are not needed. The number of Bit Mask fields MUST be determined from the length field in the frame.

2.2.49.6 Humidity Control Setpoint Scale Supported Get Command

This command is used to retrieve the supported scales of the humidity control setpoints in the device.

The Humidity Control Scale Supported Report Command MUST be returned in response to this command.

Table 2.284: Humidity Control Setpoint Scale Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_SCALE_SUPPORTED_GET (0x06)							
Reserved				Setpoint Type			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

Refer to the description under the Section 2.2.49.1 Humidity Control Setpoint Set Command.

2.2.49.7 Humidity Control Setpoint Scale Supported Report Command

This command indicates the supported scales of the humidity control setpoint in a bit mask format.

Table 2.285: Humidity Control Setpoint Scale Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_SCALE_SUPPORTED_REPORT (0x07)							
Reserved				Scale Bit Mask			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Scale Bit Mask (4 bits)

Refer to the description under Section 2.2.49.1 Humidity Control Setpoint Set Command.

2.2.49.8 Humidity Control Setpoint Capabilities Get Command

This command is used to request the minimum and maximum setpoint values for a given humidity control Setpoint Type.

The Humidity Control Setpoint Capabilities Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.286: Humidity Control Setpoint Capabilities Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_CAPABILITIES_GET (0x08)							
Reserved				Setpoint Type			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

Refer to the description under Section 2.2.49.1 Humidity Control Setpoint Set Command.

2.2.49.9 Humidity Control Setpoint Capabilities Report Command

This command is used to report the minimum and maximum values of the requested humidity control setpoint type.

Table 2.287: Humidity Control Setpoint Capabilities Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT (0x64)							
Command = HUMIDITY_CONTROL_SETPOINT_CAPABILITIES_REPORT (0x09)							
Reserved				Setpoint Type			
Precision			Scale		Size		
Minimum Value 1							
...							
Minimum Value N							
Precision			Scale		Size		
Maximum Value 1							
...							
Maximum Value N							

Refer to the description under Section 2.2.49.1 Humidity Control Setpoint Set Command for other parameter/field descriptions.

2.2.50 IR Repeater Command Class, version 1

The IR Repeater Command Class is used to get supporting nodes to learn and/or repeat IR signals used to control appliances.

A supporting node MUST have the capability to issue or receive IR signals.

2.2.50.1 IR Repeater Capabilities Get Command

This command is used to request the IR Repeater capabilities of a node.

The IR Repeater Capabilities Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.288: IR Repeater Capabilities Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_CAPABILITIES_GET (0x01)							

2.2.50.2 IR Repeater Capabilities Report Command

This command is used to advertise the IR Repeater capabilities of the sending node.

Table 2.289: IR Repeater Capabilities Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_CAPABILITIES_REPORT (0x02)							
Number of IR Code identifiers for learning							
IRR	LCR	RLC	PES	BES	Duty Cycle bitmask		
Carrier							
Minimum Sub Carrier							
Maximum Sub Carrier							
Minimum Pulse time unit (MSB)							
Minimum Pulse time unit (LSB)				Maximum Pulse time unit (MSB)			
Maximum Pulse time unit (LSB)							

Reserved / Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of IR Code identifiers for learning (8 bits)

This field indicates how many slots the sending node supports for learning and repeating IR Codes.

The value 0 MUST indicate that the no code can be learned and repeated.

Values in the range 1..255 MUST indicate the number of codes that can be learnt and repeated.

A node advertising a value greater than 0 MUST support the following commands:

- IR Repeater IR Code Learning Start Command
- IR Repeater IR Code Learning Stop Command
- IR Repeater IR Code Learning Status Command
- IR Repeater Learnt IR Code Remove Command

- IR Repeater Learnt IR Code Get Command
- IR Repeater Learnt IR Code Report Command

A node advertising a value greater than 0 MUST set the *LCR* field and/or *RLC* field to 1.

IRR (IR Repeater Repeat Command) (1 bit)

This field indicates if the nodes supports the following commands:

- IR Repeater Repeat Command

The value 0 MUST indicate that the commands above are not supported.

The value 1 MUST indicate that the commands above are supported.

LCR (Learnt Code Readback) (1 bit)

This field indicates if the supporting node can be read back learnt codes, so that a controlling node can read back current learnt codes. This field indicates if the nodes supports the following commands:

- IR Repeater Learnt IR Code Readback Get Command
- IR Repeater Learnt IR Code Readback Report Command

The value 0 MUST indicate that the commands above are not supported.

The value 1 MUST indicate that the commands above are supported.

If this field is set to 1, the *Number of IR Code Identifiers for learning* field MUST be set to a value greater than 0.

If this field is set to 1, the *IRR* field MUST also be set to 1.

RLC (Repeat Learnt Code) (1 bit)

This field indicates if the supporting node can be repeat learnt codes. This field indicates if the nodes supports the following commands:

- IR Repeater Repeat Learnt Code Command

The value 0 MUST indicate that the command above is not supported.

The value 1 MUST indicate that the command above is supported.

If this field is set to 1, the *Number of IR Code Identifiers for learning* field MUST be set to a value greater than 0.

If this field is set to 1, the *IRR* field MUST also be set to 1.

PES (Pulse Encoding Support) (1 bit)

This field indicates if the supporting node supports Pulse Encoding of the repeat and/or learn codes.

The value 0 MUST indicate that Pulse Encoding of IR signals is not supported.

The value 1 MUST indicate that Pulse Encoding of IR signals is supported.

If this field is set to 0, the *BES* field MUST be set to 1.

BES (Bit Encoding Support) (1 bit)

This field indicates if the supporting node supports Bit Encoding of the repeat and/or learn codes.

The value 0 MUST indicate that Bit Encoding of IR signals is not supported.

The value 1 MUST indicate that Bit Encoding of IR signals is supported.

If this field is set to 0, the *PES* field MUST be set to 1.

If this field is set to 1, the sending node MUST support the following commands:

- IR Repeater Configuration Set Command
- IR Repeater Configuration Get Command
- IR Repeater Configuration Report Command.

Duty Cycle bitmask (3 bits)

This field is used to advertise the supported Duty Cycles.

This field MUST be treated as a bitmask and encoded as follows:

- Bit 0 in Duty Cycle field MUST represent '1/2 Duty Cycle'.
- Bit 1 in Duty Cycle field MUST represent '1/3 Duty Cycle'.
- Bit 2 in Duty Cycle field MUST represent '1/4 Duty Cycle'.

If a Duty Cycle is supported, the corresponding bit MUST be set to '1'.

If a Duty Cycle is not supported, the corresponding bit MUST be set to '0'.

Carrier (8 bits)

This field is used to indicate the IR Carrier that the supporting node will use for repeating IR Codes indicated when receiving the IR Repeater Repeat Command.

This field MUST be expressed as 820 + x nm (nanometers). The values 0x00..0xFF MUST represents 820nm..1075nm.

Minimum Sub Carrier (8 bits)

This field MUST indicates the lower limit value that can be set at the supporting node in the IR Repeater Repeat Command for the *Sub Carrier* field.

This field MUST be expressed in kHz (kilo Hertz).

Maximum Sub Carrier (8 bits)

This field MUST indicates the upper limit value that can be set at the supporting node in the IR Repeater Repeat Command for the *Sub Carrier* field.

This field MUST be expressed in kHz (kilo Hertz).

Minimum Pulse time Unit (12 bits)

This field MUST indicates the lower limit value that can be set at the supporting node in the IR Repeater IR Code Learning Start Command and IR Repeater Repeat Command for the *Pulse time unit* field.

If the *PES* field is set to 0, this field MUST set to 0.

Maximum Pulse time Unit (12 bits)

This field MUST indicates the upper limit value that can be set at the supporting node in the IR Repeater IR Code Learning Start Command and IR Repeater Repeat Command for the *Pulse time unit* field.

If the *PES* field is set to 0, this field MUST set to 0.

2.2.50.3 IR Repeater IR Code Learning Start Command

This command is used to instruct a supporting node to start learning an IR Code.

This command MUST be ignored by a receiving node if it advertises 0 *Number of IR Code identifiers for learning* in the IR Repeater Capabilities Report Command.

A supporting node MUST stop learning IR Code and return a IR Repeater IR Code Learning Status Command a when a code has been received/learnt.

This command MUST be ignored if the supporting node is already learning an IR code.

Table 2.290: IR Repeater IR Code Learning Start Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_IR_CODE_LEARNING_START (0x03)							
IR Code identifier							
Timeout							
Reserved				Pulse time unit (MSB)			
Pulse time unit (LSB)							

IR Code identifier (8 bits)

This field indicates which IR Code identifier to use for storing the IR Code that will be learnt.

Values in the range 1..255 MUST indicate the actual IR Code identifier to use for learning a new code.

This command MUST be ignored if this field is set to 0 or a larger value than the *Number of IR Code identifiers for learning* field advertised in the IR Repeater Capabilities Report Command.

Timeout (8 bits)

This field is used to indicate how long the supporting node MUST wait for an IR code to learn.

If no IR code is learnt within the time in seconds indicated by this field, a supporting node MUST stop learning codes and return a IR Repeater IR Code Learning Status Command with the *status* field set to 0x02 (Failed:timeout).

Pulse time unit (12 bits)

This field is used to configure the IR Code learning Pulse Encoding time unit.

This field MUST be expressed in (µs) and be in the range 1..4096.

Values lower than the *Minimum Pulse time* unit advertised by the supporting node in the IR Repeater Capabilities Report Command MUST be ignored by a supporting node.

Values higher than the *Maximum Pulse time* unit advertised by the supporting node in the IR Repeater Capabilities Report Command MUST be ignored by a supporting node.

2.2.50.4 IR Repeater IR Code Learning Stop Command

This command is used to instruct a supporting node to stop learning an IR Code.

This command MUST be ignored by a receiving node if it advertises 0 Number of *IR Code identifiers for learning* in the IR Repeater Capabilities Report Command.

A supporting node MUST stop learning IR Code if a learning process was ongoing when this command is received.

A supporting node MUST return a IR Repeater IR Code Learning Status Command in response to this command.

Table 2.291: IR Repeater IR Code Learning Stop Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_IR_CODE_LEARNING_STOP (0x04)							

2.2.50.5 IR Repeater IR Code Learning Status Command

This command is used to report the status of the last IR Code learning operation. It MUST be issued by a supporting node when it stops learning IR Codes.

Table 2.292: IR Repeater IR Code Learning Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_IR_CODE_LEARNING_STATUS (0x05)							
IR Code identifier							
Status							

IR Code identifier (8 bits)

This field indicates which IR Code identifier has been used for storing the IR Code during IR Code learning.

If a learning process was completed, interrupted or timed out, this field MUST be in the range 1..255 and MUST be set to the same value as the *IR Code identifier* field of the IR Repeater IR Code Learning Start Command that initiated IR Code learning.

If a no learning process was ongoing and a IR Repeater IR Code Learning Stop Command was received, this field MUST be set to 0.

Status(8 bit)

This field is used to indicate the status of the IR Code learning operation. This field MUST encoded according to Table 2.293.

Table 2.293: IR Code Learning Status::Status encoding

Value	Description
0x00	**Failed: Learning buffer overflow** The node learning buffer is too small to learnt the entire IR Code.
0x01	**Failed: Forced learning stop** The node has received an IR Repeater IR Code Learning Stop Command before any code was learnt
0x02	**Failed: timeout** The node has timed out and did not learn any code
0xFF	**Success** A new code was learnt.

2.2.50.6 IR Repeater Learnt IR Code Remove Command

This command is used to instruct a supporting node to erase a previously learnt an IR Code.

This command MUST be ignored by a receiving node if it advertises 0 *Number of IR Code identifiers for learning* in the IR Repeater Capabilities Report Command.

Table 2.294: IR Repeater Learnt IR Code Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_LEARNT_IR_CODE_REMOVE (0x06)							
IR Code identifier							

IR Code identifier (2 bytes)

This field indicates which IR Code identifier to erase.

Values in the range 1..255 MUST indicate the actual IR Code identifier that MUST be erased by the supporting node.

The value 0 MUST indicate to erase all IR codes.

This command MUST be ignored if this field is set to a larger value than the *Number of IR Code identifiers for learning* field advertised in the IR Repeater Capabilities Report Command.

2.2.50.7 IR Repeater Learnt IR Code Get Command

This command is used to request a supporting node which IR Code identifiers have a valid IR Code learnt.

This command MUST be ignored by a receiving node if it advertises 0 *Number of IR Code identifiers for learning* in the IR Repeater Capabilities Report Command.

The IR Repeater Learnt IR Code Report Command MUST be returned in response to this command if it was not ignored.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.295: IR Repeater Learnt IR Code Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_LEARNT_IR_CODE_GET (0x07)							

2.2.50.8 IR Repeater Learnt IR Code Report Command

This command is used to advertise the current IR learnt code stored at the supporting node.

Table 2.296: IR Repeater Learnt IR Code Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_LEARNT_IR_CODE_REPORT (0x08)							
Reserved			IR Code identifier status length				
IR Code identifier status bitmask 1							
...							
IR Code identifier status bitmask N							

IR Code identifier status length (5 bits)

This field is used to indicate the length in bytes of the *IR Code identifier status bitmask* field.

This field MUST be set to the minimum value which allows advertising all IR Code currently set at the supporting node.

IR Code identifier status bitmask (N bytes)

This field advertises the supported IR Code identifier status values. The length of this field in bytes MUST match the value advertised in the *IR Code Identifier Status Length* field.

This field MUST be encoded as a bitmask:

- Bit 0 in Bit Mask 1 represents IR Code Identifier 0x01.
- Bit 1 in Bit Mask 1 represents IR Code Identifier 0x02.
- ...

If an IR Code is stored for the IR Code Identifier, the corresponding bit MUST be set to '1'.

If no IR Code is stored for the IR Code Identifier, the corresponding bit MUST be set to '0'.

Only IR Codes with their bit set to '1' can be repeated with the IR Repeater Repeat Learnt Code Command

2.2.50.9 IR Repeater Learnt IR Code Readback Get Command

This command is used to request a supporting node has learnt an IR Code to return the code details to a controlling node.

This command MUST be ignored by a receiving node if it advertises no support for *LCR (Learnt Code Readback)* in the IR Repeater Capabilities Report Command.

The IR Repeater Learnt IR Code Report Command MUST be returned in response to this command if it was not ignored.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.297: IR Repeater Learnt IR Code Readback Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_LEARNT_IR_CODE_READBACK_GET (0x09)							
IR Code identifier status length							
Res	Report Number (MSB)						
Report Number (LSB)							

IR Code identifier (8 bits)

This field is used to indicate the IR Code identifier that the supporting node MUST return in response to this command.

A supporting node receiving a not supported IR Code Identifier or current unset/empty IR Code Identifier MUST return a report with *Report Number* set to 0, Last set to 1 and no Data block field.

Report Number (15 bits)

This field indicates the data block sequence number of the requested Learnt IR Code. The first data block MUST be identified by the Report Number value 1.

If this field is set to 0 or a value higher than the total number of reports (indicated with the *last* field in the report), a supporting node MUST return a report with no *Data block* field.

2.2.50.10 IR Repeater Learnt IR Code Readback Report Command

This command is used to advertise the "contents" of a Learn IR Code.

Table 2.298: IR Repeater Learnt IR Code Readback Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_LEARNT_IR_CODE_READBACK_REPORT (0x0A)							
IR Code identifier							
Sub Carrier							
Reserved		Duty Cycle		Pulse time unit (MSB)			
Pulse time unit (LSB)							
Last	Report Number (MSB)						
Report Number (LSB)							
Data block type 1		Data block length 1					
Data block value 1, 1							
...							
Data block value 1, N							
...							
Data block type M		Data block length M					
Data block value M, 1							
...							
Data block value M, N							

IR Code identifier (8 bits)

This field is used to indicate the IR Code identifier that the supporting node is currently sending.

Sub Carrier (8 bits)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

Duty Cycle (2 bits)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

Pulse time unit (12 bits)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

Report Number(15 bits)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

Last (1 bit)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

Data block type (2 bits)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

Data Block Length (5 bits)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

Data Block value (N bytes)

For field description, refer to Section 2.2.50.15 IR Repeater Repeat Command

2.2.50.11 IR Repeater Configuration Set Command

This command is used to configure the IR signal properties before the supporting node would repeat IR Codes.

This command MUST be ignored by a receiving node if it advertises no support for Bit Encoding Support (BES) in the IR Repeater Capabilities Report Command.

A supporting node MUST have a default bit 1, bit 0 and period configured after reset.

Table 2.299: IR Repeater Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_CONFIGURATION_SET (0x0B)							
Period (MSB)							
Reserved				Period (LSB)			
Data bit 0 encoding length (MSB)							
Data bit 0 encoding length (LSB)							
Data bit 0 encoding 1							
...							
Data bit 0 encoding N							
Data bit 1 encoding length (MSB)							
Data bit 1 encoding length (LSB)							
Data bit 1 encoding 1							
...							
Data bit 1 encoding M							

Period (12 bits)

This field is used to indicate the time period used for data bit encoding.

This field MUST be expressed in micro seconds (µs). This field MUST be in the range 1..4096

Data bit '0' encoding length (2 bytes)

This field is used to indicate the length in **bits** of the *Data bit '0' encoding* field.

This field MUST be set to the exact number of bits to be interpreted from the *Data bit '0' encoding* field.

This field MUST be ignored by a supporting node if the *BES* field is set to 0 in the IR Repeater Capabilities Report Command.

Data bit '0' encoding (N bytes)

This field advertises the Hi/Low polarization encoding to apply to encode a 0 in an IR Code using the IR Repeater Repeat Command and IR Repeater Learnt IR Code Readback Report Command.

This field MUST be ignored by a supporting node if the *BES* field is set to 0 in the IR Repeater Capabilities Report Command.

The length of this field MUST be the minimum amount of full bytes allowing to have the number of bits indicated in the the *Data bit '0' encoding Length* field. .e.g if the *Data bit '0' encoding* field is set to 13, this field MUST be 2 bytes long.

Bits MUST be encoded as follows:

- The value 0 MUST indicate a Low polarization
- The value 1 MUST indicate a High polarization

The whole sequence indicated in this field MUST be repeated by a supporting node when receiving a IR Repeater Repeat Command indicating to repeat a bit 0.

Data bit '1' encoding length (2 bytes)

This field is used to indicate the length in **bits** of the Data bit '1' encoding field.

This field MUST be set to the exact number of bits to be interpreted from the *Data bit '1' encoding* field.

This field MUST be ignored by a supporting node if the *BES* field is set to 0 in the IR Repeater Capabilities Report Command.

Data bit '1' encoding (N bytes)

This field advertisesthe Hi/Low polarization encoding to apply to encode a 1 in an IR Code using the IR Repeater Repeat Command and IR Repeater Learnt IR Code Readback Report Command.

This field MUST be ignored by a supporting node if the *BES* field is set to 0 in the IR Repeater Capabilities Report Command.

The length of this field MUST be the minimum amount of full bytes allowing to have the number of bits indicated in the the *Data bit '1' encoding Length* field. .e.g if the *Data bit '1' encoding* field is set to 13, this field MUST be 2 bytes long.

Bits MUST be encoded as follows:

- The value 0 MUST indicate a Low polarization
- The value 1 MUST indicate a High polarization

The whole sequence indicated in this field MUST be repeated by a supporting node when receiving a IR Repeater Repeat Command indicating to repeat a bit 1.

2.2.50.12 IR Repeater Configuration Get Command

This command is used to request the IR signal configuration at the supporting node.

The IR Repeater Configuration Report Command MUST be returned in response to this command.

This command MUST be ignored by a receiving node if it advertises no support for *Bit Encoding Support (BES)* in the IR Repeater Capabilities Report Command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.300: IR Repeater Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_CONFIGURATION_GET (0x0C)							

2.2.50.13 IR Repeater Configuration Report Command

This command is used to advertise the current IR signal repeating configuration at the supporting node.

Table 2.301: IR Repeater Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_CONFIGURATION_REPORT (0x0D)							
Period (MSB)							
Reserved				Period (LSB)			
Data bit 0 encoding length (MSB)							
Data bit 0 encoding length (LSB)							
Data bit 0 encoding 1							
...							
Data bit 0 encoding N							
Data bit 1 encoding length (MSB)							
Data bit 1 encoding length (LSB)							
Data bit 1 encoding 1							
...							
Data bit 1 encoding M							

For fields' description, refer to Section 2.2.50.11 IR Repeater Configuration Set Command

2.2.50.14 IR Repeater Repeat Learnt Code Command

This command is used to request the receiving node to send an IR Code.

Table 2.302: IR Repeater Repeat Learnt Code Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_REPEAT_LEARNT_CODE (0x0E)							
IR Code identifier							

IR Code identifier (8 bits)

This field indicates which learnt IR Code identifier to repeat.

Values in the range 1..255 MUST indicate the actual IR Code identifier to use for repeating the learnt code.

This command MUST be ignored if this field is set to 0, if no IR Code is learnt for the received Identifier or if this field is set to a larger value than the *Number of IR Code identifiers for learning* field advertised in the IR Repeater Capabilities Report Command.

A supporting node MUST repeat the IR code if a code has been learnt for the actual IR Code identifier.

2.2.50.15 IR Repeater Repeat Command

This command is used to request the receiving node to send/repeat an IR Code.

This command MUST be ignored if the supporting node advertises no support for IR Code Repeating *(IRR)* in the IR Repeater Capabilities Report Command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.303: IR Repeater Repeat Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IR_REPEATER (0xA0)							
Command = IR_REPEATER_REPEAT (0x0F)							
Sequence number							
Sub Carrier							
Reserved		Duty Cycle		Pulse time unit (MSB)			
Pulse time unit (LSB)							
Last	Report Number (MSB)						
Report Number (LSB)							
Data block type 1	Data block length 1						
Data block value 1, 1							
...							
Data block value 1, N							
...							
Data block type M	Data block length M						
Data block value M, 1							
...							
Data block value M, N							

Sequence number (8 bits)

This field is used to keep track of the command that initiated the IR Repeat Command.

Sub carrier (8 bits)

This field is used to indicate the IR Sub carrier that the supporting node MUST use for repeating IR Codes.

This field MUST be expressed in kHz (kiloHertz).

Duty cycle (2 bits)

This field is used to indicate the IR Carrier that the supporting node MUST use for repeating IR Codes.

This field MUST be encoded according to Table 2.304.

Table 2.304: IR Repeater Repeat::Duty Cycle encoding

Value	Description
0x00	1/2 Duty Cycle
0x01	1/3 Duty Cycle
0x02	1/4 Duty Cycle
0x03	*Reserved*

Values not advertised as supported in the IR Repeater Capabilities Report Command MUST be ignored by a supporting node.

Pulse time unit (12 bits)

This field is used to interpret the data blocks using Pulse Encoding type.

This field MUST be expressed in (µs) and be in the range 1..4096.

Values lower than the *Minimum Pulse Time Unit* advertised by the supporting node in the IR Repeater Capabilities Report Command MUST be ignored by a supporting node.

Values higher than the *Maximum Pulse Time Unit* advertised by the supporting node in the IR Repeater Capabilities Report Command MUST be ignored by a supporting node.

Report number (15 bits)

A sending node typically needs several report commands to describe or encode an entire IR Code. This field is used to indicate the current report number being issued. The first report MUST have this field set to 1 and incremented at every new unique report Command.

A receiving node MUST reassemble the IR Code by concatenating the data blocks of each repeat command.

Data block 1,1 of the first report MUST represent the beginning of the IR Code signal and data block M,N of the last report MUST represent the end of the IR Code signal.

Last (1 bit)

This field is used to indicate that this is the Last report.

The value 0 MUST indicate that this is not the last report and more are to follow.

The value 1 MUST indicate that this is the last report and the supporting node MUST start repeating the IR Code.

Data block type (2 bits)

This field indicates how to interpret the corresponding Data block Value field.

- 0 : bit encoding (0/1)
- 1 : pulse encoding

Data block length (6 bits)

This field indicates the length in bytes of the corresponding Data block Value field.

Data block value (6 bits)

This field MUST interpreted according to the corresponding *Data Block Type* field.

If the corresponding *Data Block Type* field is set to 0, this field MUST be interpreted as follow:

Table 2.305: Data Block Value Field::Data Block Type = '0'

7	6	5	4	3	2	1	0
Number of bits 1							
Number of bits 2							
Data bits 1							
...							
Data bits L							

Number of bits (2 bytes)

This subfield indicates how many bits to take from the *Data bits* subfield, starting from the LSB. For example, if this subfield is set to 12, a receiving node MUST interpret the first byte and the bits 0..3 of the second byte as data bits.

Data bits (L bytes)

This subfield is used to describe successive bits to be transmitted as part of the IR Code. Each data bit (bit 0 or bit 1) MUST be repeated/interpreted according to the IR Configuration set at the supporting node. (IR Repeater Configuration Set Command)

If the corresponding Data Block Type is set to 1, this field MUST be interpreted as follow:

Table 2.306: Data Block Value Field::Data Block Type = '1'

7	6	5	4	3	2	1	0
Number of time units 1							
...							
Number of time units P							

Number of time units (8 bits)

This subfield MUST represent the amount of time to use for an IR code pulse. The time unit is set in the *Pulse time unit* field.

This field MUST be interpreted byte-by-byte. The first byte is the first to be interpreted. It represents how long to transmit a Hi or Lo polarity.

The first byte (1) MUST represents a Hi Polarity.

If a byte of this field is in the range 0..255, it MUST be interpreted as a multiplier of the time unit.

If a byte of this field is in the range 0..254, the polarity MUST be toggled for the next byte (from Hi to Lo or Lo to Hi)

If a byte of this field is set to 255, the polarity MUST be the same for the next byte (Hi stays Hi and Lo stays Lo)If a byte of this field is set to 0, it must ignored. The value 0 can be used for example at the first byte to start an IR Code with a Lo polarity.

2.2.51 Irrigation Command Class, version 1

The Irrigation Command Class provides commands to manage irrigation systems.

2.2.51.1 Terminology

In an irrigation system, a **zone valve** is turned on to allow a certain group of sprinklers to water an area (or **zone**) for a specified duration. The terms **zone valve** and **valve** are used synonymously, whereas a **main valve** is always identified as such.

A valve that is **on** or **open** lets the water run through the valve.

A valve that is **off** or **closed** blocks the water from running through the valve.

A **main valve** allows water to flow to a set of zone valves, and must be turned on before water flows to the zone valves. In version 1 of this Command Class, there is only one main valve in an irrigation system.

A valve is identified with a **ValveID**. The main valve and zone valves have their own ID pools, each starting with 1.

In some cases a main valve switch may drive a pump relay rather than an actual irrigation valve. In this case, there may be a configurable delay between the main valve (or pump) being turned on and the rest of the zone valves being turned on. When a given zone is turned on, a main valve (if present) should automatically turn on. The main valve should also automatically turn off after a device specific duration of zone valve inactivity.

A valve can be physically **connected** or **disconnected**. Attach points where valves can be connected always keep the same valve ID. A disconnected valve cannot be operated and the water does not run through a disconnected valve.

In an irrigation system, a **scheduled run** or **run** is when a set of zone valves are open and closed sequentially, one after another. Each zone valve will be turned on for a specified duration and once complete, that valve will be turned off and the next zone valve will be turned on. In a (scheduled) run, each zone valve has its own specified run duration, which is unique to that zone valve and that specific (scheduled) run. At the end of a scheduled run, the Irrigation System Shutoff Command can also be used to turn everything off.

A main valve remains on/open for the entire duration of any run, so that all zone valves will be able to provide water to their associated sprinklers.

A **valve table** is a list of zone valves and their corresponding run durations. A valve table can be created, configured and stored in an irrigation device. The valve table can subsequently be executed (or run) using a single command.

2.2.51.2 Compatibility considerations

2.2.51.2.1 Multi Channel considerations

CC:006B.01.00.22.001 Multi Channel End Points SHOULD NOT support the Irrigation Command Class.

2.2.51.3 Interoperability considerations

CC:006B.01.00.32.001 A node supporting the Irrigation Command Class SHOULD be implemented as a Multi Channel Device with the main valve and each zone valve implemented as a separate Multi Channel End Point.

CC:006B.01.00.32.002 The Root Device SHOULD support the following Command Classes:

- Irrigation Command Class
- Schedule Command Class

CC:006B.01.00.32.003 Each End Point SHOULD support the following Command Class:

- Binary Switch Command Class

CC:006B.01.00.32.004 A node supporting optional sensors SHOULD implement each sensor as a separate Multi Channel End Point.

The Notification Command Class, version 7 or newer can be used for reporting Irrigation events/states, using the Irrigation Notification Type.

Scheduling can be done using the Schedule Command Class. If supported, the Schedule Command CC:006B.01.00.32.005 Class SHOULD support alternating days and odd/even date processing.

CC:006B.01.00.33.001 Some supporting nodes MAY support having only 1 zone valve open at a time. A controlling node CC:006B.01.00.32.006 SHOULD be aware that opening a zone valve may cause another currently open zone valve to close.

2.2.51.3.1 Controlling methods

A system integrator should be aware that if an irrigation control device is controlled using both End Point Binary Switch Command Class and Root Device Irrigation Command Class at the same time, this may cause unpredictable behavior. This applies whether commands are issued instantly or scheduled.

CC:006B.01.00.32.007 It is RECOMMENDED that an irrigation controlling system is managed either via direct End Point control or via the Irrigation Command Class.

An irrigation device can be driven directly from a Z-Wave controlling node, turning on the main valve and sequentially turning on each zone valve for a specified duration.

The controlling node can also use the concept of a valve table to turn on a set of valves sequentially. A valve table allows a single command, the Irrigation Valve Table Run Command, to trigger a run. The controlling node must first create the valve tables, using the Irrigation Valve Table Set Command. The run can triggered by sending the Irrigation Valve Table Run Command at the desired time. The irrigation device will sequentially run the valves in each of the specified valve tables for their associated runtimes. Depending on the irrigation system, the controlling node may need to turn on the main valve manually using the Irrigation Valve Run Command.

2.2.51.4 Irrigation System Info Get Command

This command is used to request a receiving node about its irrigation system information.

CC:006B.01.01.11.001 The Irrigation System Info Report Command MUST be returned in response to this command.

CC:006B.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:006B.01.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.307: Irrigation System Info Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_INFO_GET (0x01)							

2.2.51.5 Irrigation System Info Report Command

This command is used to advertise irrigation system information.

Table 2.308: Irrigation System Info Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_INFO_REPORT (0x02)							
Reserved							Main Valve
Total Number of Valves							
Total Number of Valve Tables							
Reserved				Valve Table Max Size			

Reserved

CC:006B.01.02.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Main Valve (1 bit)

This field is used to indicate if a main valve is supported by the sending node.

CC:006B.01.02.11.002 The value 1 MUST indicate that the sending node supports a main valve.

CC:006B.01.02.11.003 The value 0 MUST indicate that the sending node does not support a main valve.

Total Number of Valves (8 bits)

This field is used to advertise the total number of zone valves supported by the device

CC:006B.01.02.11.004 The implemented Valve ID values MUST be in a sequence starting from 1, i.e. a node supporting 10 valves MUST accept Valve ID values in the range 1..10.

Total Number of Valve Tables (8 bits)

This field is used to advertise the total number of valve tables that can be created/stored in the device.

CC:006B.01.02.11.005 The implemented Valve Table ID MUST be in a sequence starting from 1, i.e. a node supporting 10 valve tables MUST accept Valve Table ID values in the range 1..10.

Valve Table Max Size (4 bits)

This field is used to advertise the maximum number of entries per valve table supported by the sending node.

CC:006B.01.02.11.006 The value MUST be in the range 1..15.

2.2.51.6 Irrigation System Status Get Command

This command is used to request a receiving node about its irrigation system status.

CC:006B.01.03.11.001 The Irrigation System Status Report Command MUST be returned in response to this command.

CC:006B.01.03.11.002 This command MUST NOT be issued via multicast addressing.

CC:006B.01.03.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.309: Irrigation System Status Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_STATUS_GET (0x03)							

2.2.51.7 Irrigation System Status Report Command

This command is used to advertise irrigation system status.

Table 2.310: Irrigation System Status Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_STATUS_REPORT (0x04)							
System Voltage							
Sensor Status							
Flow Precision			Flow Scale=l/h		Flow Size		
Flow Value 1							
...							
Flow Value N							
Pressure Precision			Pressure Scale=kPa		Pressure Size		
Pressure Value 1							
...							
Pressure Value N							
Shutoff Duration							
System Error Status							
Reserved							Main Valve
Valve ID							

System Voltage (8 bits)

This field advertises the voltage level applied at the sending node.

CC:006B.01.04.11.001 The advertised value MUST be expressed in Volt. This field does not indicate the voltage type (AC or DC).

CC:006B.01.04.11.002 If the node does not have a voltage sensor measuring the voltage, this field MUST be set to 0.

Sensor Status (8 bits)

This field is used to advertise if optional sensors are currently reporting values or detecting events at the sending node.

CC:006B.01.04.11.003 This field MUST be treated as a bit mask and MUST comply with Table 2.311

Table 2.311: Irrigation System Status Report::Sensor Status encoding

Bit	Sensor Status
0	Flow Sensor currently active If this bit is set to 1, the Flow Value field value MUST advertise a valid sensor reading.
1	Pressure Sensor currently active If this bit is set to 1, the Pressure Value field value MUST advertise a valid sensor reading.
2	Rain Sensor attached and active If a rain sensor is currently attached to the device and detecting rain, this bit MUST be set to 1.
3	Moisture Sensor attached and active If a moisture sensor is currently attached to the device and detecting moisture, this bit MUST be set to 1.
4..7	*Reserved*

CC:006B.01.04.11.004 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Flow Precision (3 bits)

CC:006B.01.04.11.005

This field MUST indicate how many decimals are contained in the "Flow Value" field value. For example, if the "Flow Value" field is set to 1025 and the "Flow Precision" field is set to 2, the Flow Value MUST be interpreted as 10.25.

CC:006B.01.04.11.006 If the Sensor Status field's bit 0 is set to 0, this field MUST be set to 0.

Flow Scale (2 bits)

CC:006B.01.04.11.007 This field MUST indicate which unit is used for the "Flow Value" field. This field MUST be set to 0.

CC:006B.01.04.11.008 The value 0 MUST indicate that the unit is l/h (liter/hour)

Flow Size (3 bits)

CC:006B.01.04.11.009 This field is used to advertise the length in bytes of the "Flow Value" field. This field MUST be set to 1, 2 or 4. All other values are reserved and MUST NOT be used by a sending node.

CC:006B.01.04.11.00A If the Sensor Status field's bit 0 is set to 0, this field MUST be set to 1.

Flow Value (N bytes)

This field is used to advertise the Flow value measured by the flow sensor.

CC:006B.01.04.11.00B The length of this field in byte MUST comply with the value advertised in the Flow Size field.

CC:006B.01.04.11.00C If the Sensor Status field's bit 0 is set to 0, this field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Pressure Precision (3 bits)

CC:006B.01.04.11.00D This field MUST indicate how many decimals are contained in the "Pressure Value" field value. For example, if the "Pressure Value" field is set to 1025 and the "Pressure Precision" field is set to 2, the Pressure Value MUST be interpreted as 10.25.

CC:006B.01.04.11.00E If the Sensor Status field's bit 1 is set to 0, this field MUST be set to 0.

Pressure Scale (2 bits)

CC:006B.01.04.11.00F This field MUST indicate which unit is used for the "Pressure Value" field. This field MUST be set to 0.

CC:006B.01.04.11.010 The value 0 MUST indicate that the unit is kPa (kilopascal)

Pressure Size (3 bits)

CC:006B.01.04.11.011 This field is used to advertise the length in bytes of the "Pressure Value" field. This field MUST be set to 1, 2 or 4. All other values are reserved and MUST NOT be used by a sending node.

CC:006B.01.04.11.012 If the Sensor Status field's bit 1 is set to 0, this field MUST be set to 1.

Pressure Value (N bytes)

This field is used to advertise the Pressure value measured by the pressure sensor.

CC:006B.01.04.11.013 The length of this field in bytes MUST comply with the value advertised in the Pressure Size field.

CC:006B.01.04.11.014 If the Sensor Status field's bit 1 is set to 0, this field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Shutoff Duration (8 bits)

This field is used to indicate how many hours are left in the "shut off" mode.

For description, refer to Irrigation System Shutoff Command (Section 2.2.51.21).

System Error Status (8 bits)

This field is used to advertise if any system error is being active at the sending node.

CC:006B.01.04.11.015 This field MUST be treated as a bit mask and MUST comply with Table 2.312.

Table 2.312: System Status Report::System Error Status encoding

Bit	Description
0	The device has not been programmed
1	The device has experienced an emergency shutdown.
2	The device's pressure high threshold has been triggered.
3	The device's pressure low threshold has been triggered.
4	A valve or the main valve is reporting error. Valves can be individually checked using the Irrigation Valve Info Get Command
5..7	*Reserved*

In some cases, errors are no longer active. Any inactive errors (except valve errors) can be cleared by sending an Irrigation System Config Set Command. Refer to the Irrigation Valve Info Report Command for more details on valve specific errors.

Main Valve (1 bit)

This field is used to indicate if a main valve is currently open or closed.

CC:006B.01.04.11.016 The value 1 MUST indicate that the main valve is On / Open.

CC:006B.01.04.11.017 The value 0 MUST indicate that the main valve is Off / Closed.

Valve ID (8 bits)

This field is used to indicate the Valve ID of the first open zone valve currently On / Open.

CC:006B.01.04.11.018 If no zone valve is On / Open, this field MUST be set to 0.

CC:006B.01.04.11.019 Else, this field MUST be to a value in the range 1..{Total number of supported zone valves} by a sending node

2.2.51.8 Irrigation System Config Set Command

This command allows the irrigation system to be configured accordingly.

Table 2.313: Irrigation System Config Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_CONFIG_SET (0x05)							
Main Valve Delay							
High Pressure Threshold Precision			High Pressure Threshold Scale=kPa		High Pressure Threshold Size		
High Pressure Threshold Value 1							
...							
High Pressure Threshold Value N							
Low Pressure Threshold Precision			Low Pressure Threshold Scale=kPa		Low Pressure Threshold Size		
Low Pressure Threshold Value 1							
...							
Low Pressure Threshold Value N							
Sensor Polarity							

Main Valve Delay (8 bits)

This field is used to configure a delay in seconds between turning on the mainvalve and turning on any "zone" valve.

CC:006B.01.05.11.001 This field MUST ignored by a receiving node if it does not implement a main valve.

CC:006B.01.05.11.002 The value MUST be expressed in seconds and the value 0 MUST indicate that no delay is applied when turning main and zone valves on.

Pressure High Threshold

These fields are used to configure the pressure high threshold at the receiving node.

CC:006B.01.05.13.001 A receiving node having no support for pressure sensor MAY ignore this field.

CC:006B.01.05.11.003 If the receiving node supports a Pressure Sensor, the node MUST issue an Irrigation System Status Report Command with the System Error Status indicating high pressure when the pressure crosses above this threshold.

CC:006B.01.05.11.004 The format of the Pressure High Threshold fields MUST comply with the format used by the Multilevel Sensor Command Class and MUST be expressed in kPa.

Pressure Low Threshold

These fields are used to configure the pressure low threshold at the receiving node.

CC:006B.01.05.13.002 A receiving node having no support for pressure sensor MAY ignore this field.

CC:006B.01.05.11.005 If the receiving node supports a Pressure Sensor, the node MUST issue an Irrigation System Status Report Command with the System Error Status indicating low pressure when the pressure crosses below this threshold.

CC:006B.01.05.11.006 The format of the Pressure Low Threshold fields MUST comply with the format used by the Multilevel Sensor Command Class and MUST be expressed in kPa.

Sensor Polarity (8 bits)

This field is used to configure optional sensors' polarity at the receiving node.

CC:006B.01.05.11.007 This field MUST be treated as a bit mask and MUST comply with Table 2.314. This field MUST be ignored if the 'valid' bit (bit 7) is set to 0.

Table 2.314: Irrigation System Config Set::Sensor Polarity encoding

Bit	Description
0	Rain Sensor Polarity (0 LOW, 1 HIGH) A receiving node having no support for rain sensor MAY ignore this field.
1	Moisture Sensor Polarity (0 LOW, 1 HIGH) A receiving node having no support for moisture sensor MAY ignore this field
2..6	*Reserved*
7	Valid This bit MUST be set to 1 to indicate that the other bits in the bitmask contain valid data.

CC:006B.01.05.11.008 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.51.9 Irrigation System Config Get Command

This command is used to request a receiving node about its current irrigation system configuration.

CC:006B.01.06.11.001 The Irrigation System Config Report Command MUST be returned in response to this command.

CC:006B.01.06.11.002 This command MUST NOT be issued via multicast addressing.

CC:006B.01.06.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.315: Irrigation System Config Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_CONFIG_GET (0x06)							

2.2.51.10 Irrigation System Config Report Command

This command is used to advertise the current irrigation system configuration.

Table 2.316: Irrigation System Config Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_CONFIG_REPORT (0x07)							
Main Valve Delay							
High Pressure Threshold Precision			High Pressure Threshold Scale=kPa		High Pressure Threshold Size		
High Pressure Threshold Value 1							
...							
High Pressure Threshold Value N							
Low Pressure Threshold Precision			Low Pressure Threshold Scale=kPa		Low Pressure Threshold Size		
Low Pressure Threshold Value 1							
...							
Low Pressure Threshold Value N							
Sensor Polarity							

CC:006B.01.07.11.001 For fields' description, refer to Section 2.2.51.8 Irrigation System Config Set Command. A sending node MUST comply with field descriptions in Section 2.2.51.8 Irrigation System Config Set Command.

2.2.51.11 Irrigation Valve Info Get Command

This command is used to request general information about the specified valve.

CC:006B.01.08.11.001 The Irrigation Valve Info Report Command MUST be returned in response to this command.

CC:006B.01.08.11.002 This command MUST NOT be issued via multicast addressing.

CC:006B.01.08.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.317: Irrigation Valve Info Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_INFO_GET (0x08)							
Reserved							Main Valve
Valve ID							

Reserved

CC:006B.01.08.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Main Valve (1 bit)

This field is used to indicate whether the sending node requests the information of the main valve or of a zone valve.

CC:006B.01.08.11.005 The value 1 MUST indicate that the sending node requests the main valve information.

CC:006B.01.08.11.006 The value 0 MUST indicate that the sending node requests information related to the zone valve corresponding to Valve ID field.

Valve ID (8 bits)

This field is used to indicate the Valve ID if the sending node requests the information about a zone valve.

CC:006B.01.08.11.007 This field MUST be set to a Valve ID supported by the receiving node.

CC:006B.01.08.12.001 A node receiving this command for an unsupported zone valve ID SHOULD return a report for the first supported zone valve.

CC:006B.01.08.11.008 If the Main Valve field is set to 1, this field MUST be set to 1. Other values MUST be ignored by a receiving node.

2.2.51.12 Irrigation Valve Info Report Command

This command is used to advertise general information about a given valve.

Table 2.318: Irrigation Valve Info Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_INFO_REPORT (0x09)							
Reserved						Conn- ected	Main Valve
Valve ID							
Nominal Current							
Valve Error Status							

Reserved

CC:006B.01.09.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Main Valve (1 bit)

This field is used to indicate whether the sending node advertises the information of the main valve or of a zone valve.

CC:006B.01.09.11.002 The value 1 MUST indicate that the sending node advertises the main valve information.

CC:006B.01.09.11.003 The value 0 MUST indicate that the sending node advertises information related to the zone valve corresponding to Valve ID field.

Connected (1 bit)

This field indicates if the actual valve is currently connected to the node or not.

CC:006B.01.09.11.004 The value 0 MUST indicate that the valve is disconnected from the node.

CC:006B.01.09.11.005 The value 1 MUST indicate that the valve is connected to the node.

Valve ID (8 bits)

This field is used to indicate the Valve ID if the sending node requests the information about a zone valve.

CC:006B.01.09.11.006 This field MUST be set to a Valve ID supported by the receiving node.

CC:006B.01.09.11.007 If the Main Valve field is set to 1, this field MUST be set to 1. Other values MUST be ignored by a receiving node.

Nominal Current (8 bits)

This field is used to advertise the valve's nominal electric current when the valve is On / Open.

CC:006B.01.09.11.008 This field MUST be expressed as a multiple of 10mA. E.g. the value 23 represents 230 mA.

CC:006B.01.09.11.009 This field MUST be set to 0 if the Connected field is set to 0.

CC:006B.01.09.11.00A This field MUST advertise the last measured current (measured when the valve was on) when the valve is Off / Closed.

Valve Error Status (8 bits)

CC:006B.01.09.11.00B This bit mask provides valve error status fields. The field MUST be encoded according the following:

- Bit 0 indicates short circuit has been detected.
- Bit 1 indicates current high threshold has been detected.
- Bit 2 indicates current low threshold has been detected.
- Bit 3 indicates maximum flow has been detected (zone valves only).
- Bit 4 indicates flow high threshold has been detected (zone valves only).
- Bit 5 indicates flow low threshold has been detected (zone valves only).
- Bits 6-7 are reserved.

Any inactive errors can be cleared by sending an Irrigation Valve Config Set Command. Also, if the valve is turned on again, any inactive errors should be cleared as the valve is turned on.

2.2.51.13 Irrigation Valve Config Set Command

This command allows an irrigation valve to be configured accordingly.

Table 2.319: Irrigation Valve Config Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_CONFIG_SET (0x0A)							
Reserved							Main Valve
Valve ID							
Nominal Current High Threshold							
Nominal Current Low Threshold							
Maximum Flow Precision			Maximum Flow Scale=l/h		Maximum Flow Size		
Maximum Flow Value 1							
...							
Maximum Flow Value N							
Flow High Threshold Precision			Flow High Threshold Scale=l/h		Flow High Threshold Size		
Flow High Threshold Value 1							
...							
Flow High Threshold Value N							
Flow Low Threshold Precision			Flow Low Threshold Scale=l/h		Flow Low Threshold Size		
Flow Low Threshold Value 1							
...							
Flow Low Threshold Value N							
Sensor Usage							

Reserved

CC:006B.01.0A.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Main Valve (1 bit)

This field is used to indicate whether the valve to configure is the main valve or a zone valve.

CC:006B.01.0A.11.002 The value 1 MUST indicate that the valve to configure is the main valve.

CC:006B.01.0A.11.003 The value 0 MUST indicate that the valve to configure is the zone valve corresponding to Valve ID field.

Valve ID (8 bits)

This field is used to indicate the Valve ID to configure.

CC:006B.01.0A.11.004 This field MUST be set to a Valve ID supported by the receiving node.

CC:006B.01.0A.11.005 If the Main Valve field is set to 1, this field MUST be set to 1. Other values MUST be ignored by a receiving node.

Nominal Current High Threshold (8 bits)

This field is used to configure the nominal current high threshold for the actual valve.

CC:006B.01.0A.11.006 The receiving node MUST issue an Irrigation System Status Report Command with the System Error Status indicating that a valve is reporting errors when the current crosses above this threshold.

CC:006B.01.0A.11.007 This field MUST be expressed as a multiple of 10mA. E.g. the value 23 represents 230 mA.

Nominal Current Low Threshold (8 bits)

This field is used to configure the nominal current low threshold for the actual valve at the receiving node.

CC:006B.01.0A.11.008 The receiving node MUST issue an Irrigation System Status Report Command with the System Error Status indicating that a valve is reporting errors when the current crosses below this threshold.

CC:006B.01.0A.11.009 This field MUST be expressed as a multiple of 10mA. E.g. the value 23 represents 230 mA.

Maximum Flow

These fields are used to configure the maximum allowed water flow for the specified valve.

CC:006B.01.0A.11.00A These fields MUST be ignored if the Main Valve field is set to 1.

CC:006B.01.0A.13.001 A receiving node having no support for flow sensor MAY ignore these fields.

CC:006B.01.0A.11.00B If the receiving node supports a flow sensor, the node MUST issue an Irrigation System Status Report Command with the System Error Status indicating that a valve is reporting errors when the water flow crosses above this threshold.

CC:006B.01.0A.11.00C This field MUST be expressed in l/h (liter/hour).

Flow High Threshold

These fields are used to configure the flow high threshold for the specified valve.

CC:006B.01.0A.11.00D These fields MUST be ignored if the Main Valve field is set to 1.

CC:006B.01.0A.13.002 A receiving node having no support for flow sensor MAY ignore these fields.

CC:006B.01.0A.11.00E If the receiving node supports a flow sensor, the node MUST issue an Irrigation System Status Report Command with the System Error Status indicating that a valve is reporting errors when the water flow crosses above this threshold.

CC:006B.01.0A.11.00F The format of the Flow High Threshold fields MUST comply with the format used by the Multilevel Sensor Command Class and MUST be expressed in l/h (liter/hour).

Flow Low Threshold

These fields are used to configure the flow low threshold for the specified valve.

CC:006B.01.0A.11.010 These fields MUST be ignored if the Main Valve field is set to 1.

CC:006B.01.0A.13.003 A receiving node having no support for flow sensor MAY ignore these fields.

CC:006B.01.0A.11.011 If the receiving node supports a flow sensor, the node MUST issue an Irrigation System Status Report Command with the System Error Status indicating that a valve is reporting errors when the water flow crosses below this threshold.

CC:006B.01.0A.11.012 The format of the Flow Low Threshold fields MUST comply with the format used by the Multilevel Sensor Command Class and MUST be expressed in l/h (liter/hour).

Sensor Usage (8 bits)

This field is used to configure if the actual valve must turn off / close when the specified sensors are active.

CC:006B.01.0A.11.013 This field MUST be ignored if the Main Valve field is set to 1.

CC:006B.01.0A.11.014 This field MUST be treated as a bit mask and MUST comply with Table 2.320, Irrigation Valve Config Set::Sensor Usage.

Table 2.320: Irrigation Valve Config Set::Sensor Usage

Bit	Description
0	Use Rain Sensor The valve MUST turn off / close if rain is detected A receiving node having no support for rain sensor MAY ignore this field.
1	Use Moisture Sensor The valve MUST turn off / close if moisture is detected A receiving node having no support for moisture sensor MAY ignore this field.
2..7	*Reserved*

CC:006B.01.0A.11.015 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.51.14 Irrigation Valve Config Get Command

This command is used to request the current configuration of an irrigation valve.

CC:006B.01.0B.11.001 The Irrigation Valve Config Report Command MUST be returned in response to this command.

CC:006B.01.0B.11.002 This command MUST NOT be issued via multicast addressing.

CC:006B.01.0B.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.321: Irrigation Valve Config Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_CONFIG_GET (0x0B)							
Reserved							Main Valve
Valve ID							

Reserved

CC:006B.01.0B.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Main Valve (1 bit)

This field is used to indicate whether the sending node requests the configuration of the main valve or of a zone valve.

CC:006B.01.0B.11.005 The value 1 MUST indicate that the sending node requests the main valve configuration.

CC:006B.01.0B.11.006 The value 0 MUST indicate that the sending node requests the configuration related to the zone valve corresponding to Valve ID field.

Valve ID (8 bits)

This field is used to indicate the Valve ID if the sending node requests the information about a zone valve.

CC:006B.01.0B.11.007 This field MUST be set to a Valve ID supported by the receiving node. A node receiving this command
CC:006B.01.0B.12.001 for an unsupported zone valve ID SHOULD return a report for the first supported zone valve.

CC:006B.01.0B.11.008

If the Main Valve field is set to 1, this field MUST be set to 1. Other values MUST be ignored by a receiving node.

2.2.51.15 Irrigation Valve Config Report Command

This command is used to advertise the configuration of a valve.

Table 2.322: Irrigation Valve Config Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_CONFIG_REPORT (0x0C)							
Reserved							Main Valve
Valve ID							
Nominal Current High Threshold							
Nominal Current Low Threshold							
Maximum Flow Precision			Maximum Flow Scale=l/h		Maximum Flow Size		
Maximum Flow Value 1							
...							
Maximum Flow Value N							
Flow High Threshold Precision			Flow High Threshold Scale=l/h		Flow High Threshold Size		
Flow High Threshold Value 1							
...							
Flow High Threshold Value N							
Flow Low Threshold Precision			Flow Low Threshold Scale=l/h		Flow Low Threshold Size		
Flow Low Threshold Value 1							
...							
Flow Low Threshold Value N							
Sensor Usage							

For fields' description, refer to Section 2.2.51.13 Irrigation Valve Config Set Command.

CC:006B.01.0C.11.001 A sending node MUST comply with field descriptions in Section 2.2.51.13 Irrigation Valve Config Set Command.

2.2.51.16 Irrigation Valve Run Command

The Irrigation Valve Run Command will run the specified valve for a specified duration.

Table 2.323: Irrigation Valve Run Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_RUN (0x0D)							
Reserved							Main Valve
Valve ID							
Duration MSB							
Duration LSB							

Reserved

CC:006B.01.0D.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Main Valve (1 bit)

This field is used to indicate if the specified valve is the main valve or a zone valve.

CC:006B.01.0D.11.002 The value 1 MUST indicate that the specified valve is the main valve.

CC:006B.01.0D.11.003 The value 0 MUST indicate that the specified valve is the zone valve corresponding to Valve ID field.

Valve ID (8 bits)

This field is used to specify the actual Valve ID.

CC:006B.01.0D.11.004 This field MUST be set to a Valve ID supported by the receiving node.

CC:006B.01.0D.11.005 If the Main Valve field is set to 1, this field MUST be set to 1. Other values MUST be ignored by a receiving node.

Duration (16 bits)

This field is used to specify the duration of the run in seconds.

CC:006B.01.0D.11.006 The value 0 MUST indicate that the valve MUST be turned off / Closed immediately.

2.2.51.17 Irrigation Valve Table Set Command

This command is used to set a valve table with a list of valves and durations.

Table 2.324: Irrigation Valve Table Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_TABLE_SET (0x0E)							
Valve Table ID							
Valve ID 1							
Duration MSB 1							
Duration LSB 1							
...							
Valve ID N							
Duration MSB N							
Duration LSB N							

Valve Table ID (8 bits)

This field is used to specify the valve table ID.

CC:006B.01.0E.11.001 This field MUST be set to a Valve Table ID supported by the receiving node.

Valve ID and Duration (N * 3 bytes)

These fields are used to specify valve IDs and their associated run duration.

CC:006B.01.0E.11.002 A sending node MUST NOT specify more Valve IDs than the “Valve Table Max Size” advertised by a receiving node in the Irrigation System Info Report Command

CC:006B.01.0E.11.003 The duration MUST be expressed in seconds.

CC:006B.01.0E.13.001 This field MAY be omitted in order to erase/empty the specified valve table.

2.2.51.18 Irrigation Valve Table Get Command

This command is used to request the contents of the specified Valve Table ID.

CC:006B.01.0F.11.001 The Irrigation Valve Table Report Command MUST be returned in response to this command.

CC:006B.01.0F.11.002 This command MUST NOT be issued via multicast addressing.

CC:006B.01.0F.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.325: Irrigation Valve Table Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_TABLE_GET (0x0F)							
Valve Table ID							

Valve Table ID (8 bits)

This field is used to specify the valve table ID.

CC:006B.01.0F.11.004 Valves tables MUST be identified sequentially from 1 to the total number available on the device.

2.2.51.19 Irrigation Valve Table Report Command

This command provides the contents of the specified Valve Table ID.

Table 2.326: Irrigation Valve Table Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_TABLE_REPORT (0x10)							
Valve Table ID							
Valve ID 1							
Duration MSB 1							
Duration LSB 1							
...							
Valve ID N							
Duration MSB N							
Duration LSB N							

For fields' description, refer to Section 2.2.51.17 Irrigation Valve Table Set Command.

CC:006B.01.10.11.001 A sending node MUST comply with field descriptions in Section 2.2.51.17 Irrigation Valve Table Set Command.

2.2.51.20 Irrigation Valve Table Run Command

This command is used to run the specified valve tables sequentially.

Table 2.327: Irrigation Valve Table Run Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_VALVE_TABLE_RUN (0x11)							
Valve Table ID 1							
...							
Valve Table ID N							

Valve Table ID (N bytes)

This field is used to specify the list of Valve Tables to run sequentially.

CC:006B.01.11.11.001 A receiving node MUST run the indicated Valve Table ID starting by the entry 1. These fields contain a variable list of valve table IDs that will be run sequentially in the order they are listed.

2.2.51.21 Irrigation System Shutoff Command

This command is used to prevent any irrigation activity triggered by the Schedule CC for a specified duration.

Table 2.328: Irrigation System Shutoff Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION (0x6B)							
Command = IRRIGATION_SYSTEM_SHUTOFF (0x12)							
Duration							

Duration (8 bits)

This field is used to indicate the duration of the system shutoff.

CC:006B.01.12.11.001 Values in the range 1..254 MUST indicate how many hours the irrigation system MUST stay shut
CC:006B.01.12.11.002 off after reception of this command. While active, the shutoff duration MUST prevent all enabled
CC:006B.01.12.11.003 schedules (including the back-up schedule) from running. After the shutoff duration is over, any enabled schedule MUST run as configured.

CC:006B.01.12.11.004 The value 0 MUST indicate to turn off any running valve (including the main valve) as well as cancel
CC:006B.01.12.11.005 any active Irrigation Valve Table Run or Schedule. Any subsequent schedule MUST run as configured.

CC:006B.01.12.11.006 The value 255 MUST indicate that the irrigation system MUST stay permanently shut off until the node receives one of the following commands:

- Irrigation System Shutoff Command with a Duration different than 255.
- Irrigation Valve Run Command
- Irrigation Valve Table Run Command

CC:006B.01.12.11.007 When permanently shut off, the node MUST NOT run any schedule

CC:006B.01.12.11.008 A receiving node MUST cancel the current shutoff and advertise a shutoff duration of 0 in the Irrigation System Status Report Command after receiving one of the following commands:

- Irrigation Valve Run Command
- Irrigation Valve Table Run Command

2.2.52 Language Command Class, version 1

The Language Command Class is used to specify the language settings on a device.

2.2.52.1 Language Set Command

This command is used to set the language at the receiving node. The receiving node SHOULD use the default language in case the received language is not supported.

Table 2.329: Language Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LANGUAGE (0x89)							
Command = LANGUAGE_SET (0x01)							
Language 1							
Language 2							
Language 3							
Country 1							
Country 2							

Language (24 bits)

The code definition of the languages may be found in ISO 639-2:1998 'Codes for the representation of names of languages - Part 2: Alpha-3 code'. In the table below are some examples of the alpha-3 codes listed:

Table 2.330: Example ISO 639-2:1998 alpha-3 codes

Language	Language 1	Language 2	Language 3
English	e	n	g
Flemish; Dutch	d	u	t
French	f	r	e
German	g	e	r
Italian	i	t	a
Polish	p	o	l
Russian	r	u	s
Walloon	w	l	n

Country (16 bits) The code definition of the countries may be found in ISO 3166-1 'Country Codes: Alpha-2 codes''. The use of the country field is OPTIONAL and only defined in case it is necessary for distinguishing geographical variants. The number of data fields transmitted MUST be determined from the length field in the frame. In the table below are some examples of the alpha-2 codes listed:

Table 2.331: Example ISO 3166-1 alpha-2 codes

Country	Country 1	Country 2
Belgium	B	E
Italy	I	T
Netherlands	N	L
Poland	P	L
United Kingdom	G	B
United States	U	S

2.2.52.2 Language Get Command

This command is used to request the current language setting in a device.

The Language Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.332: Language Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LANGUAGE (0x89)							
Command = LANGUAGE_GET (0x02)							

2.2.52.3 Language Report Command

This command returns the current language setting in a device.

Table 2.333: Language Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LANGUAGE (0x89)							
Command = LANGUAGE_REPORT (0x03)							
Language 1							
Language 2							
Language 3							
Country 1							
Country 2							

For fields' description, refer to Section 2.2.52.1 Language Set Command.

2.2.53 Lock Command Class, version 1 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Door Lock Command Class.

If implementing this command class, it is RECOMMENDED that the Door Lock Command Class is also implemented.

The Lock Command Class is used to lock and unlock a "lock" type device, e.g. a door or window lock

2.2.53.1 Lock Set Command

This command is used to set the lock state in a device.

Table 2.334: Lock Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LOCK (0x76)							
Command = LOCK_SET (0x01)							
Lock State							

Lock State (8 bits)

The lock state field used to set the lock state of the device. The value 0 indicates that the device is unlocked. The value 1 indicates that the device is locked.

2.2.53.2 Lock Get Command

This command is used to request the lock state from a device.

The Lock Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.335: Lock Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LOCK (0x76)							
Command = LOCK_GET (0x02)							

2.2.53.3 Lock Report Command

This command is used to report the lock state of a device.

Table 2.336: Lock Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LOCK (0x76)							
Command = LOCK_REPORT (0x03)							
Lock State							

Lock State (8 bits)

The Lock state field used to report the lock state of the device. The value 0 indicates that the device is unlocked. The value 1 indicates that the device is locked.

2.2.54 Manufacturer Proprietary Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST NOT support this Command Class. Consult the Z-Wave Alliance when your application does not seem to fit any existing Command Class.

The Manufacturer Proprietary Command Class is used to transfer data between devices in the Z-Wave network. The data content MUST be vendor specific and MUST be non-value added with respect to the Home Automation application in general. An example could be data used to diagnose the hardware in a device.

Note: Do not use the Manufacturer Proprietary Command Class without written approval from the Z-Wave Alliance.

2.2.54.1 Manufacturer Proprietary Command

This command is used to transfer proprietary commands between devices. The Command features a manufacturer specific identifier to allow the receiving device to check if this Command can be interpreted. The vendor is responsible for establishing a Command structure to differ between the set of Commands supported.

In order not to congest the Z-Wave network, large data transfers MUST leave transmit opportunities for other nodes in the network. If sending a command longer than two frames, a node MUST implement a delay between every transmitted frame. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 2 frames back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

Table 2.337: Manufacturer Proprietary Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_PROPRIETARY (0x91)							
Manufacturer ID 1							
Manufacturer ID 2							
Data 1							
...							
Data N							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device. Manufacturer identifiers can be found in [32].

The first byte (Manufacturer ID 1) is the most significant byte.

Data (N bytes)

The data fields may be used for data transfer etc. The number of data fields transmitted MUST be determined from the length field in the frame.

2.2.55 Meter Command Class, version 1

The Meter Command Class is used to advertise instantaneous and accumulated numerical readings.

The Command Class is intended for accumulated values in physical units from a water meter or metering device (gas, electric etc.) and thereby enabling some automatic meter reading capabilities.

2.2.55.1 Terminology

A meter is used to monitor a resource. The meter **accumulates** the resource flow over time. As an option, the meter may report not only the most recent accumulated reading but also the previous reading and the time that elapsed since then. A meter may also be able to report the current resource flow. This is known as the **instant** value.

Meters may report two rate types. A utility installs **production** meters in its production facilities, while it installs **consumption** meters at consumer premises. The difference is the mapping of the physical flow direction to the **forward direction** of the meter:

- The accumulated value of a production meter grows when more resources are produced
- The accumulated value of a production meter drops when more resources are consumed
- The accumulated value of a consumption meter grows when more resources are consumed
- The accumulated value of a consumption meter drops when more resources are produced

If a meter only supports one rate type, the meter may **run backwards**. If a meter runs backwards, the accumulated value may eventually become **negative**. While the production meter of a power generator rarely runs backwards, the consumption meter in a private household with solar panels runs backwards every time the solar panels deliver more than is consumed in the household. The actual reading of a one-rate type meter reflects the net accumulated value since the meter was installed with a factory default reading of zero. The accumulated value over an arbitrary interval may be calculated by subtracting a previous accumulated value from the current accumulated value. Normal arithmetic rules ensure that this also works in case of negative values.

A production meter may be used as a consumption meter by applying a sign change to all production meter readings before using the values as consumption meter readings.

A meter device may advertise that it implements two separate registers for the production and consumption, respectively. In that case, both of these meters are always running forward: Production makes the production meter run forward while consumption makes the consumption meter run forward. Two meter reports must be used to report the values of the respective meters.

2.2.55.2 Meter Get Command

This command is used to request the current meter reading to a supporting node.

The Meter Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.338: Meter Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_GET (0x01)							

2.2.55.3 Meter Report Command

This command is used to advertise the current meter reading at the sending node.

This command MUST NOT be issued using broadcast addressing.

Table 2.339: Meter Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_REPORT (0x02)							
Meter Type							
Precision			Scale		Size		
Meter Value 1							
...							
Meter Value N							

Meter Type (8 bits)

This field is used to specify what type of metering physical unit is being reported.

This field MUST be encoded according to Table 2.358.

Precision (3 bits)

This field MUST indicate how many decimal places are included in the Meter Value field. For example, the *Meter Value* field set to 1025 and this field set to 2 MUST be interpreted as equal to 10.25.

The value of the precision field MUST be in the range 0..7.

Scale (2 bits)

This field MUST advertise the unit used for the *Meter Value* field.

This field MUST be encoded according to Table 2.359.

Size (3 bits)

This field indicates the length in bytes of the *Meter Value* field. This field MUST be set to 1, 2 or 4.

Meter Value (N bytes)

This field is used to advertise the actual meter reading.

The length of this field in bytes MUST be according to the *Size* field.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12.

2.2.56 Meter Command Class, version 2

The Meter Command Class is used to advertise instantaneous and accumulated numerical readings.

The Command Class is intended for accumulated values in physical units from a water meter or metering device (gas, electric etc.) and thereby enabling some automatic meter reading capabilities.

2.2.56.1 Compatibility Considerations

The Meter Command Class, version 2 introduces the following functionalities:

- Commands to interview supporting nodes Meter capabilities
- New field 'Previous Meter Value' added to the Meter Report indicating consumption since previous report.
- An optional Meter Reset Command for resetting accumulated consumption.
- New meter scales have been added

Commands and fields not mentioned in this version MUST remain unchanged from version 1.

Supporting nodes MAY support several scales. A manufacturer MUST define which scale is the default scale and it MUST be described in the product manual.

2.2.56.2 Meter Supported Get Command

This command is used to request the supported scales in a sub meter.

The Meter Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.340: Meter Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_SUPPORTED_GET (0x03)							

2.2.56.3 Meter Supported Report Command

This command is used to advertise the supported scales and capabilities of the sending node.

Table 2.341: Meter Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_SUPPORTED_REPORT (0x04)							
Meter Reset	Reserved		Meter Type				
Reserved				Scale Supported			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Meter Reset (1 bit)

This field is used to indicate if the sending node supports the Meter Reset Command.

The value 1 MUST indicate that the Meter Reset Command is supported.

The value 0 MUST indicate that the Meter Reset Command is not supported and MUST be ignored by the sending node.

Meter Type (5 bits)

This field is used to advertise the implemented meter type by the sending node.

This field MUST be encoded according to Table 2.358.

Scale Supported (4 bits)

This field is used to advertise the supported scales of the sending node.

The bit value '1' MUST indicate support for the actual scale.

The bit value '0' MUST indicate that there is no support for the actual scale.

This MUST be interpreted as a bitmask in combination with the Meter Type field and MUST be according to Table 2.356.

2.2.56.4 Meter Reset Command

This command is used to reset all accumulated values stored at the receiving node.

Table 2.342: Meter Reset Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_RESET (0x05)							

A supporting node MUST reset all its supported accumulated values when receiving this command if it advertises support for Meter Reset in the Meter Supported Report Command.

A supporting node MUST ignore this command if it advertises no support for Meter Reset in the Meter Supported Report Command

2.2.56.5 Meter Get Command

This command is used to request the current meter reading to a supporting node .

The Meter Report Command MUST be returned in response to this command if the requested scale is supported.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.343: Meter Get Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_GET (0x01)							
Reserved			Scale		Reserved		

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Scale (2 bits)

This field is used to request an actual scale for the meter reading.

This field MUST be encoded according to Table 2.359.

If this field is not present or set to 0x00, a supporting node MUST return a report using its default scale.

A supporting node MUST ignored this command if this field is set to a non-supported scale.

For improved compatibility with version 1 controlling nodes, a supporting node receiving this command without this field (version 1 format) SHOULD either return a version 1 Report Command or a Report Command with the *Rate Type* field set to 0x00

2.2.56.6 Meter Report Command

This command is used to advertise the current meter reading at the sending node.

This command MUST NOT be issued using broadcast addressing.

Table 2.344: Meter Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_REPORT (0x02)							
Res	Rate Type		Meter Type				
Precision			Scale		Size		
Meter Value 1							
...							
Meter Value N							
Delta Time 1							
Delta Time 2							
Previous Meter Value 1 (optional)							
...							
Previous Meter Value N (optional)							

Fields not described below MUST remain unchanged from version 1.

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Type (2 bits)

This field is used to indicate if the actual reading advertises import or export values.

The Import value for a meter reading MUST indicate that the reading represents a consumed amount.

The Export value for a meter reading MUST indicate that the reading represents a produced amount.

This field MUST be encoded according to Table 2.345.

Table 2.345: Meter Report::Rate Type encoding

Value	Rate Type	Version
0x00	Unspecified	1
0x01	Import (consumed)	2
0x02	Export (produced)	2
0x03	Reserved	-

Meter Type (5 bits)

This field is used to specify what type of metering physical unit is being reported.

This field MUST be encoded according to Table 2.358.

Precision (3 bits)

This field MUST indicate how many decimal places are included in the *Meter Value* and *Previous Meter Value* fields. For example, the *Meter Value* set to 1025 and this field set to 1 MUST be interpreted as equal to 102.5.

The value of the precision field MUST be in the range 0..7.

Scale (2 bits)

This field MUST advertise the unit used for the *Meter Value* and *Previous Meter Value* fields.

This field MUST be encoded according to Table 2.359.

Size (3 bits)

This field MUST indicate the length in bytes of the *Meter Value* and *Previous Meter Value* fields.

This field MUST be set to 1, 2 or 4.

Meter Value (N bytes)

This field is used to advertise the actual meter reading.

The length of this field in bytes MUST be according to the *Size* field.

The first byte MUST be the most significant byte.

The field MUST be encoded using signed representation and comply with Table 2.12.

Delta Time (16 bits)

This field MUST advertise the elapsed time in seconds between the readings advertised by the *Meter Value* and the *Previous Meter Value* fields. This field MUST be according to Table 2.346.

Table 2.346: Meter Report v2::Delta Time encoding

Value	Description
0x0000	No Previous Meter Value field is included in this command
0x0001..0xFFFE	1..65534 seconds have elapsed between reporting the Previous Meter Value reading and the current meter reading
0xFFFF	Unknown elapsed time between the Previous Meter Value reading and the current meter reading

Previous Meter Value (N bytes)

This field is used to advertise the last issued meter reading for the actual scale.

This field MUST be omitted if the *Delta Time* field is set to the value 0.

The length of this field in bytes MUST be according to the *Size* field if the *Delta Time* field is set to a non-zero value.

The length of this field in bytes MUST be according to the *Size* field.

The first byte MUST be the most significant byte.

The field MUST be encoded using signed representation and comply with Table 2.12.

2.2.57 Meter Command Class, version 3

The Meter Command Class is used to advertise instantaneous and accumulated numerical readings.

The Command Class is intended for accumulated values in physical units from a water meter or metering device (gas, electric etc.) and thereby enabling some automatic meter reading capabilities.

2.2.57.1 Compatibility Considerations

The Meter Command Class, version 3 introduces the following functionalities:

- New meter scales have been added.

Commands and fields not mentioned in this version MUST remain unchanged from version 2.

The manufacturer MUST define which scale is the default scale and it MUST be described in the product manual.

The default scale MUST be in the range 0..3 as version 1 and version 2 controlling nodes will not be able to interpret scale values in the range 4..7.

2.2.57.2 Meter Supported Report Command

This command is used to advertise the supported scales and capabilities of the sending node.

Table 2.347: Meter Supported Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_SUPPORTED_REPORT (0x04)							
Meter Reset	Reserved		Meter Type				
Scale Supported							

Fields not described below MUST remain unchanged from version 2.

Scale Supported (8 bits)

This field is used to advertise the supported scales by the sending node.

The field MUST be treated as a bitmask in combination with the *Meter Type* field and MUST be according to Table 2.356.

The bit value '1' MUST indicate support for the actual scale.

The bit value '0' MUST indicate that there is no support for the actual scale.

2.2.57.3 Meter Get Command

This command is used to request the current meter reading to a supporting node.

The Meter Report Command MUST be returned in response to this command if the requested scale is supported.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.348: Meter Get Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_GET (0x01)							
Reserved			Scale		Reserved		

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Scale (3 bits)

This field is used to request an actual scale for the meter reading.

This field MUST be encoded according to Table 2.359.

If this field is not present or set to 0x00, a supporting node MUST return a report using its default scale.

A supporting node MUST ignored this command if this field is set to a non-supported scale.

For improved compatibility with version 1 controlling nodes, a supporting node receiving this command without this field (version 1 format) SHOULD either return a version 1 Report Command or a Report Command with the *Rate Type* field set to 0x00

2.2.57.4 Meter Report command

This command is used to advertise the current meter reading at the sending node.

This command MUST NOT be issued using broadcast addressing.

Table 2.349: Meter Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_REPORT (0x02)							
Scale (2)	Rate Type		Meter Type				
Precision			Scale (1:0)		Size		
Meter Value 1							
...							
Meter Value N							
Delta Time 1							
Delta Time 2							
Previous Meter Value 1 (optional)							
...							
Previous Meter Value N (optional)							

Fields not described below MUST remain unchanged from version 2.

Scale (3 bits)

This field MUST advertise the unit used for the *Meter Value* and *Previous Meter Value* fields.

This field is composed of two sub-fields *Scale (2)* and *Scale (1:0)* which MUST be composed and interpreted as one unit. *Scale (2)* MUST be the most significant bit.

This field MUST be encoded according to Table 2.359.

2.2.58 Meter Command Class, version 4

The Meter Command Class is used to advertise instantaneous and accumulated numerical readings.

The Command Class is intended for accumulated values in physical units from a water meter or metering device (gas, electric etc.) and thereby enabling some automatic meter reading capabilities.

2.2.58.1 Compatibility Considerations

The Meter Command Class, version 4 introduces the following functionalities:

- A supporting node can now support several rate types
- New meter scales have been added

Commands and fields not mentioned in this version MUST remain unchanged from version 3.

The manufacturer MUST define which scale and rate types are the default and it MUST be described in the product manual.

The default scale MUST be in the range 0..3 as version 1 and version 2 controlling nodes will not be able to interpret scale values in the range 4..7.

2.2.58.2 Meter Supported Report Command

This command is used to advertise the supported scales and capabilities of the sending node.

Table 2.350: Meter Supported Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_SUPPORTED_REPORT (0x04)							
Meter Reset	Rate Type		Meter Type				
M.S.T	Scale Supported “Byte 1”						
Number of Scale Supported Bytes to Follow (optional)							
Scale Supported “Byte 2” 1 (optional)							
...							
Scale Supported “Byte N+1” N (optional)							

Fields not described below MUST remain unchanged from version 3.

Rate Type (2 bits)

This field is used to indicate the supported rate types by the sending node.

This field MUST be encoded according to Table 2.351.

Table 2.351: Meter Supported Report::Rate Type encoding

Value	Supported rate types	Version
0x00	Reserved	1..3
0x01	Import only (consumed)	4
0x02	Export only (produced)	4
0x03	Both Import and Export	4

Scale Supported “Byte 1” (7 bits)

This field is used to advertise the supported scales by the sending node.

The field MUST be treated as a bitmask in combination with the *Meter Type* field and MUST be according to Table 2.356.

The bit value ‘1’ MUST indicate support for the actual scale.

The bit value '0' MUST indicate that there is no support for the actual scale.

M.S.T. (More Scale Types) (1 bit)

This field is used to indicate if the *Number of Scale Supported Bytes to Follow* field is included in the command.

The value 0 MUST indicate that the *Number of Scale Supported Bytes to Follow* and subsequent optional *Scale Supported* fields are omitted from the command.

The value 1 MUST indicate that the Number of *Scale Supported Bytes to Follow* field is present in the command.

Number of Scale Supported to Follow (8 bits)

This field MUST indicate the length in bytes of the subsequent *Scale Supported (bytes 2..N+1)* field.

This field MUST be omitted if the *M.S.T.* field is set to 0.

Scale Supported (bytes 2..N+1) (N bytes)

This field is used to advertise the supported scales by the sending node.

The length of this field in bytes MUST be according to the *Number of Scale Supported Bytes to Follow* field.

This field MUST be omitted if the *M.S.T.* field is set to 0 or if the *Number of Scale Supported Bytes to Follow* field is set to 0.

The field MUST be treated as a bitmask in combination with the *Meter Type* field and MUST be according to Table 2.356.

The bit value '1' MUST indicate support for the actual scale.

The bit value '0' MUST indicate that there is no support for the actual scale.

2.2.58.3 Meter Get Command

This command is used to request the current meter reading to a supporting node.

The Meter Report Command MUST be returned in response to this command if the requested scale and rate type are supported.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.352: Meter Get Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_GET (0x01)							
Rate Type		Scale			Reserved		
Scale 2							

Fields not described below MUST remain unchanged from version 3.

Rate Type (2 bits)

This field is used to request a rate type for the meter reading.

This field MUST be encoded according to Table 2.353.

Table 2.353: Meter Get::Rate Type encoding

Value	Description	Version
0x00	Default	1
0x01	Import (consumed)	4
0x02	Export (produced)	4
0x03	Reserved	-

If supported, a receiving node MUST return a report for the requested Rate Type.

If this field is not present or set to 0x00, a receiving node MUST return a Report using its default Rate Type.

Scale (3 bits)

This field is used to request an actual scale for the meter reading.

This field MUST be encoded according to Table 2.359.

If this field is not present or set to 0x00, a supporting node MUST return a report using its default scale.

A supporting node MUST ignored this command if this field is set to a non-supported scale.

For improved compatibility with version 1 controlling nodes, a supporting node receiving this command without this field (version 1 format) SHOULD either return a version 1 Report Command or a Report Command with the *Rate Type* field set to 0x00

The value 7 MUST indicate that the scale of the requested meter reading is advertised by the *Scale 2* field.

Scale 2 (8 bits)

This field is used to request an actual scale for the meter reading.

This field MUST be encoded according to Table 2.359.

This field MUST be present in the command if the *Scale* field is set to 7 (M.S.T).

This field MUST be omitted if the *Scale* field is set to a different value than 7.

2.2.58.4 Meter Report Command

This command is used to advertise the current meter reading at the sending node.

This command MUST NOT be issued using broadcast addressing.

Table 2.354: Meter Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_REPORT (0x02)							
Scale (2)	Rate Type		Meter Type				
Precision			Scale (1:0)		Size		
Meter Value 1							
...							
Meter Value N							
Delta Time 1							
Delta Time 2							
Previous Meter Value 1 (optional)							
...							
Previous Meter Value N (optional)							
Scale 2							

Fields not described below MUST remain unchanged from version 3.

Scale (3 bits)

This field MUST advertise the unit used for the *Meter Value* and Previous* Meter Value fields.

This field is composed of two sub-fields *Scale (2)* and *Scale (1:0)* which MUST be composed and interpreted as one unit. *Scale (2)* MUST be the most significant bit.

This field MUST be encoded according to Table 2.359.

The value 7 MUST indicate that the scale of the advertised meter reading is advertised by the *Scale 2* field.

Scale 2 (8 bits)

This field is used to advertise the unit used for the *Meter Value* and *Previous Meter Value* fields.

This field MUST be present in the command if the *Scale* field is set to 7 (M.S.T).

This field MUST be omitted if the *Scale* field is set to a different value than 7.

This field MUST be encoded according to Table 2.359

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.59 Meter Command Class, version 5

The Meter Command Class is used to advertise instantaneous and accumulated numerical readings.

The Command Class is intended for accumulated values in physical units from a water meter or metering device (gas, electric etc.) and thereby enabling some automatic meter reading capabilities.

2.2.59.1 Compatibility considerations

The Meter Command Class, version 5 adds support for Heating and Cooling Meter types.

The Meter Command Class, version 5 is backwards compatible with version 4.

Commands and fields not described in this version MUST remain unchanged from version 4.

2.2.59.2 Meter Supported Report Command

This command is used to advertise supported scales and capabilities of the sending.

Table 2.355: Meter Supported Report Command, version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_SUPPORTED_REPORT (0x04)							
Meter Reset	Rate Type		Meter Type				
M.S.T	Scale Supported “Byte 1”						
Number of Scale Supported Bytes to Follow (Version 4 Extension)							
Scale Supported 1 “Byte 2”							
...							
Scale Supported N “Byte N+1”							

All fields not described below MUST remain unchanged from version 4.

Table 2.356: Meter Supported Report::Scale Supported Bitmask encoding

Meter Type	Scale	Scale Supported Byte Number	Scale Supported Bit position	Measurement mode	Version
Electric Meter	kWh	Byte 1	Bit 0	Accumulated	1
	kVAh	Byte 1	Bit 1	Accumulated	2
	W	Byte 1	Bit 2	Instant	2
	Pulse count	Byte 1	Bit 3	Accumulated	2
	V	Byte 1	Bit 4	Instant	3
	A	Byte 1	Bit 5	Instant	3
	Power factor	Byte 1	Bit 6	Instant	3
	kVar	Byte 2	Bit 0	Instant	4
	kVarh	Byte 2	Bit 1	Accumulated	4
	Reserved	Byte 2	Bit 2-7	Reserved	-
Gas meter	Cubic meters	Byte 1	Bit 0	Accumulated	1
	Cubic feet	Byte 1	Bit 1	Accumulated	2
	Reserved	Byte 1	Bit 2	n/a	-
	Pulse count	Byte 1	Bit 3	Accumulated	2
	Reserved	Byte 1	Bit 4-6	Reserved	-
	Reserved	Byte 2	Bit 0-7	Reserved	-
Water meter	Cubic meters	Byte 1	Bit 0	Accumulated	1
	Cubic feet	Byte 1	Bit 1	Accumulated	1
	US gallons	Byte 1	Bit 2	Accumulated	1
	Pulse count	Byte 1	Bit 3	Accumulated	2
	Reserved	Byte 1	Bit 4-6	Reserved	-
	Reserved	Byte 2	Bit 0-7	Reserved	-
Heating meter	kWh	Byte 1	Bit 0	Accumulated	5
	Reserved	Byte 1	Bit 1-6	Reserved	-
	Reserved	Byte 2	Bit 0-7	Reserved	-
Cooling	kWh	Byte 1	Bit 0	Accumulated	5
	Reserved	Byte 1	Bit 1-6	Reserved	-
	Reserved	Byte 2	Bit 0-7	Reserved	-

2.2.59.3 Meter Report Command

This command is used to advertise the current meter reading at the sending node.

This command MUST NOT be issued using broadcast addressing.

Table 2.357: Meter Report Command, version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_REPORT (0x02)							
Scale (2)	Rate Type		Meter Type				
Precision			Scale (1:0)		Size		
Meter Value 1							
...							
Meter Value N							
Delta Time 1							
Delta Time 2							
Previous Meter Value 1 (optional)							
...							
Previous Meter Value N (optional)							
Scale 2							

Fields not described below MUST remain unchanged from version 4.

Meter Type (5 bits)

This field is used to specify what type of metering physical unit is being reported.

This field MUST be encoded according to Table 2.358.

Table 2.358: Meter Report::Meter Type encoding

Value	Meter Type	Version
0x01	Electric meter	1
0x02	Gas meter	1
0x03	Water meter	1
0x04	Heating meter	5
0x05	Cooling meter	5

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Scale (3 bits)

This field MUST advertise the unit used for the *Meter Value* and *Previous Meter Value* fields . This field is composed of two sub-fields *Scale (2)* and *Scale (1:0)*, which MUST be composed and interpreted as one unit. *Scale (2)* MUST be the most significant bit.

This field MUST be encoded according to Table 2.359.

The value 0x07 MUST indicate that the scale of the advertised meter reading is advertised through the *Scale 2* field.

Table 2.359: Meter Report::Scale / Scale 2 encoding

Meter Type	Scale (0:2) Value	Scale 2 Value	Unit	Version
Electric meter	0x00	0x00 (See Scale 0:2)	kWh	1
	0x01		kVAh	2
	0x02		W	2
	0x03		Pulse count	2
	0x04		V	3
	0x05		A	3
	0x06		Power Factor	3
	0x07 (See Scale 2)	0x00	kVar	4
		0x01	kVarh	4
Gas meter	0x00	0x00 (See Scale 0:2)	Cubic meters	1
	0x01		Cubic feet	2
	0x02		Reserved	-
	0x03		Pulse count	2
	0x04..0x06		Reserved	-
Water meter	0x00	0x00 (See Scale 0:2)	Cubic meters	1
	0x01		Cubic feet	1
	0x02		US Gallons	1
	0x03		Pulse count	2
	0x04..0x06		Reserved	-
Heating meter	0x00	0x00 (See Scale 0:2)	kWh	5
Cooling meter	0x00	0x00 (See Scale 0:2)	kWh	5

2.2.60 Meter Command Class, version 6

The Meter Command Class is used to advertise instantaneous and accumulated numerical readings.

The Command Class is intended for accumulated values in physical units from a water meter or metering device (gas, electric etc.) and thereby enabling some automatic meter reading capabilities.

2.2.60.1 Compatibility Considerations

The Meter Command Class, version 6 adds a functionality to reset the accumulated value stored at receiving node to specific.values.

The Meter Command Class, version 6 is backwards compatible with version 5.

Commands and fields not described in this version MUST remain unchanged from version 5.

2.2.60.2 Meter Reset Command

This command is used to reset the accumulated values stored at the receiving node.

If the node advertises support for Meter Reset in the Meter Supported Report Command, the supporting node MUST reset the accumulated values to specific values, instead of setting to zero, when receiving this command.

A supporting node MUST ignore this command if it advertises no support for Meter Reset in the Meter Supported Report Command.

A supporting node MUST ignore this command if it receives it for a non-supported Meter Type, Rate Type or Scale combination.

Table 2.360: Meter Reset Command, version 6

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER (0x32)							
Command = METER_RESET (0x05)							
Scale (2)	Rate Type		Meter Type				
Precision			Scale (1:0)		Size		
Meter Value 1							
...							
Meter Value N							
Scale 2							

For fields description, refer to Section 2.2.59.3 Meter Report Command.

2.2.61 Meter Table Configuration Command Class, version 1

The Meter Table Configuration Command Class defines the Commands necessary to configure the fundamental properties of the meter.

The Meter Table configuration commands are separated from the Meter Table monitoring commands in the Meter Table Monitor Command Class, allowing the classes to be optionally supported at different Z-Wave security levels. (E.g. Meter table monitoring commands could be supported in any device, while enabling a strict and certificate based security solution for the Meter Table Configuration Command class). Refer to Section 4 for more details about Z-Wave security.

2.2.61.1 Meter Table Point Adm Number Set Command

This command is used to set the Meter Point Administration Number in the metering device. The Meter Point Administration Number is used to identify the customer.

Table 2.361: Meter Table Point Adm Number Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_CONFIG (0x3C)							
Command = METER_TBL_TABLE_POINT_ADM_NO_SET (0x01)							
Reserved			Number of Meter Point Adm Number Characters				
Meter Point Adm Number Character 1							
...							
Meter Point Adm Number Character N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Meter Point Adm Number Characters (5 bits)

Number of characters in the meter point administration number(1…32).

Meter Point Adm Number Character (N bytes)

The Meter Point Adm Number character fields hold the string identifying the customer. The character presentation uses standard ASCII codes (values 128-255 are ignored).

2.2.62 Meter Table Monitor Command Class, version 1

The Meter Table Monitor Command Class defines the Commands necessary to read historical and accumulated values in physical units from a water meter or other metering device (gas, electric etc.) and thereby enabling automatic meter reading capabilities

2.2.62.1 Meter Table Point Adm. Number Get Command

This command is used to request the Meter Point Administration Number to identify customer.

The Meter Table Adm. Number Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.362: Meter Table Point Adm Number Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_POINT_ADM_NO_GET (0x01)							

2.2.62.2 Meter Table Point Adm. Number Report Command

This command reports parameters used for identification of customer and metering device.

Table 2.363: Meter Table Point Adm Number Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_POINT_ADM_NO_REPORT (0x02)							
Reserved			Number of Meter Point Adm. Number Characters				
Meter Point Adm. Number Character 1							
...							
Meter Point Adm. Number Character N							

Fields MUST be according to descriptions in the Meter Table Point Adm Number Set Command from the Meter Table Configuration Command Class, version 1.

2.2.62.3 Meter Table ID Get Command

This command is used to request the parameters used for identification of customer and metering device.

The Meter Table ID Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.364: Meter Table ID Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_ID_GET (0x03)							

2.2.62.4 Meter Table ID Report Command

This command reports parameters used for identification of customer and metering device.

Table 2.365: Meter Table ID Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_ID_REPORT (0x04)							
Reserved			Number of Meter ID Characters				
Meter ID Character 1							
...							
Meter ID Character N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Meter ID Characters (5 bits)

This field is used to indicate the length of the Meter ID Character field in bytes.

This field MUST be in the range 1..32.

Meter ID Character (N bytes)

This field is used to identify the individual metering device.

The length of this field in bytes MUST be according to the *Number of Meter ID Characters* field.

Each byte of this field MUST be encoded with ASCII representation and be in the range 0x00..0x7F.

A controlling node can use the Manufacturer Specific Command Class in conjunction for product identification.

2.2.62.5 Meter Table Capability Get Command

This command is used to request the meter table capabilities of a supporting device.

The Meter Table Capability Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.366: Meter Table Capability Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_CAPABILITY_GET (0x05)							

2.2.62.6 Meter Table Capability Report Command

This command is used to advertise meter table capabilities of the sending node.

Table 2.367: Meter Table Capability Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_CAPABILITY_REPORT (0x06)							
Rate Type		Meter Type					
Reserved				Pay Meter			
Dataset Supported 1							
Dataset Supported 2							
Dataset Supported 3							
Dataset History Supported 1							
Dataset History Supported 2							
Dataset History Supported 3							
Data History Supported 1							
Data History Supported 2							
Data History Supported 3							

Rate Type (2 bits)

This field is used to indicate if the actual reading advertises import or export values.

The Import value for a meter reading MUST indicate that the reading indicates a consumed amount.

The Export value for a meter reading MUST indicate that the reading indicates a produced amount.

This field MUST be encoded according to the Rate Types values defined in [26].

Meter Type (6 bits)

This field is used to specify the type of metering physical unit that is being reported.

This field MUST be encoded according to the Meter Types values defined in [26].

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Pay Meter (4 bits)

This field is used to indicate the type of payment that is used for the meter.

This field MUST be encoded according to the Pay meter values defined in [26].

Dataset Supported (24 bits)

This field is used to advertise which datasets are supported and can be requested from the metering node with the Meter Table Current Data Get Command.

This field MUST be encoded according to the Meter dataset bitmask values defined in [26].

Dataset History Supported (24 bits)

This field is used to advertise which datasets are supported and can be requests from the metering node with the Meter Table Historical Data Get Command.

This field MUST be encoded according to the Meter dataset bitmask values defined in [26].

Data History Supported (24 bits)

This field is used to advertise the total number of entries that the supporting node can hold in memory for historical values.

This field MUST be in the range 0..16777215.

The value 0 MUST indicate that Historical data cannot be requested to the sending node.

2.2.62.7 Meter Table Status Supported Get Command

This command is used to request the supported operating status event parameters and logging depth of the metering device.

The Meter Table Status Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.368: Meter Table Status Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_STATUS_SUPPORTED_GET (0x07)							

2.2.62.8 Meter Table Status Supported Report Command

This command is used to report the supported operation status and logging depth of these in the meter.

Table 2.369: Meter Table Status Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_STATUS_SUPPORTED_REPORT (0x08)							
Supported Operating Status 1							
Supported Operating Status 2							
Supported Operating Status 3							
Status Event Log Depth							

Supported Operating Status (24 bits)

This field is used to advertise which operating statuses are supported and can be sent by the metering node in the Meter Table Status Report Command.

This field MUST be encoded as a bitmask according to the Operating status values defined in [26].

If the Operating Status event is supported, the corresponding bit MUST be set to 1.

If the Operating Status event is not supported, the corresponding bit MUST be set to 0.

A supporting node MAY support no operating status event and set this field to 0.

Status Event Log Depth (8 bits)

This field MUST indicate the supported depth of the event log.

This field MUST be in the range 0..255.

If the supporting node can only return the current status, this field MUST be set to 0.

2.2.62.9 Meter Table Status Depth Get Command

This command is used to request the current operating status of the metering device or to request a number of the latest status event from the logs.

The Meter Table Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.370: Meter Table Status Depth Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_STATUS_DEPTH_GET (0x09)							
Status Event Log Depth							

Status Event Log Depth (8 bits)

This field is used to indicate the number of last recorded events that should be returned.

The value 0x00 MUST indicate that the receiving node MUST return the current status only.

Values in the range 0x01..0xFE MUST indicate the number of last recorded event that the receiving node SHOULD return.

The value 0xFF MUST indicate that the receiving node SHOULD return its entire status event log.

2.2.62.10 Meter Table Status Date Get Command

This command is used to request a number of status events recorded in a certain time interval.

The Meter Table Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

If a supporting node does not support a status event log history, it MUST return the current state of the meter (refer to the Meter Table Status Report Command).

Table 2.371: Meter Table Status Date Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_STATUS_DEPTH_GET (0x09)							
Maximum Reports							
Start - Year 1							
Start - Year 2							
Start - Month							
Start - Day							
Start - Hour Local Time							
Start - Minute Local Time							
Start - Second Local Time							
Stop - Year 1							
Stop - Year 2							
Stop - Month							
Stop - Day							
Stop - Hour Local Time							
Stop - Minute Local Time							
Stop - Second Local Time							

Maximum Reports (8 bits)

This field is used to indicate the maximum number of Meter Table Status Report Command that can be returned to advertise the requested status event history. The most recent recorded log entries MUST be returned first.

The value 0x00 MUST indicate that there is no maximum number of reports for returning the event log and the supporting node MUST return as many Reports as necessary to advertise the requested entry logs.

Values in the range 0x01..0xFF MUST indicate an actual upper limit number of reports to advertise the requested entry logs.

Start/Stop - Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start/Stop - Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Start/Stop - Day (8 bits)

Specify the day of the month between 01 and 31.

Start/Stop - Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start/Stop - Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Start/Stop - Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

2.2.62.11 Meter Table Status Report Command

This command is used to advertise the current status and optionally historical status data of the meter.

Table 2.372: Meter Table Status Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_STATUS_DEPTH_GET (0x09)							
Reports to follow							
Current Operating Status 1							
Current Operating Status 2							
Current Operating Status 3							
Event 1 - Type	Reserved		Event 1 - Operating Status Event ID				
Event 1 - Year 1							
Event 1 - Year 2							
Event 1 - Month							
Event 1 - Day							
Event 1 - Hour Local Time							
Event 1 - Minute Local Time							
Event 1 - Second Local Time							
...							
Event N - Type	Reserved		Event N - Operating Status Event ID				
Event N - Year 1							
Event N - Year 2							
Event N - Month							
Event N - Day							
Event N - Hour Local Time							
Event N -Minute Local Time							
Event N - Second Local Time							

Reports to follow (8 bits)

This field MUST be used if multiple Meter Table Status Report Commands are used to report the requested Operating status.

Values in the range 0x00..0xFE MUST indicate the actual number of commands following the actual command.

The value 0xFF MUST indicate that the number of remaining commands have not been calculated yet or is higher than 255.

Current Operating Status (24 bits)

This field is used to advertise which operating statuses are currently active at the sending node.

This field MUST be encoded as a bitmask according to the Operating status values defined in [26].

If the *Operating Status event* is active, the corresponding bit MUST be set to 1.

If the *Operating Status event* is not active, the corresponding bit MUST be set to 0.

Event - Type (N * 1 bit)

This field is used to advertise the type of the actual event in the event log. It represents the transition to or from a state.

The value 0 MUST indicate that the sending node entered the state described by the corresponding Operating Status Event ID at the reported time.

The value 1 MUST indicate that the sending node left the state described by the corresponding Operating Status Event ID at the reported time.

Event - Operating Status Event ID (N * 5 bits)

This field is used to advertise the identifier of the actual event in the event log.

This field MUST be encoded according to the Operating status Event identifier values defined in [26].

Event - Year (N * 16 bits)

This field is used to specify the year in the usual Gregorian calendar for the actual event. The first byte (Year 1) MUST be the most significant byte.

Event - Month (N * 8 bits)

This field is used to specify the month of the year between 01 (January) and 12 (December) for the actual event. This field MUST be in the range 1..12.

Event - Day (N * 8 bits)

This field is used to specify the day of the month for the actual event. This field MUST be in the range 1..31

Event - Hour Local Time (N * 8 bits)

This field is used to specify the number of complete hours that have passed since midnight in local time for the actual event. This field MUST be in the range 0..23.

Event - Minute Local Time (N * 8 bits)

This field is used to specify the number of complete minutes that have passed since the start of the hour in local time for the actual event. This field MUST be in the range 0..59.

Event - Second Local Time (N * 8 bits)

This field is used to specify the number of complete seconds since the start of the minute in local time for the actual event. The value 60 used to keep UTC from wandering away is not supported. This field MUST be in the range 0..59.

2.2.62.12 Meter Table Current Data Get Command

This command is used to request a number of time stamped values (current) in physical units according to the dataset mask.

The Meter Table Current Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.373: Meter Table Current Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_CURRENT_DATA_GET (0x0C)							
Dataset Requested 1							
Dataset Requested 2							
Dataset Requested 3							

Dataset Requested (24 bits)

This field is used to indicate which datasets are requested from the supporting node.

This field MUST be encoded as a bitmask and according to the Meter dataset bitmask values defined in [26].

2.2.62.13 Meter Table Current Data Report Command

This command is is used to report a number of time stamped values.

Table 2.374: Meter Table Current Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_CURRENT_DATA_REPORT (0x0D)							
Reports to Follow							
Reserved						Rate Type	
Dataset 1							
Dataset 2							
Dataset 3							
Year 1							
Year 2							
Month							
Day							
Hour Local Time							
Minute Local Time							
Second Local Time							
Current Meter Precision 1			Current Meter Scale 1				
Current Value 1,1							
Current Value 1,2							
Current Value 1,3							
Current Value 1,4							
...							
Current Meter Precision N			Current Meter Scale N				
Current Value N,1							
Current Value N,2							
Current Value N,3							
Current Value N,4							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Reports to follow (8 bits)

This field MUST be used if multiple Meter Table Current Data Report Commands are used to report the requested values.

Values in the range 0x00..0xFE MUST indicate the actual number of commands following the actual command.

The value 0xFF MUST indicate that the number of remaining commands have not been calculated yet or is higher than 255.

Rate Type (2 bits)

This field is used to specify the type of parameters advertised in this command. This field MUST be encoded according to the rate types values defined in [26].

Dataset (24 bits)

This field is used to indicate which datasets are included in this command. This field MUST be encoded as a bitmask and according to the Meter dataset bitmask values defined in [26].

The length of the *Current Meter Precision*, *Current Meter Scale* and *Current Value* fields MUST be according to the number of bits set in this field.

Year (16 bits)

This field is used to specify the year in the usual Gregorian calendar. The first byte (Year 1) MUST be the most significant byte. A year equal to 0x0000 MUST indicate that an accumulated value is not determined yet.

Month (8 bits)

This field is used to specify the month of the year between 01 (January) and 12 (December). This field MUST be in the range 1..12.

Day (8 bits)

This field is used to specify the day of the month. This field MUST be in the range 1..31.

Hour Local Time (8 bits)

This field is used to specify the number of complete hours that have passed since midnight in local time. This field MUST be in the range 0..23.

Minute Local Time (8 bits)

This field is used to specify the number of complete minutes that have passed since the start of the hour in local time. This field MUST be in the range 0..59.

Second Local Time (8 bits)

This field is used to specify the number of complete seconds since the start of the minute in local time. The value 60 used to keep UTC from wandering away is not supported. This field MUST be in the range 0..59.

Meter Precision (N * 3 bits)

This field is used to indicate how many decimal places are included in the corresponding Current Value field. For example, the Current Value field set to 1025 with this field set to 2 MUST be interpreted as equal to 10.25.

Meter Scale (N * 5 bits)

This field is used to indicate the scale (unit) for the corresponding Current Value field reading. This field MUST be encoded according to the Meter Scale values defined in [26].

Current Value (N * 4 bytes)

This field is used to advertise an actual meter reading.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12, Signed field encoding (two's complement representation).

Readings MUST be advertised according to the *Dataset* field. The first reading MUST correspond to the first bit set in the *Dataset* field, the second reading MUST correspond to the second bit set in the *Dataset* field, and so on.

A controlling node MUST always show the value even if the reading type or scale are unknown.

A controlling node SHOULD implement the capability to update its list of Meter datasets and scales, so that new Meter readings and scales added in [26] are not presented as unknown. If a controlling node receives an unknown Meter reading or scale, it SHOULD allow the user to assign a free-text description to that Meter reading

2.2.62.14 Meter Table Historical Data Get Command

This command is used to request a number of time stamped values (historical) in physical units according to rate type, dataset bitmask and time interval.

The Meter Table Historical Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.375: Meter Table Historical Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_HISTORICAL_DATA_GET (0x0E)							
Maximum Reports							
Historical Dataset Requested 1							
Historical Dataset Requested 2							
Historical Dataset Requested 3							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Start Second Local Time							
Stop Year 1							
Stop Year 2							
Stop Month							
Stop Day							
Stop Hour Local Time							
Stop Minute Local Time							
Stop Second Local Time							

Maximum Reports (8 bits)

This field is used to indicate the maximum number of Meter Table Historical Data Report Commands that can be returned to advertise the requested historical data. The most recent recorded values MUST be returned first.

The value 0x00 MUST indicate that there is no maximum number of reports for returning the event log and the supporting node MUST return as many Reports as necessary to advertise the requested historical values.

Values in the range 0x01..0xFF MUST indicate an actual upper limit number of reports to advertise the requested historical values.

Dataset History (24 bits)

This field is used to indicate which datasets are requested from the supporting node.

This field MUST be encoded as a bitmask and according to the Meter dataset bitmask values defined in [26].

Start/Stop Year (16 bits)

This field is used to specify the year in the usual Gregorian calendar. The first byte (Year 1) MUST be the most significant byte.

Start/Stop Month (8 bits)

This field is used to specify the month of the year between 01 (January) and 12 (December). This field MUST be in the range 1..12.

Start/Stop Day (8 bits)

This field is used to specify the day of the month. This field MUST be in the range 1..31.

Start/Stop Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start/Stop Minute Local Time (8 bits)

This field is used to specify the number of complete minutes that have passed since the start of the hour (00-59) in local time. This field MUST be in the range 0..59.

Start/Stop Second Local Time (8 bits)

This field is used to specify the number of complete seconds since the start of the minute in local time. The value 60 used to keep UTC from wandering away is not supported. This field MUST be in the range 0..59.

2.2.62.15 Meter Table Historical Data Report Command

This command is used to report a number of time stamped values.

Table 2.376: Meter Table Historical Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_HISTORICAL_DATA_REPORT (0x0F)							
Reports to Follow							
Reserved						Rate Type	
Dataset 1							
Dataset 2							
Dataset 3							
Historical Year 1							
Historical Year 2							
Historical Month							
Historical Day							
Historical Hour Local Time							
Historical Minute Local Time							
Historical Second Local Time							
Historical Precision 1				Historical Scale 1			
Historical Value 1,1							
Historical Value 1,2							
Historical Value 1,3							
Historical Value 1,4							
...							
Historical Precision N				Historical Scale N			
Historical Value N,1							
Historical Value N,2							
Historical Value N,3							
Historical Value N,4							

Reports to Follow (8 bits)

This field MUST be used if multiple Meter Table Current Data Report Commands are used to report the requested values.

Values in the range 0x00..0xFE MUST indicate the actual number of commands following the actual command.

The value 0xFF MUST indicate that the number of remaining commands have not been calculated yet or is higher than 255.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Type (2 bits)

This field is used to specify the type of parameters advertised in this command. This field MUST be encoded according to the rate types values defined in [26].

Dataset (24 bits)

This field is used to indicate which datasets are included in this command. This field MUST be encoded as a bitmask and according to the Meter dataset bitmask values defined in [26].

If no historical data has been registered for the requested period, this field MUST be set to 0 and the *Historical Scale*, *Historical Precision* and *Historical Value* fields MUST be omitted from the command.

Historical Year (16 bits)

This field is used to specify for the dataset the year in the usual Gregorian calendar. The first byte (Year 1) MUST be the most significant byte. A year equal to 0x0000 indicates that an accumulated value is not determined yet.

Historical Month (8 bits)

This field is used to specify for the dataset the month of the year between 01 (January) and 12 (December). This field MUST be in the range 1..12.

Historical Day (8 bits)

This field is used to specify for the dataset the day of the month. This field MUST be in the range 1..31.

Historical Hour Local Time (8 bits)

This field is used to specify for the dataset the number of complete hours that have passed since midnight in local time. This field MUST be in the range 0..23.

Historical Minute Local Time (8 bits)

This field is used to specify for the dataset the number of complete minutes that have passed since the start of the hour in local time. This field MUST be in the range 0..59.

Historical Second Local Time (8 bits)

This field is used to specify for the dataset the number of complete seconds since the start of the minute in local time. The value 60 used to keep UTC from wandering away is not supported. This field MUST be in the range 0..59.

Historical Precision (N * 3 bits)

This field is used to specify how many decimal places are included in the corresponding *Historical Value* field. For example, the *Historical Value* field set to 1025 with this field set to 2 MUST be interpreted as equal to 10.25.

Historical Scale (5 bits)

This field is used to indicate the scale (unit) of the corresponding *Historical Value* field reading. This field MUST be encoded according to the Meter Scale values defined in [26].

Historical Value

This field is used to advertise an actual historical meter reading.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12, Signed field encoding (two's complement representation).

Readings MUST be advertised according to the *Dataset* field. The first reading MUST correspond to the first bit set in the *Dataset* field, the second reading MUST correspond to the second bit set in the *Dataset* field, and so on.

A controlling node MUST always show the value even if the reading type or scale are unknown.

A controlling node SHOULD implement the capability to update its list of Meter datasets and scales, so that new Meter readings and scales added in [26] are not presented as unknown. If a controlling node receives an unknown Meter reading or scale, it SHOULD allow the user to assign a free-text description to that Meter reading

2.2.63 Meter Table Monitor Command Class, version 2

The Meter Table Monitor Command Class defines the Commands necessary to read historical and accumulated values in physical units from a water meter or other metering device (gas, electric etc.) or electric sub-metering device and thereby enabling automatic meter reading capabilities.

2.2.63.1 Compatibility Considerations

The Meter Table Monitor Command Class, version 2 is backwards compatible with the Meter Table Monitor Command Class, version 1.

All commands and fields not mentioned in this version MUST remain unchanged from the Meter Table Monitor Command Class, version 1.

2.2.63.2 Meter Table Point Adm. Number Report Command

This command reports parameters used for identification of customer and metering device.

Table 2.377: Meter Table Point Adm Number Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_POINT_ADM_NO_REPORT (0x02)							
Reserved			Number of Meter Point Adm. Number Characters				
Meter Point Adm. Number Character 1							
...							
Meter Point Adm. Number Character N							

Fields not described below MUST remain unchanged from version 1.

Number of Meter Point Adm. Number Characters (5 bits)

If the *Meter Table Point Adm, Number* has not been set using the Meter Table Configuration Command Class, this field MUST be set to 0x00 and the *Meter Point Adm. Numbers Character* field MUST be omitted.

2.2.63.3 Meter Table ID Report Command

This command reports parameters used for identification of customer and metering device.

Table 2.378: Meter Table ID Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_ID_REPORT (0x04)							
Reserved			Number of Meter ID Characters				
Meter ID Character 1							
...							
Meter ID Character N							

Fields not described below MUST remain unchanged from version 1.

Number of Meter ID Characters (5 bits)

This field is used to indicate the length of the *Meter ID Character* field in bytes.

This field MUST be in the range 0..32.

The value 0 MUST indicate that the Meter ID is not used and the *Meter ID Character* field MUST be omitted from the command.

2.2.63.4 Meter Table Capability Report Command

This command is used to advertise meter table capabilities.

Table 2.379: Meter Table Capability Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_TABLE_CAPABILITY_REPORT (0x06)							
Rate Type		Meter Type					
Reserved				Pay Meter			
Dataset Supported 1							
Dataset Supported 2							
Dataset Supported 3							
Dataset History Supported 1							
Dataset History Supported 2							
Dataset History Supported 3							
Data History Supported 1							
Data History Supported 2							
Data History Supported 3							

Fields not described below MUST remain unchanged from version 1.

Rate Type (2 bits)

This field is used to indicate if the actual reading advertises import or export values.

The Import value for a meter reading MUST indicate that the reading indicates a consumed amount.

The Export value for a meter reading MUST indicate that the reading indicates a produced amount.

This field MUST be encoded according to the Rate Types values defined in [26].

If the *Meter Type* field is set to Submeter (0x0B), this field MUST be set to Import (0x01).

2.2.63.5 Meter Table Current Data Report Command

This command is used to report a number of time stamped values.

Table 2.380: Meter Table Current Data Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_CURRENT_DATA_REPORT (0x0D)							
Reports to Follow							
Operating Status Indication	Reserved						Rate Type
Dataset 1							
Dataset 2							
Dataset 3							
Year 1							
Year 2							
Month							
Day							
Hour Local Time							
Minute Local Time							
Second Local Time							
Current Meter Precision 1			Current Meter Scale 1				
Current Value 1,1							
Current Value 1,2							
Current Value 1,3							
Current Value 1,4							
...							
Current Meter Precision N			Current Meter Scale N				
Current Value N,1							
Current Value N,2							
Current Value N,3							
Current Value N,4							

Fields not described below MUST remain unchanged from version 1.

Operating Status Indication (1 bit)

This field is used to indicate that the reported meter data is measured while the meter is in an operating status different from Normal (0x00) e.g. accuracy warning or clock not accurate.

The value 1 MUST indicate that the meter is operating in a status different from Normal

The value 0 MUST indicate that the meter is operating in Normal mode (Operating status=0x00).

Rate Type (2 bits)

This field is used to indicate the type of parameters in the report. This field MUST be encoded according to the Rate Types values defined in [26].

If the *Meter Type* field is set to Submeter (0x0B), this field MUST be set to Import (0x01).

2.2.63.6 Meter Table Historical Data Report Command

This command is used to report a number of time stamped values.

Table 2.381: Meter Table Historical Data Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR (0x3D)							
Command = METER_TBL_HISTORICAL_DATA_REPORT (0x0F)							
Reports to Follow							
Operating Status Indication	Reserved						Rate Type
Dataset 1							
Dataset 2							
Dataset 3							
Historical Year 1							
Historical Year 2							
Historical Month							
Historical Day							
Historical Hour Local Time							
Historical Minute Local Time							
Historical Second Local Time							
Historical Precision 1			Historical Scale 1				
Historical Value 1,1							
Historical Value 1,2							
Historical Value 1,3							
Historical Value 1,4							
...							
Historical Precision N			Historical Scale N				
Historical Value N,1							
Historical Value N,2							
Historical Value N,3							
Historical Value N,4							

Fields not described below MUST remain unchanged from version 1.

Operating Status Indication (1 bit)

This field is used to indicate that the reported meter data is measured while the meter is in an operating status different from Normal (0x00) e.g. accuracy warning or clock not accurate.

The value 1 MUST indicate that the meter is operating in a status different from Normal

The value 0 MUST indicate that the meter is operating in Normal mode (Operating status=0x00).

Rate Type (2 bits)

This field is used to indicate the type of parameters in the report. This field MUST be encoded according to the Rate Types values defined in [26].

If the *Meter Type* field is set to Submeter (0x0B), this field MUST be set to Import (0x01).

2.2.64 Meter Table Monitor Command Class, version 3

The Meter Table Monitor Command Class, version 3 is used for advanced metering applications. It allows reading historical and accumulated values in physical units from a water meter or other metering device (gas, electric etc.) or electric sub-metering device and thereby enabling automatic meter reading capabilities

2.2.64.1 Compatibility Considerations

The Meter Table Monitor Command Class, version 3 is backwards compatible with the Meter Table Monitor Command Class, version 2.

All commands and fields not mentioned in this version MUST remain unchanged from the Meter Table Monitor Command Class, version 2.

The list of supported Meter Types, Meter scales and Meter dataset bitmasks is moved to [26]. Values not defined in [26] are reserved and MUST NOT be used by a supporting node.

2.2.65 Meter Table Push Configuration Command Class, version 1 [OBSOLETED]

The Meter Table Push Configuration Command Class is used to configure the meter to send a Current Data Report at a given interval. The meter may be configured to return different data sets at different intervals using both the primary and secondary push commands.

2.2.65.1 Meter Table Push Configuration Set Command

This command is used to request the meter to send a Current Data Report at a given interval. The meter may be configured to return different data sets at different intervals using both the primary and secondary push commands.

Table 2.382: Meter Table Push Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_PUSH (0x3E)							
Command = METER_TBL_PUSH_CONFIGURATION_SET (0x01)							
Reserved			P/S	Operating Status Push Mode			
Push Dataset 1							
Push Dataset 2							
Push Dataset 3							
Interval Months							
Interval Days							
Interval Hours							
Interval Minutes							
Push Node ID							

Operating Status Push Mode (4 bits)

This field is used to configure if the Meter Table Status Report Command (refer to Section 2.2.62.11) participates in the Push Functionality

Table 2.383: Operating Status Push Mode

Operating Status Push Mode Identifier	Description
0x00	Operating Status push disabled
0x01	Operating Status push based on Interval
0x02	Operating Status push based on Status Change
0x03	Operating Status push Based on Interval AND status change
0x04-0x0F	Reserved

P/S (1 bit)

Table 2.384: P/S

P/S	Description
0x00	Primary push configuration
0x01	Secondary push configuration

Push Dataset (24 bits)

The Push Dataset parameter is use to indicate which parameters are requested to be pushed from the meter. This field MUST be encoded according to the Meter Dataset defined in [26].

Interval Months (8 bits)

Specify the number of months between pushing the push dataset.

Interval Day (8 bits)

Specify the number of days between pushing the push dataset.

Interval Hours (8 bits)

Specify the number of hours between pushing the push dataset.

Interval Minute (8 bits)

Specify the number of minutes between pushing the push dataset.

Push Node ID (8 bits)

Specify the node ID of the node to receive push dataset in the given interval.

2.2.65.2 Meter Table Push Configuration Get Command

This command is used to request the meters push configuration

The Meter Table Push Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.385: Meter Table Push Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_PUSH (0x3E)							
Command = METER_TBL_PUSH_CONFIGURATION_GET (0x02)							

2.2.65.3 Meter Table Push Configuration Report Command

This command is used report the current Push Configuration

Table 2.386: Meter Table Push Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_PUSH (0x3E)							
Command = METER_TBL_PUSH_CONFIGURATION_REPORT (0x03)							
Reserved			P/S	Operating Status Push Mode			
Push Dataset 1							
Push Dataset 2							
Push Dataset 3							
Interval Months							
Interval Days							
Interval Hours							
Interval Minutes							
Push Node ID							

Refer to Meter Table Push Configuration Set Command (Section 2.2.65.1) for detailed description of the fields.

2.2.66 Move to Position Window Covering Command Class, version 1

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST NOT support this command class.

Window covering device implementations SHOULD support the Window Covering Command Class.

The Move To Position Window Covering Command Class is used to control the position of a window covering device.

The Move To Position Window Covering Command Class is an actuator control command class. Refer to Section 2.1.6

2.2.66.1 Move To Position Set Command

This command is used to instruct a window covering to move to a new position.

Table 2.387: Move To Position Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MTP_WINDOW_COVERING (0x51)							
Command = MOVE_TO_POSITION_SET (0x01)							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 2.388.

Table 2.388: Move To Position Set::Value

Value	Level	State
0 (0x00)	0%	Closed
1..99 (0x01..0x63)	Almost closed .. 100% Open	Open
...	*Reserved*	*Reserved*
255 (0xFF)	100% Open	Open

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.66.2 Move To Position Get Command

This command is used to request the status of a window covering device.

The Move To Position Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.389: Move To Position Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MTP_WINDOW_COVERING (0x51)							
Command = MOVE_TO_POSITION_GET (0x02)							

2.2.66.3 Move To Position Report Command

This command is used to advertise the status of a window covering device.

Table 2.390: Move To Position Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MTP_WINDOW_COVERING (0x51)							
Command = MOVE_TO_POSITION_REPORT (0x03)							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 2.388.

The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command when a transition has ended.

2.2.67 Multilevel Sensor Command Class, version 1-4

The Multilevel Sensor Command Class is used to advertise numerical sensor readings..

2.2.67.1 Multilevel Sensor Get Command

This command is used to request the current reading from a multilevel sensor.

CC:0031.01.04.11.001 The Multilevel Sensor Report Command MUST be returned in response to this command.

CC:0031.01.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0031.01.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.391: Multilevel Sensor Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_GET (0x04)							

2.2.67.2 Multilevel Sensor Report Command

This command is used by a supporting node to advertise its current sensor reading for its supported sensor type.

Table 2.392: Multilevel Sensor Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_REPORT (0x05)							
Sensor Type							
Precision			Scale		Size		
Sensor Value 1							
...							
Sensor Value N							

Sensor Type (8 bits)

This field is used to specify the sensor type of the actual sensor reading.

CC:0031.01.05.11.001 This field MUST be set to a value defined in [27]. Values not defined in [27] are reserved and MUST NOT be used.

CC:0031.01.05.11.002 A node MUST support as a minimum the highest Multilevel Sensor Command Class version associated with the Scale and Type it supports. The minimum required version for each Scale and Type is specified in [27].

Precision (3 bits)

CC:0031.01.05.11.003 This field is used to indicate how many decimal places are included the Sensor Value field. For example, the Sensor Value 1025 with precision 2 MUST be interpreted as equal to 10.25.

Scale (2 bits)

This field is used to indicate what scale is used for the actual sensor reading.

CC:0031.01.05.11.004 This field MUST be set to a value defined in [27]. Values not defined in [27] are reserved and MUST NOT be used.

CC:0031.01.05.11.005 A node MUST implement as a minimum the highest Multilevel Sensor Command Class version associated with the Scale and Type it supports. The minimum required version for each Scale and Type is specified in [27].

Size (3 bits)

This field is used to indicate the length in bytes of the Sensor Value field.

CC:0031.01.05.11.006 This field MUST be set to 1, 2 or 4.

Sensor Value (N bytes)

This field is used to advertise the value of the actual sensor reading.

CC:0031.01.05.11.007 The length of this field in bytes MUST be according to the Size field value.

CC:0031.01.05.11.008 The first byte MUST be the most significant byte.

CC:0031.01.05.11.009 This field MUST be encoded using signed representation and comply with Table 2.12, Signed field encoding (two's complement representation).

CC:0031.01.05.11.00A A controlling node receiving this command MUST always show the sensor value as is even though the Sensor Type and/or Scale are unknown.

CC:0031.01.05.12.001 A controlling node SHOULD implement the capability to update its list of Sensor Type and Scales, so that new Sensor Types and Scales added in [27] are not presented as unknown. If a controlling node

CC:0031.01.05.12.002 receives an unknown Sensor Type or Scale, it SHOULD allow the user to assign a free-text description to the sensor reading.

2.2.68 Multilevel Sensor Command Class, version 5-11

The Multilevel Sensor Command Class is used to advertise numerical sensor readings .

2.2.68.1 Compatibility considerations

A node supporting version 5 or newer may support several sensor types and advertise the readings for each of them.

Version 5 of this command class is extended with the following functionality:

- A "get-supported" mechanism for the controlling device to interview the multilevel sensor for its supported sensor types and/or scales
- Additional sensor type and scale fields to the Multilevel Sensor Get command to request for a specific sensor report
- Additional sensor types and/or scales to the list of multilevel sensors

Version 6 of this command class is extended with the following functionality:

- Additional sensor types

Version 7 of this command class is extended with the following functionality:

- Additional sensor types

Version 8 of this command class is extended with the following functionality:

- New Sensor Types and Scale Values for rotation and linear movement.
- New Sensor Types and Scale Values for Smoke Density

Multilevel Sensor Command Class, version 8 deprecates the Sensor Type "Angle Position".

Version 9 of this command class is extended with the following functionality:

- New Sensor Types and Scale Values for Water Flow and Water Pressure
- New Sensor Types and Scale Values for RF Signal Strength

Version 10 of this command class is extended with the following functionality:

- New Sensor Types and Scale Values for Particulate Matter 10 and Respiratory rate
- New Scale Values for CO and VOC Sensors Types

Version 11 of this command class is extended with the following functionality:

- New Sensor Types and Scale Values.
- Moved the list of assigned Sensor Types and Scale Values to an external registry [27].
- Deprecated the "General Purpose" Sensor Type

2.2.68.1.1 Unknown Multilevel Sensor Types and Scales

CC:0031.05.05.22.001 A controlling node SHOULD implement the capability to update its Multilevel sensor types and scales list, so that new Types and Scales added in [27] are not presented as unknown. If a controlling node CC:0031.05.05.22.002 receives an unknown Type or Scale, it SHOULD allow the user to assign a free-text description to that Type or Scale.

2.2.68.2 Multilevel Sensor Get Supported Sensor Command

This command is used to request the supported Sensor Types from a supporting node.

CC:0031.05.01.11.001 The Multilevel Sensor Supported Sensor Report Command MUST be returned in response to this command.

CC:0031.05.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:0031.05.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.393: Multilevel Sensor Get Supported Sensor Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_SUPPORTED_GET_SENSOR (0x01)							

2.2.68.3 Multilevel Sensor Supported Sensor Report Command

This command is used to advertise the supported Sensor Types by a supporting node.

Table 2.394: Multilevel Sensor Supported Sensor Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_SUPPORTED_SENSOR_REPORT (0x02)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

This field is used to advertise the supported sensor types by sending node.

CC:0031.05.02.11.001 This field MUST be treated as a bit mask and interpreted as follows:

- Bit 0 in Bit Mask 1 indicates if Sensor Type = Air Temperature (0x01) is supported
- Bit 1 in Bit Mask 1 indicates if Sensor Type = General Purpose (0x02) is supported
- Bit 2 in Bit Mask 1 indicates if Sensor Type = luminance (0x03) is supported
- ...

The list of Sensor Types is defined in [27].

CC:0031.05.02.11.002 The value 1 MUST indicate that the corresponding Sensor Type is supported.

The value 0 MUST indicate that the corresponding Sensor Type is not supported.

CC:0031.05.02.11.003 It is only necessary to send the Bit Mask fields from 1 and up to the Bit Mask N indicating the last supported Sensor Type. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

Note that the mapping of bit 0 to Sensor Type = 1 differs from the support mapping used by the Notification Command Class. The Notification Command Class maps bit 1 to Notification Type =1.

2.2.68.4 Multilevel Sensor Get Supported Scale Command

This command is used to retrieve the supported scales of the specific sensor type from the Multilevel Sensor device.

CC:0031.05.03.11.001 The Multilevel Sensor Supported Scale Report Command MUST be returned in response to this command.

CC:0031.05.03.11.002 This command MUST NOT be issued via multicast addressing.

CC:0031.05.03.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.395: Multilevel Sensor Get Supported Scale Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_SUPPORTED_GET_SCALE (0x03)							
Sensor Type							

Sensor Type (8 bits)

This field is used to request a given sensor type.

CC:0031.05.03.12.001 If the specified sensor type is not supported, a receiving node SHOULD set the *Scale Bit Mask* field to 0 in the returned response.

2.2.68.5 Multilevel Sensor Supported Scale Report Command

This command is used to advertise the supported scales of a specified multilevel sensor type.

Table 2.396: Multilevel Sensor Supported Scale Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_SUPPORTED_SCALE_REPORT (0x06)							
Sensor Type							
Reserved				Scale Bit Mask			

Sensor Type (8 bits)

This field is used to indicate the actual Sensor Type for which the supported scales are being advertised.

Scale Bit Mask (4 bits)

This field is used to advertise the supported scales for the actual sensor type.

CC:0031.05.06.11.001 This field MUST be treated as a bit mask and interpreted as follow:

- Bit 0 indicates support for the first scale of the actual Sensor Type
- Bit 1 indicates support for the second scale of the actual Sensor Type
- …

The list of scales for a given sensor type is defined in [27].

CC:0031.05.06.11.002 The value 1 MUST indicate that the corresponding scale is supported for the sensor type.

The value 0 MUST indicate that the corresponding scale is not supported for the sensor type.

Reserved

CC:0031.05.06.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.68.6 Multilevel Sensor Get Command

This command is used to request the current reading from a multilevel sensor.

CC:0031.05.04.11.001 The Multilevel Sensor Report Command MUST be returned in response to this command.

CC:0031.05.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0031.05.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods

Table 2.397: Multilevel Sensor Get Command, version 5-11

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_GET (0x04)							
Sensor Type							
Reserved			Scale		Reserved		

Sensor Type (8 bits)

This field is used to request a node to report a reading for the specified sensor type.

CC:0031.05.04.11.004 If this field is unspecified or a receiving node does not support the specified Sensor Type, it MUST reply with a pre-defined default Sensor Type and Scale.

Scale (2 bits)

This field is used to request a node to report a reading with a particular scale for the actual Sensor Type.

CC:0031.05.04.11.005 A node receiving a non-supported scale for the actual Sensor Type MUST reply with a supported scale within the Sensor type.

CC:0031.05.04.11.006 A sending node MUST ensure that the receiver supports the requested Sensor Types and/or Scales using the Multilevel Sensor Get Supported Sensor/Scale commands.

Reserved

CC:0031.05.04.11.007 These fields MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.68.7 Multilevel Sensor Report Command

This command is used to advertise a multilevel sensor reading.

Table 2.398: Multilevel Sensor Report Command, version 5-11

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL (0x31)							
Command = SENSOR_MULTILEVEL_REPORT (0x05)							
Sensor Type							
Precision			Scale		Size		
Sensor Value 1							
...							
Sensor Value N							

Sensor Type (8 bits)

This field is used to specify the sensor type of the actual sensor reading.

CC:0031.05.05.11.001 This field MUST be set to a value defined in [27]. Values not defined in [27] are reserved and MUST NOT be used.

CC:0031.05.05.11.002

A node MUST support as a minimum the highest Multilevel Sensor Command Class version associated with the Scale and Type it supports. The minimum required version for each Scale and Type is specified in [27].

Precision (3 bits)

This field is used to indicate how many decimal places are included the Sensor Value field. For
CC:0031.05.05.11.003 example, the Sensor Value 1025 with precision 2 MUST be interpreted as equal to 10.25.

Scale (2 bits)

This field is used to indicate what scale is used for the actual sensor reading.

CC:0031.05.05.11.004 This field MUST be set to a value defined in [27]. Values not defined in [27] are reserved and MUST NOT be used.

CC:0031.05.05.11.005 A node MUST implement as a minimum the highest Multilevel Sensor Command Class version associated with the Scale and Type it supports. The minimum required version for each Scale and Type is specified in [27].

Size (3 bits)

This field is used to indicate the length in bytes of the Sensor Value field.

CC:0031.05.05.11.006 This field MUST be set to 1, 2 or 4.

Sensor Value (N bytes)

This field is used to advertise the value of the actual sensor reading.

CC:0031.05.05.11.007 The length of this field MUST be according to the Size field value.

CC:0031.05.05.11.008 The first byte MUST be the most significant byte.

CC:0031.05.05.11.009 This field MUST be encoded using signed representation and comply with Table 2.12, Signed field encoding (two's complement representation).

CC:0031.05.05.11.00A A controlling node receiving this command MUST always show the sensor value as is even though the Sensor Type and/or Scale are unknown.

CC:0031.05.05.12.001 A controlling node SHOULD implement the capability to update its list of Sensor Type and Scales, so that new Sensor Types and Scales added in [27] are not presented as unknown. If a controlling node
CC:0031.05.05.12.002 receives an unknown Sensor Type or Scale, it SHOULD allow the user to assign a free-text description to the sensor reading.

2.2.68.7.1 Detailed description: Sensor Types for Movement and Rotation

A device may report position, velocity (position change over time) or acceleration (velocity change over time). Position, velocity and acceleration may all refer to a linear scale following an axis or a polar scale circling an axis. Position may be reported in an absolute or relative fashion.

The 3D reference coordinate system outlined below MUST be used for reporting changes in the physical orientation.

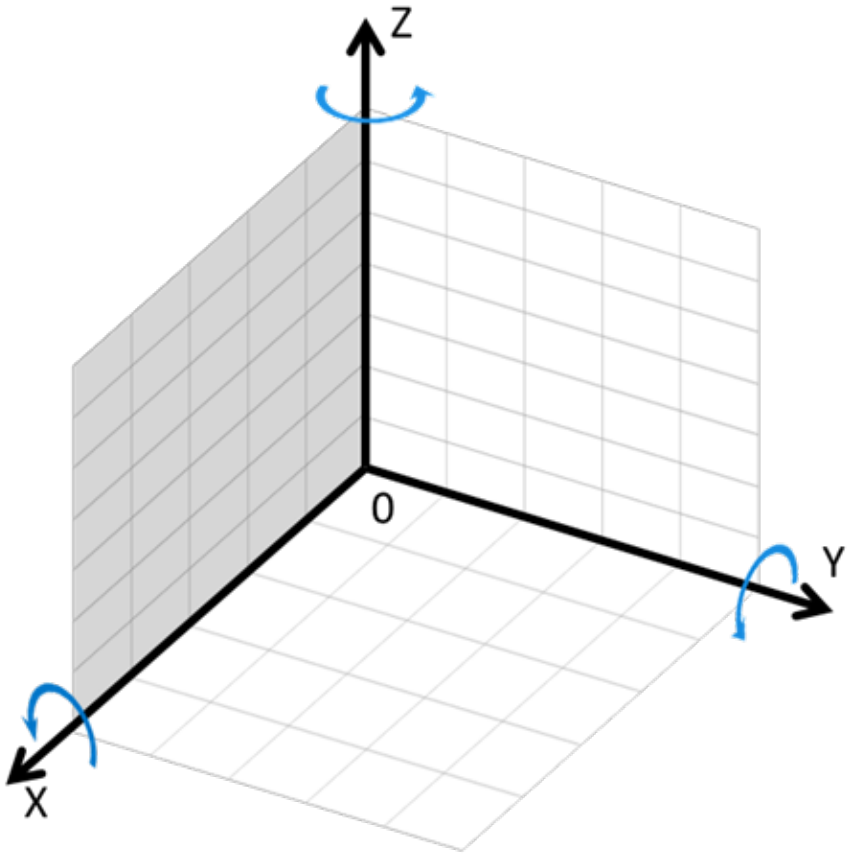


Figure 2.12: 3D reference coordinate system

CC:0031.05.05.11.00B Information relating to linear position, velocity and acceleration MUST refer to the zero position on a given axis. Thus a position change or a velocity MUST be positive if moving towards a larger position, measured from the zero position.

CC:0031.05.05.11.00C Information relating to an angle or change in angle MUST use the right-hand rule. This means that if one (virtually) grabs around an axis with the right hand, with the thumb in the direction of the axis, the angle increases in the direction of the index finger.

Table 2.399: Definition of position, velocity or acceleration

Application	Definition
Linear position, absolute	Position with reference to 0 (e.g. 1 meter)
Linear position, relative	Position with reference to previous position (e.g. 1 meter)
Linear velocity	Position change per time unit (e.g. 1 meter/second)
Linear acceleration	Velocity change per time unit (e.g. 1 meter/second^2)
Polar position, absolute	Angle with reference to 0 (e.g. 45 degrees)
Polar position, relative	Angle with reference to previous angle (e.g. 45 degrees)
Polar velocity (rotation)	Angle change per time unit (e.g. 1 degree/second)
Polar acceleration	Velocity change per time unit (e.g. 1 degree/second^2)

Depending on the number of axes supported by a given device, changes in the physical orientation are mapped to one, two or three axes as outlined in Table 2.400.

Table 2.400: Mapping of 1D, 2D and 3D movement and rotation

Dimensions	Movement	Rotation
1	Along X axis	Around X axis
2	Along X and Y axes	Around X and Y axes
3	Along X, Y and Z axes	Around X, Y and Z axes

A number of Sensor Types allow a device to report changes in the physical orientation.

Table 2.401: Recommended Sensor Types for reporting movement and rotation

Application	**Sensor Type**	**Intended Usage**
1D Linear position, absolute	Distance (v3)	Single axis measurement of position (m)
1D Linear position, relative	*(no support)*	Single axis measurement of position change (m)
1D Linear velocity	Velocity (v2)	Single axis measurement of velocity (m/s)
1D Linear acceleration	Acceleration, X (v8)	Single axis measurement of acceleration (m/s^2)
1D Polar position, absolute	Direction (v2)	Single axis measurement of angle (degrees)
1D Polar position, relative	*(no support)*	Single axis measurement of angle change
1D Polar velocity	Rotation (v5)	Single axis measurement of velocity (RPM)
1D Polar acceleration	*(no support)*	Single axis measurement of acceleration (degree/s^2)
2D Linear position, absolute	*(no support)*	Two axis measurement of position (m)
2D Linear position, relative	*(no support)*	Two axis measurement of position change (m)
2D Linear velocity	*(no support)*	Two axis measurement of velocity (m/s)
2D Linear acceleration	Acceleration, X (v8), Acceleration, Y (v8)	Two axis measurement of acceleration (m/s^2)
2D Polar position, absolute	*(no support)*	Two axis measurement of angle (degrees from North)
2D Polar position, relative	*(no support)*	Two axis measurement of angle change
2D Polar velocity	*(no support)*	Two axis measurement of velocity (RPM)
2D Polar acceleration	*(no support)*	Two axis measurement of acceleration (degree/s^2)
3D Linear position, absolute	*(no support)*	Three axis measurement of position (m)
3D Linear position, relative	*(no support)*	Three axis measurement of position change (m)
3D Linear velocity	*(no support)*	Three axis measurement of velocity (m/s)

continues on next page

Table 2.401 – continued from previous page

Application	Sensor Type	Intended Usage
3D Linear accelleration	Acceleration, X (v8), Acceleration, Y (v8), Acceleration, Z (v8)	Three axis measurement of acceleration (m/s^2)
3D Polar position, absolute	*(no support)*	Three axis measurement of angle (degrees from North)
3D Polar position, relative	*(no support)*	Three axis measurement of angle change
3D Polar velocity	*(no support)*	Three axis measurement of velocity (RPM)
3D Polar accelleration	*(no support)*	Three axis measurement of acceleration (degree/s^2)

2.2.68.7.2 Sensor Type = Acceleration

The sensor types "Acceleration, X", "Acceleration, Y" and "Acceleration, Z" are used to advertise the acceleration of a device along the X, Y and Z axes, respectively.

CC:0031.05.05.11.00D A one-dimensional device MUST report acceleration using the "Acceleration, X" type.

CC:0031.05.05.11.00E A two-dimensional device MUST report acceleration using the "Acceleration, X" and "Acceleration, Y" types.

CC:0031.05.05.11.00F The Scale used MUST be m/s2. An Acceleration value reported with the Scale m/s2 may be converted by a receiving node to a "g-force" level by using the formula below:

1g = 9.81m/s^2

(g represents the unit of Earth Gravity; NOT the weight unit "gram")

2.2.68.7.3 Detailed description: Smoke Density

Figure 2.13 shows the principle of how a photoelectric smoke detector works. As shown the smoke sensor uses the smoke to reflect the light onto a photo cell. So when no smoke is present, the light will not be reflected onto the photo cell. When smoke is present, the light will be reflected onto the photo cell. Dependent on the smoke density, more light will be reflected onto the photo cell. So by measuring the received light strength on the photo cell, it is possible to estimate the smoke density.

It is not possible to determine an accurate unit for this, as it is estimated from the light strength on the photo cell. So the exact particle density of the smoke is not known. Also the light strength interval measured on the photo cell, may vary between sensors. So it is decided to report the smoke density in percent, where 0% is no smoke and 100% is the maximum received light strength on the photo cell.

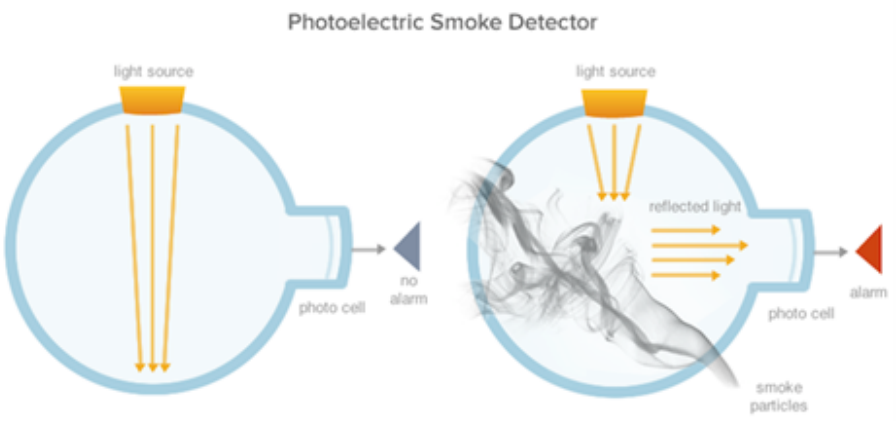


Figure 2.13: Photoelectric smoke detector

2.2.68.7.4 Detailed description: RF Signal Strength

The RF Signal Strength sensor type may report values using two different scales. While the dBm is a well-defined unit, the RSSI value represents a relative measurement where the internal sampling circuits and the actual sampling method is product specific.

CC:0031.05.05.11.010 The RSSI value MUST be reported in the range 0..100, where the value 100 represents the highest power level that can be measured.

2.2.69 Multilevel Switch Command Class, version 1

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control command class. Refer to Section 2.1.6.

2.2.69.1 Multilevel Switch Set Command

This command is used to set a multilevel value in a supporting device.

The device MAY apply a non-zero duration to the transition from one value to a new value.

Table 2.402: Multilevel Switch Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_SET (0x01)							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 2.403

Table 2.403: Multilevel Switch Set::Value

Value	Level	State
0 (0x00)	0%	Off
1..99 (0x01..0x63)	Lowest non-zero level .. 100%	On
...	*Reserved*	*Reserved*
255 (0xFF)	Restore most recent (non-zero) level.	On

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The above mapping of the Multilevel Switch Command Class Value to hardware levels allows a controlling device to control a mixed group of Binary Switch and Multilevel Switch devices via Basic Set commands. Devices implementing the Binary Switch CC turn On or Off while devices implementing the Multilevel Switch CC sets the specified level.

The values 0x00 and 0xFF are special values which MUST be treated as state control commands indicating "Off" and "On", respectively. A supporting device MUST restore the most recent (non-zero) value in response to the "On" state control command.

A device MAY implement up to 100 hardware levels (including 0). If a device implements less than 100 hardware levels, the hardware levels SHOULD be distributed uniformly over the entire range.

The mapping of command values to hardware levels MUST be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level. Refer to Section 2.1.6.2.

2.2.69.2 Multilevel Switch Get Command

This command is used to request the status of a multilevel device.

The Multilevel Switch Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.404: Multilevel Switch Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_GET (0x02)							

2.2.69.3 Multilevel Switch Report Command

This command is used to advertise the status of a multilevel device.

Table 2.405: Multilevel Switch Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_REPORT (0x03)							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 2.406.

Table 2.406: Multilevel Switch Report::Value

Value	Hardware Level	State
0 (0x00)	0%	Off
1..99 (0x01..0x63)	Lowest non-zero level .. 100%	On
...	*Reserved*	On
254 (0xFE)	Unknown	Unknown
255 (0xFF)	(100%)	On [Deprecated]

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The value 255 (0xFF) has been deprecated as it provides no information on the actual value in the device. A sending node MUST NOT use the value 255. A receiving node MUST interpret the value 255 as 100%.

The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command when a transition has ended.

2.2.69.4 Multilevel Switch Start Level Change Command

This command is used to initiate a transition to a new level.

Table 2.407: Multilevel Switch Start Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_START_LEVEL_CHANGE (0x04)							
Reserved	Up/ Down	Ignore Start Level	Reserved				
Start Level							

Up/Down (1 bit)

This field MUST specify the direction of the level change.

If the Up/Down bit is set to 0 the level change MUST be increasing.

If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

A receiving device SHOULD respect the start level if the Ignore Start Level bit is 0.

A receiving device MUST ignore the start level if the Ignore Start Level bit is 1.

A controlling device SHOULD set the Ignore Start Level bit to 1.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Start Level (8 bits)

The Start Level field MUST specify the initial level of the level change.

2.2.69.5 Multilevel Switch Stop Level Change Command

This command is used to stop an ongoing transition.

Table 2.408: Multilevel Switch Stop Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_STOP_LEVEL_CHANGE (0x05)							

2.2.70 Multilevel Switch Command Class, version 2

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control command class. Refer to Section 2.1.6

2.2.70.1 Compatibility considerations

A device supporting Multilevel Switch CC, version 2 MUST support Multilevel Switch CC, version 1.

Version 2 adds a "Duration" parameter to the Multilevel Switch Set and Multilevel Switch Start/Stop Level Change commands.

Commands not described in Version 2 remain unchanged from Version 1.

2.2.70.2 Multilevel Switch Set Command

This command is used to set a multilevel value in a supporting device.

Table 2.409: Multilevel Switch Set Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_SET (0x01)							
Value							
Duration							

Value (8 bits)

Refer to *Multilevel Switch Set Command*.

Duration (8 bits)

The Duration field MUST specify the time that the transition should take from the current value to the new target value.

A supporting device SHOULD respect the specified Duration value.

The encoding of the Duration field MUST be according to Table 2.9.

The factory default duration SHOULD be the same as the duration used for the *Multilevel Switch Set Command*.

2.2.70.3 Multilevel Switch Start Level Change Command

This command is used to initiate a transition to a new level.

Table 2.410: Multilevel Switch Start Level Change Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_START_LEVEL_CHANGE (0x04)							
Re-served	Up/ Down	Ignore Start Level	Reserved				
Start Level							
Duration							

Up/Down (1 bit)

This field MUST specify the direction of the level change.

If the Up/Down bit is set to 0 the level change MUST be increasing.

If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

A receiving device SHOULD respect the start level if the Ignore Start Level bit is 0.

A receiving device MUST ignore the start level if the Ignore Start Level bit is 1.

A controlling device SHOULD set the Ignore Start Level bit to 1.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Start Level (8 bits)

The Start Level field MUST specify the initial level of the level change.

Duration (8 bits)

The dimming rate to use MUST be calculated to match a transition from 0 to 99 during the time specified by the Duration field.

A supporting device SHOULD respect the specified Duration value.

For encoding of the duration value refer to Section 2.2.70.2.

2.2.71 Multilevel Switch Command Class, version 3

Warning: The Secondary Switch Type of the Multilevel Switch Command Class, version 3 has been DEPRECATED.

The Primary Switch Type 0x00, indicating "Not supported", of the Multilevel Switch Command Class, version 3 has been OBSOLETED.

The implementation of Secondary Switch Type functionality is NOT RECOMMENDED.

While the functionality related to the Secondary Switch Type of the Multilevel Switch Command Class, version 3 is deprecated, a supporting device claiming compliance with this version MUST implement support for the Multilevel Switch Supported Get Command.

For backwards compatibility reasons, a device MAY implement Secondary Switch Type functionality. It is however RECOMMENDED that Multi Channel Command Class support is also implemented if a device provides multiple controllable resources in the same physical entity.

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control Command Class. Refer to Section 2.1.6

Multilevel Switch CC, version 3 adds two-dimensional resource control capabilities to the Multilevel Switch Start/Stop Level Change commands and introduces two new commands for the discovery of those capabilities.

The Secondary Switch Type is intended for devices providing two-dimensional control, e.g. Window blinds, where the Primary Switch Type is used for Up/Down control and the Secondary Switch Type is used for controlling the slat tilt angle.

The Multilevel Switch Set, Get and Report commands MUST address the primary device functionality. The Multilevel Switch Start/Stop Level Change commands MUST manipulate the Primary Switch Type functionality based on the Up/Down parameter.

The Multilevel Switch Start/Stop Level Change commands MUST manipulate the Secondary Switch Type functionality based on the Inc/Dec parameter.

If the Secondary Switch Type is 0x00 (Undefined / Not supported), the Multilevel Switch Start/Stop Level Change (Inc/Dec) command parameter MUST be ignored.

2.2.71.1 Compatibility considerations

A device supporting Multilevel Switch CC, version 3 MUST support Multilevel Switch CC, version 2.

A device supporting Multilevel Switch CC, version 3 MUST implement the Primary Switch type.

Commands not described in Version 3 remain unchanged from Version 2.

2.2.71.2 Multilevel Switch Supported Get Command

This command is used to request the supported Switch Types of a supporting device.

The Multilevel Switch Supported Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.411: Multilevel Switch Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_SUPPORTED_GET (0x06)							

2.2.71.3 Multilevel Switch Supported Report Command

This command is used to advertise the supported Switch Types implemented by a supporting device.

Table 2.412: Multilevel Switch Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_SUPPORTED_REPORT (0x07)							
Reserved			Primary Switch Type				
Reserved			Secondary Switch Type				

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Primary Switch Type (5 bits)

The Primary Switch Type field MUST represent the primary device functionality.

Warning: Primary Switch Type 0x00, indicating "Not supported", has been OBSOLETED.

Previous revisions of this specification allowed that a device may (theoretically) be implemented with no primary device functionality, i.e. Primary Switch Type 0x00, and specified that such a device should indicate this situation by returning Multilevel Switch Report commands carrying the value 0xFE. The value 0xFE is incompatible with Versions 1 and 2 of the Multilevel Switch Command Class and MUST NOT be used in a Multilevel Switch Report.

The Primary Switch Type MUST comply with Table 2.413.

A supporting device MUST implement the Primary Switch type.

The Primary Switch Type SHOULD be 0x02 (Up/Down).

The Primary Switch Type MUST NOT be 0x00 (Undefined).

Table 2.413: Encoding of Primary and Secondary Switch Types

Switch Value Type	0x00 (Direction/Endpoint A)	0x63/0xFF (Direction/Endpoint B)
0x00	Undefined / Not supported (Secondary only)	
0x01	Off	On
0x02	Down	Up
0x03	Close	Open
0x04	Counter-Clockwise	Clockwise
0x05	Left	Right
0x06	Reverse	Forward
0x07	Pull	Push
0x08-0x1F	Reserved	

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Secondary Switch Type (5 bits)

The Secondary Switch Type field MUST represent the secondary device functionality. The Secondary Switch Type MUST comply with Table 2.413.

A supporting device MAY implement the Secondary Switch type.

2.2.71.4 Multilevel Switch Start Level Change Command

This command is used to initiate a transition to a new level.

The Multilevel Switch Command Class, version 3 adds a "Secondary Switch Inc/Dec" and a "Secondary Switch Step Size" field to this command to support motor controlled devices featuring two-dimensional motion.

Table 2.414: Multilevel Switch Start Level Change Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_START_LEVEL_CHANGE (0x04)							
Primary Switch Up/Down		Ignore Start Level	Secondary Switch Inc/Dec		Reserved		
Start Level							
Duration							
Secondary Switch Step Size							

Primary Switch Up/Down (2 bits)

The Up/Down field MUST be used for manipulating the primary device functionality.

This field MUST be encoded according to Table 2.415.

Table 2.415: Encoding of the Up/Down field

Value	Description	Details
0x00	Up	Increase level for Primary Switch Type
0x01	Down	Decrease level for Primary Switch Type
0x02	*Reserved*	
0x03	No Up/Down motion	Maintain current level for Primary Switch Type

Ignore Start Level (1 bit)

A receiving device SHOULD respect the start level if the Ignore Start Level bit is 0.

A receiving device MUST ignore the start level if the Ignore Start Level bit is 1.

Secondary Switch Inc/Dec (2 bits)

The Inc/Dec field MUST be used for controlling the secondary device functionality.

If the Secondary Switch Type is 0x00 (Undefined / Not supported), the Inc/Dec field MUST be ignored.

This field MUST be encoded according to Table 2.416.

Table 2.416: Encoding of the Inc/Dec field

Value	Description	Details
0x00	Increment	Increase level for Secondary Switch Type
0x01	Decrement	Decrease level for Secondary Switch Type
0x02	*Reserved*	
0x03	No Inc/Dec	Maintain current level for Secondary Switch Type

As this field defines the value "00" of two previously reserved bits as the code "Increment", a supporting device implementing Multilevel Switch CC, version 3 MUST correctly identify the Multilevel Switch

Start Level Change command to be version 3 (by the presence of the <Secondary Switch Step Size> field) before interpreting the <Secondary Switch Inc/Dec> field. Failing to do so will cause a device to start incrementing the Secondary Switch Type in response to version 1 and version 2 variants of the Multilevel Switch Start Level Change command.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Primary Switch Start Level (8 bits)

The Start Level field MUST be used for controlling the primary device functionality.

Duration (8 bits)

This field is unchanged from version 2.

Secondary Switch Step Size (8 bits)

The Step Size field MAY be used for controlling a secondary device functionality.

If the Secondary Switch Type is 0x00 (Undefined / Not supported), the Step Size field MUST be ignored. If the Secondary Switch Type is not 0x00, the Step Size field MUST be used for controlling the secondary device functionality.

If the Inc/Dec field is set to 3 (No Inc/Dec), the Step Size field MUST be set to 0.

This field MUST carry a value in the range {0x00..0x63, 0xFF}. All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

A receiving device MUST accept any of the values in the above range. 0x01 MUST represent the lowest non-zero level and 0x63 MUST represent the highest level

A device MAY implement 100 hardware levels (including 0). If a device implements less than 100 hardware levels, the mapping to hardware levels SHOULD be distributed equally over the entire range of 1..99 (0x01..0x63).

The mapping of command values to hardware levels MUST be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level.

An implementation MUST interpret the combined Inc/Dec and Step Size fields as outlined in Table 85.

Table 2.417: Interpretation of the Inc/Dec and Step Size fields

(Secondary Switch Type)	Inc/Dec	Step Size	Interpretation
(undefined)	*(ignore)*	*(ignore)*	Secondary Switch Type is undefined -> ignore fields
0x01..0x07	No Inc/Dec	*(ignore)*	No Inc/Dec -> Maintain current Secondary Switch level
0x01..0x07	Increment	x	Increase Secondary Switch level by x steps
0x01..0x07	Decrement	y	Decrease Secondary Switch level by y steps

The requested level change SHOULD take the time specified by the Duration field.

2.2.72 Multilevel Switch Command Class, version 4

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control Command Class. Refer to Section 2.1.6.

2.2.72.1 Compatibility considerations

Version 4 adds reporting of target value and duration.

A device supporting Multilevel Switch CC, Version 4 MUST support Multilevel Switch CC, Version 3.

A device receiving a V1 Multilevel Set command MAY apply a factory default duration to the transition to a new value.

2.2.72.2 Multilevel Switch Report Command

This command is used to advertise the status of a multilevel device.

Table 2.418: Multilevel Switch Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL (0x26)							
Command = SWITCH_MULTILEVEL_REPORT (0x03)							
Current Value							
Target Value							
Duration							

Current Value (8 bits)

The encoding of the Value field MUST be according to Table 2.406.

The device may be queried for its current value while in transition to a new value. The response to a Get command SHOULD be the current value of the device hardware.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command when a transition has ended.

Target Value (8 bits)

The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition.

The encoding of the Target Value field MUST be according to Table 2.406.

If queried after receiving a Set command, the Target Value field MUST advertise the target value specified in the Set command. The Target Value may change at a later time due to local control or a Multilevel Switch Stop Level Change command.

If the device is in a motion controlled transition, the Target Value field MUST advertise the value 0x00 or 0x63.

Duration (8 bits)

The Duration field SHOULD advertise the time needed to reach the Target Value at the actual transition rate. The encoding of the Duration field MUST be according to Table 2.9.

2.2.73 Multilevel Toggle Switch Command Class, version 1

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Multilevel Switch Command Class.

If implementing this command class, it is RECOMMENDED that the Multilevel Switch Command Class is also implemented.

The Multilevel Toggle Switch Command Class is used for multilevel toggle-style actuator devices.

2.2.73.1 Multilevel Toggle Switch Set Command

This command is used to set the level in a device that supports the multilevel switch functionality.

Table 2.419: Multilevel Toggle Switch Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL (0x29)							
Command = SWITCH_TOGGLE_MULTILEVEL_SET (0x01)							

2.2.73.2 Multilevel Toggle Switch Get Command

This command is used to request the state of the load controlled by the device.

The Multilevel Toggle Switch Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.420: Multilevel Toggle Switch Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL (0x29)							
Command = SWITCH_TOGGLE_MULTILEVEL_GET (0x02)							

2.2.73.3 Multilevel Toggle Switch Report Command

This command is used to advertise the level of a toggle switch.

Table 2.421: Multilevel Toggle Switch Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL (0x29)							
Command = SWITCH_TOGGLE_MULTILEVEL_REPORT (0x03)							
Value							

Value (8 bits)

The value MAY be 0x00 (off/disable) or 0xFF (on/enable).

The field MAY carry values from 1 to 99.

2.2.73.4 Multilevel Toggle Switch Start Level Change Command

This command is used to inform a multilevel toggle switch, that it should start changing the level. The speed that the switch increases or decreases the level with is implementation specific.

Table 2.422: Multilevel Toggle Switch Start Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL (0x29)							
Command = SWITCH_TOGGLE_MULTILEVEL_START_LEVEL_CHANGE (0x04)							
Roll Over	Reserved	Ignore Start Level	Reserved				
Start Level							

Roll Over (1 bit)

If the Roll Over bit is set to 0, the switch SHOULD stop when reaching the max or min level. If the roll over bit is set to 1, the switch SHOULD continually increase and decrease the level until otherwise instructed.

Ignore Start Level (1 bit)

If the Ignore Start Level bit is set to 0 the switch SHOULD use the start level specified in the Command. If field is set to 1 the switch SHOULD start from the actual level in the device.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Start Level (8 bits)

The Start Level field contains the initial level that the switch should assume when it start to change the level.

2.2.73.5 Multilevel Toggle Switch Stop Level Change Command

This command is used to inform a multilevel toggle switch, that it should stop changing the level.

Table 2.423: Multilevel Toggle Switch Stop Level Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL (0x29)							
Command = SWITCH_TOGGLE_MULTILEVEL_STOP_LEVEL_CHANGE (0x05)							

2.2.74 Notification Command Class, version 3-8

Warning: THIS COMMAND CLASS VERSIONS 3-7 HAVE BEEN DEPRECATED

CC:0071.03.00.13.001 A device MAY implement version 3 to 7, but it is RECOMMENDED that new implementations comply with Notification Command Class, version 8

CC:0071.03.00.12.001 Pull Mode has been deprecated; it is RECOMMENDED that new implementations support Push Mode

CC:0071.03.00.12.002

The Notification Command Class is used to advertise events or states, such as movement detection, door open/close or system failure. The Notification Command Class supersedes the Alarm Command Class.

2.2.74.1 Terminology

Sensors may be designed for several purposes. A multilevel sensor advertises measurements or readings. A **notification sensor** sends **event** or **state notifications**.

This Command Class is used for notification sensors. The Multilevel Sensor Command Class is used for multilevel sensors.

Notifications are categorized into logical groups called **Notification Types**. A Notification is denoted with its type and event/state: {Notification Type::event/state}. A node may send Notifications from several Notification Types.

An event only has a meaning in the moment it happens. An event does not indicate the value of a state variable. An example is given in Figure 2.14.

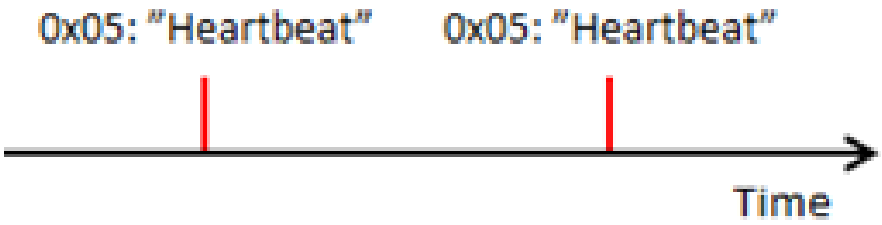


Figure 2.14: Event notifications inform only about instantaneous situations

A **state variable** may assume two or more states. Some state variables are returned to their idle state via a generic "State idle" Notification.

For example, a controlling node receiving a {Smoke alarm::Smoke detected} Notification will consider that the state has not changed until it receives a {Smoke alarm::State idle(Smoke detected)} Notification. An illustration is given in Figure 2.15.

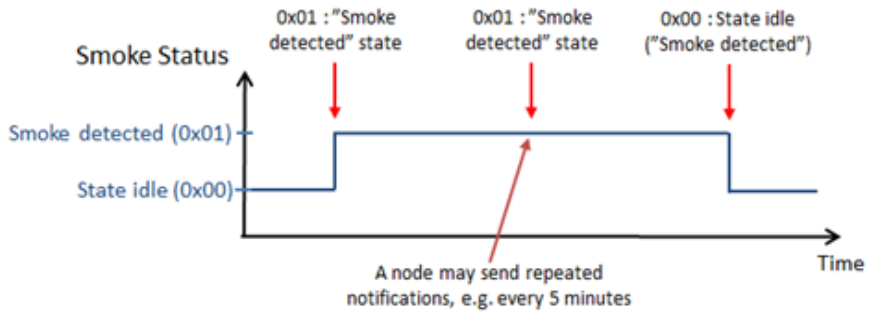


Figure 2.15: Binary state variable with generic "State idle" Notification

The "State idle" notification may also be used to return a multi-value state variable to its idle state. An example is given in Figure 2.16.

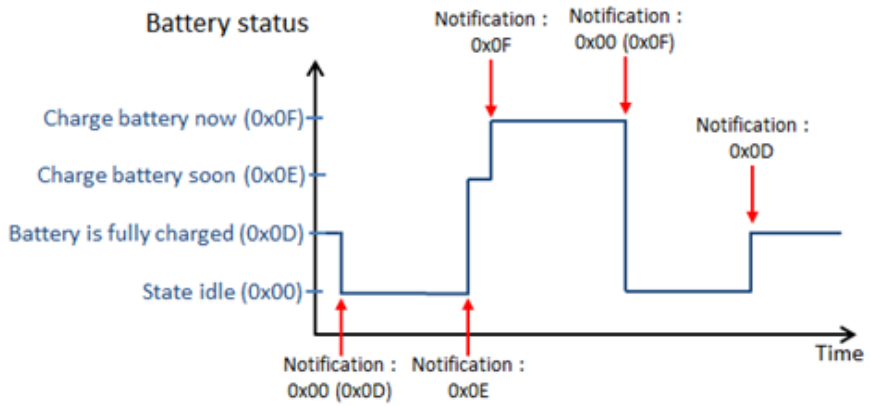


Figure 2.16: Multi-value state variable with generic "State idle" Notification

Some state variables use a specific Notification defined for each state change. Figure 2.17 shows an example of such a state variable.

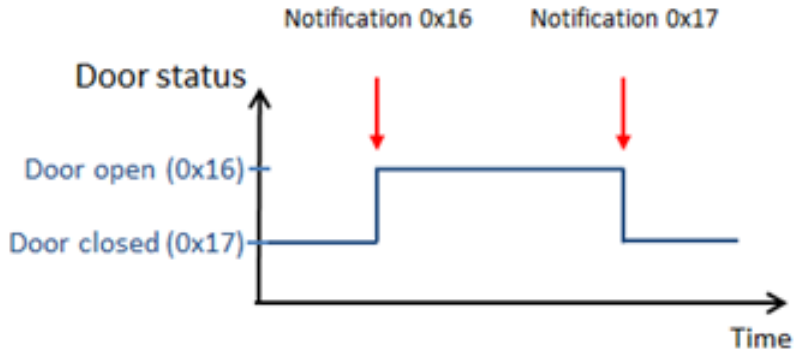


Figure 2.17: Binary state variable with specific idle state Notification

Some notifications are complemented with **event/state** parameters. For example, the {System::System software failure} Notification may be accompanied with the manufacturer's failure/error codes as event parameters.

Event/state parameters may also affect state variables and their states. For example, the Notification {Irrigation::Schedule started} takes a 1-byte parameter which identifies the Schedule ID that is started. In that case, each state variable is identified using the state parameters, thus creating a state variable array.

An illustration is given in Figure 2.18, where the same Notifications with different parameters advertise the state of different state variables.

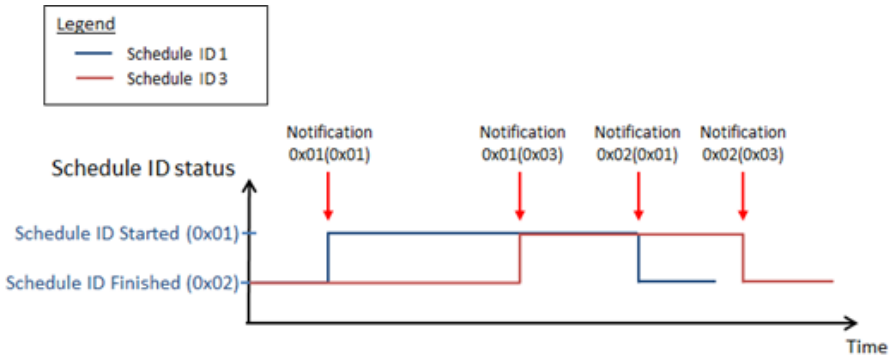


Figure 2.18: State variable array defined by additional Event/State Parameters

A complete list of Notifications (including state variables, event parameters, etc.) is given in [36].

A notification node operates either in **Push** or in **Pull** mode.

A Push node sends **unsolicited** Notifications. The transmission of unsolicited notifications may be **disabled** or **enabled**. When enabled, unsolicited Notifications are transmitted via an Association Group. It is not possible to subsequently retrieve Notifications issued by a Push node.

A Pull sensor collects events and state changes in a queue of pending Notifications. Notifications are retrieved one by one from the Pull sensor queue via the Notification Get Command. The Pull sensor advertises that its queue is empty when all Notifications have been retrieved.

A persistent Pull Notification is not removed from the Pull sensor queue until it is actively cleared by a controlling node.

2.2.74.2 Compatibility considerations

2.2.74.2.1 Notifications and Command Class version

CC:0071.03.00.21.001 New Notification Types and Notifications have been added to each new version of this Command Class. A node MUST implement as a minimum the Notification Command Class version associated with the Notifications it sends. The minimum required version for each Notification is specified in [36].

2.2.74.2.2 Push mode requirements

CC:0071.03.00.22.001 A node supporting the Notification Command Class SHOULD implement Push mode.

CC:0071.03.00.21.002 A Push node MUST implement the Association Command Class.

CC:0071.03.00.22.002 A Push node SHOULD advertise Association Groups through the Association Group Information (AGI) Command Class.

CC:0071.03.00.21.003 A Z-Wave Plus Push node MUST:

- Provide all supported Notifications via the Lifeline Association group.

- Advertise Association Groups through the Association Group Information (AGI) Command Class.

2.2.74.2.3 Pull mode requirements

Earlier text revisions presented inconsistencies and undefined behaviors for Pull nodes. Thus, existing
CC:0071.03.00.22.003 Pull nodes may behave differently than expected by a controlling node. A node supporting the Notification Command Class SHOULD NOT implement Pull mode.

CC:0071.03.00.23.001 A Pull node MAY reorder Notifications according to priority so that the first detected event is not the
CC:0071.03.00.22.004 first to be reported. A Pull node SHOULD NOT reorder states of the same state variable in the event queue. For example, the "Program in progress" and "Program completed" Notifications SHOULD stay in the same order.

CC:0071.03.00.22.005 A Pull node having its queue full SHOULD remove the oldest notification entry from the queue.

CC:0071.03.00.22.006 Persistent notifications SHOULD carry a sequence number.

2.2.74.2.4 Multi Channel considerations

CC:0071.03.00.21.004 If several End Points within a Multi Channel device support the Notification Command Class, they MUST all operate in the same mode (i.e. either all End Points operate in Push mode or all End Points operate in Pull mode).

2.2.74.2.5 Multi Channel Push nodes

CC:0071.03.00.23.002 While End Points MAY send identical notifications via the Root Device Lifeline Group, the Root
CC:0071.03.00.21.005 Device MUST NOT send identical Notifications on behalf of multiple End Points. Illustrations are given in Figure 2.19 and Figure 2.20.

Figure 2.19: Multi Channel device aggregating Notifications (Lifeline group)

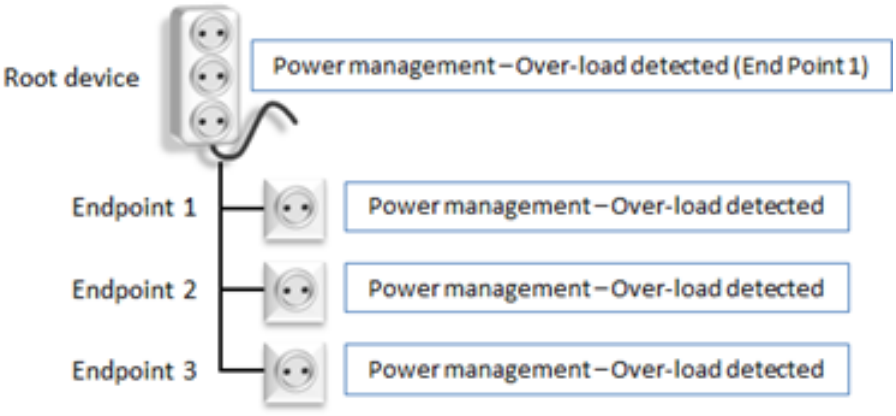


Figure 2.20: Multi Channel device with End Point notification overlap (Lifeline group)

A Multi Channel node with identical End Points issuing the same binary state notification with generic
CC:0071.03.00.23.003 state idle Notification MAY be an exception and issue binary Notifications representing the state all End Points.

CC:0071.03.00.21.008 If doing so, the Root Device MUST issue the active state notification if any of the End Points has the active binary state and MUST issue the event inactive Notification when all End Points are back to the "State idle".

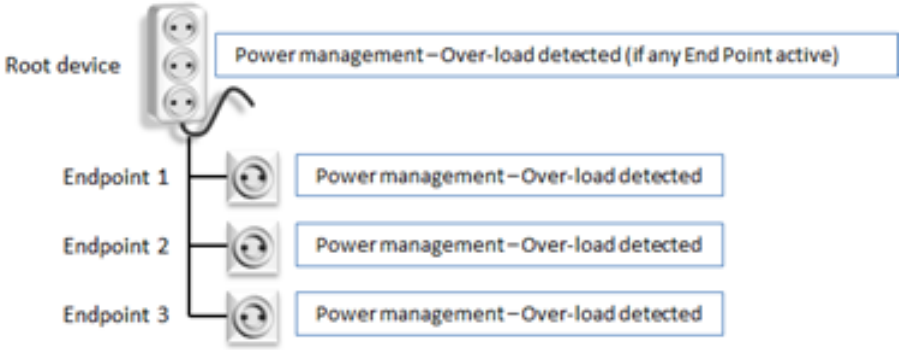


Figure 2.21: Notification representing all End Points for Binary State with generic State Idle notification

CC:0071.03.00.21.006 Z-Wave Plus Multi Channel devices MUST send End Point Push notifications via the Root Device Lifeline Association Group (if notifications are not identical).

CC:0071.03.00.22.007 A controlling node SHOULD create a Multi Channel Association to the Lifeline group of a supporting node. Thus Notifications will be sent Multi Channel encapsulated via the Root Device Lifeline group, allowing End Points to send identical Notifications.

Individual End Point Notifications can be enabled/disabled by sending a Multi Channel encapsulated Notification Set Command to an End Point, even when End Point Notifications are actually sent via the Root Device Lifeline group.

2.2.74.2.6 State idle

Every Notification Type has the same generic Notification 0x00 "State idle", (also known as "Event Inactive" in older specification text) which was introduced in version 4. The Notification allows the device to advertise that a state variable returned to its idle state.

CC:0071.04.00.23.001 A Push sensor MAY send repeated state Notifications without sending any "State idle" Notification to indicate that a given state is still active. For instance, a smoke sensor can send a "Smoke Detected" Notification every five minutes to indicate that the state variable has not returned to idle (refer to Figure 16.).

Notification Command Class, version 8 increases the requirement level for the use of the "State idle"
CC:0071.08.00.21.001 Notification from OPTIONAL to REQUIRED. Refer to Section 2.2.74.2.12.

CC:0071.04.00.22.001 A supporting node SHOULD NOT send a "State idle" notification for an event or for a state variable to which state idle does not apply. Refer to [36] for state variables to which the "State idle" notification applies

2.2.74.2.7 Version 3 [DEPRECATED]

The Notification Command Class version 3 is an extension of the Alarm Command Class version 2 and adds the following:

- Additional Notifications (refer to [36])
- Interview process includes Events (Event Supported Get / Event Supported Report)
- Sequence field added for collection management of reports
- The queue empty status (=0xFE) for Pull nodes. A Pull node can now advertise Notification Status = "no pending notifications"

Commands not mentioned in this version are unchanged from Alarm Command Class version 1 and/or Alarm Command Class version 2.

The CC identifier for Notification CC V3 is the same as the Alarm CC V1 and V2. However, the Notification Command Class is not fully backwards compatible with the Alarm Command Class, versions 1 and 2. Clarifications for ensuring backwards compatibility with version 1 and 2 are given in version 4 of this Command Class.

An implementation supporting Alarm CC V1 fields MUST map proprietary alarm types and levels to a similar Notification Type and Notification CC V3 where possible. In addition, all Alarm CC V1 alarm types and levels MUST be described in the product manual.

CC:0071.03.00.21.007

2.2.74.2.8 Version 4 [DEPRECATED]

Version 4 of this Command Class introduces the following:

- Additional Notifications (refer to [36])
- Event/State = 0x00 (State idle) for a given Notification Type to indicate that all state variables of that Notification Type returned to idle
- The Zensor Net Source Node ID field has been discontinued. It is now a reserved field.
- **Clarification to expected behavior in regards to:**
  - V1 and V2 Alarm Get Command handling
  - Notification Status field description
  - Notification Type = 0xFF, "Return first detected notification on supported list"
  - Event = 0xFE in Event Supported Report Command, which must not be advertised.

The support for state idle (event/state=0x00) cannot be advertised in the Event Supported Get Command and version 4 nodes will appear to use reserved values for version 3 controlling nodes.

CC:0071.04.00.21.001 A version 4 node MUST comply with the rules outlined in Table 2.424 and Table 2.425.

Table 2.424 outlines the required behavior when receiving an Alarm Get command, Version 1.

Table 2.425 outlines the required behavior when receiving an Alarm Get command, Version 2.

The version of an Alarm Get Command can be determined from the command length.

Table 2.424: Required behavior when receiving Alarm Get, version 1

Received Command: V1, ALARM_GET (Alarm Type = x)	
V1 Alarm Type supported (x)?	Response
YES	V1, ALARM_REPORT (Alarm Type = x, Alarm Level = level)
NO	V1, ALARM_REPORT (Alarm Type = 0x00, Alarm Level = 0x00)

Table 2.425: Required behavior when receiving Alarm Get, version 2

Received Command: V2, ALARM_GET (Alarm Type = x, Z-Wave Alarm Type = y)		
Z-Wave Alarm Type supported (y)?	V1 Alarm Type supported (x)?	Response
NO	NO	NO RESPONSE
NO	YES	V2, ALARM_REPORT (Alarm Type = x Alarm Level = level Reserved = 0x00 Z-Wave Alarm Status = 0x00 Z-Wave Alarm Type = 0x00 Z-Wave Alarm Event = 0x00 Number of Event Param = 0x00)
YES or y = 0xFF	YES/NO	CC:0071.04.00.21.002 Notifications newer than V2, idle states or empty queues MUST be represented with the unknown notification (Notification event/state field set to 0xFE). CC:0071.04.00.21.003 If no state is detected (push nodes) or no event/state queued (pull nodes), the Notification Type MUST be set to one of the supported Notification Type in response to a Get (Type = 0xFF) CC:0071.04.00.21.004 **Push nodes:** A supporting node MUST return the current states compatible with V2. Table 2.427 shows a Push node example returning responses to different V2 Alarm Get Commands. CC:0071.04.00.21.005 **Pull nodes:** CC:0071.04.00.21.006 A supporting node MUST return only V2 notifications from its queue when receiving a V2 Alarm Get Command. A supporting node MUST set the status to 0x00 and event to 0xFE when its queue is empty. Table 2.428 shows a Pull node example returning responses to different V2 Alarm Get Commands.

Table 2.427 and Table 2.428 show the responses of a node supporting Notification Types and Notifications described in Table 2.426:

Table 2.426: V2 Alarm Get node capabilities example

Node capabilities	
Notification Types	Notifications
0x01: Smoke Alarm	0x01: Smoke detected 0x03: Smoke alarm test 0x06: Alarm silenced
0x03: CO2 Alarm	0x01: Carbon dioxide detected

Table 2.427: Push node responses to V2 Alarm Get (example)

Received Get		**Current states at the receiving node**			**Returned response**		
V1 Alarm Type	Type	Smoke Alarm:: Sensor status	Smoke Alarm:: Alarm status	CO2 Alarm:: Sensor status	Type	Status	Event/ State
-*	0x01	Idle	Idle	-	0x01	0x00 or 0xFF	0xFE

continues on next page

Table 2.427 – continued from previous page

Received Get		**Current states at the receiving node**			**Returned response**		
-	0x01	Smoke detected (V2)	Idle	-	0x01	0x00 or 0xFF	0x01
-	0x01	Smoke detected (V2)	Alarm silenced (V8)	-	0x01	0x00 or 0xFF	0x01
-	0x01	Idle	Smoke alarm test (V3)	-	0x01	0x00 or 0xFF	0xFE
0x00	0x02	-	-	-	No Report		
0x01	0x02	-	-	-	No Report, unless V1 alarm is supported. Refer to *Required behavior when receiving Alarm Get, version 2*		
-	0xFF	Idle	Idle	Idle	0x01 or 0x03**	0x00 or 0xFF	0xFE
-	0xFF	Smoke detected (V2)	Idle	Idle	0x01	0x00 or 0xFF	0x01
-	0xFF	Smoke detected (V2)	Alarm silenced (V8)	Idle	0x01	0x00 or 0xFF	0x01
-	0xFF	Idle	Smoke alarm test (V3)	Idle	0x01	0x00 or 0xFF	0xFE
-	0xFF	Smoke detected (V2)	Idle	Carbon dioxide detected (V2)	0x01 or 0x03**	0x00 or 0xFF	0x01

*) The symbol '-' indicates that the value has no impact on the V2 fields of the returned response

**) It is up to the node to decide which Notification Type to report.

Table 2.428: Pull node responses to V2 Alarm Get (example)

Received Get		**Before returning a report**	**Returned response**			**After returning a report**
V1 Alarm Type	Type	Node's queue	Type	Status	Event/ State	Node's queue
-*	0x01	Empty	0x01	0x00	0xFE	Empty
-	0x01	Carbon dioxide detected (V2) Smoke detected (V2)	0x01	0x00	0x01	Carbon dioxide detected (V2)
-	0x01	Alarm silenced (V8) Smoke detected (V2)	0x01	0x00	0x01	Alarm silenced (V8)
-	0x01	Alarm silenced (V8)	0x01	0x00	0xFE	Alarm silenced (V8)

continues on next page

Table 2.428 – continued from previous page

Received Get		**Before returning a report**	**Returned response**			**After returning a report**
0x00	0x02	•	No Report			-
0x01	0x02	•	No Report, unless V1 alarm is supported. Refer to *Required behavior when receiving Alarm Get, version 2*			-
-	0xFF	Empty	0x01 or 0x03**	0x00	0xFE	Empty
-	0xFF	Carbon dioxide detected (V2) Smoke detected (V2)	0x03	0x00	0x01	Smoke detected (V2)
-	0xFF	Alarm silenced (V8)	0x01	0x00	0xFE	Alarm silenced (V8)

*) The symbol '-' indicates that the value has no impact on the V2 fields of the returned response

**) It is up to the node to decide which Notification Type to report.

2.2.74.2.9 Version 5 [DEPRECATED]

Version 5 of this Command Class introduces additional Notifications (refer to [36]) and the event/state parameters for the special purpose State idle 0x00, allowing to specify which state variable returned to idle.

A version 4 controlling node will conclude that all state variables have returned to idle when a version 5 node advertises that a single state variable has returned to idle.

2.2.74.2.10 Version 6 [DEPRECATED]

Version 6 of this Command Class introduces additional Notifications (refer to [36])

Version 6 clarifies how to use the 'User Code Report' Event Parameter. The User Code Report Command is used as Event Parameter for the Notification {Access Control::Keypad Lock/Unlock Operation} Command.

An example of Notification Event/State Parameter encapsulation is given in Section 2.2.74.6.1.

2.2.74.2.11 Version 7 [DEPRECATED]

Version 7 of this Command Class introduces additional Notifications (refer to [36]).

2.2.74.2.12 Version 8

CC:0071.08.00.22.001 It is RECOMMENDED that new nodes support version 8 of this Command Class.

Requirements for backwards compatibility with version 2 nodes introduced in version 4 have been found to be ambiguous and challenging to be observed correctly by newer nodes. Therefore, it is CC:0071.08.00.23.001 OPTIONAL for a version 8 node to comply with Table 2.425 when receiving a V2 Get Command.

Version 8 of this Command Class introduces additional Notifications (refer to [36]).

CC:0071.08.00.21.002 Version 8 increases the requirement level for the use of the "State idle" Notification from OPTIONAL to REQUIRED.

CC:0071.08.00.21.003 Supporting nodes implementing version 8 or newer of the Notification Command Class MUST issue a "State idle" Notification when a state variable returns to idle. For instance, the {Smoke alarm::Smoke CC:0071.08.00.21.004 Detected} Notification MUST be followed by a {Smoke alarm::State idle(Smoke detected)} Notification when the smoke is no longer detected by the sensor. (refer to Figure 16).

2.2.74.3 Interoperability considerations

2.2.74.3.1 Push nodes Basic control

CC:0071.03.00.32.001 For Push nodes, it is RECOMMENDED to implement additional association groups for relevant Notifications, which issue a Basic Set Command. It allows configuring a device to control other nodes directly, e.g., turn on lights when motion is detected and turn off when there is no more motion.

2.2.74.3.2 Event flood

CC:0071.03.00.32.002 Certain sensors may trigger repeatedly within a short amount of time. It is RECOMMENDED to implement timers to arbitrate the transmission of notifications. Illustrations are given in Figure 23 and Figure 24.

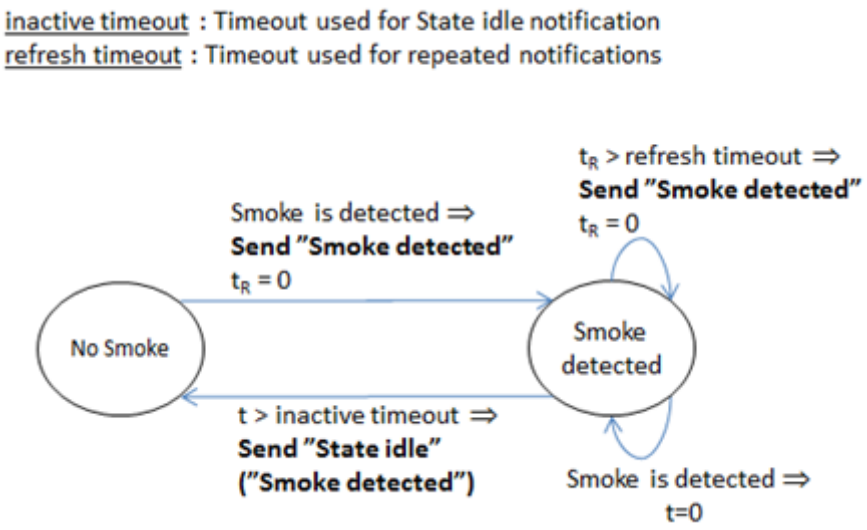


Figure 2.22: Recommended timer mechanism for sending notifications (1)

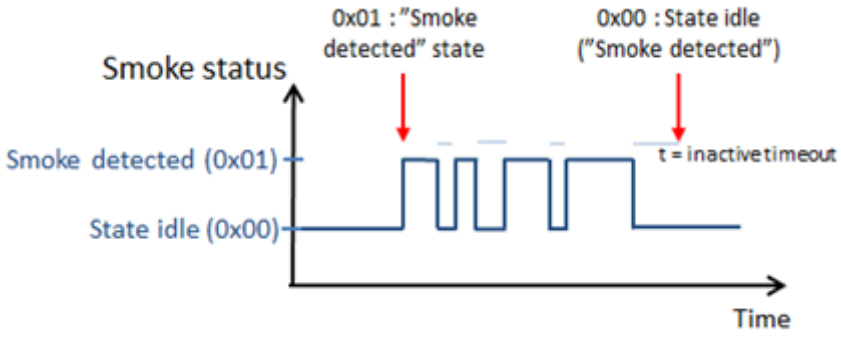


Figure 2.23: Recommended timer mechanism for sending notifications (2)

2.2.74.4 Notification Set Command

Push nodes:

This command is used to enable or disable the unsolicited transmission of a specific Notification Type.

Pull nodes:

This command is used to clear a persistent Notification in the notification queue.

Table 2.429: Notification Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NOTIFICATION (0x71)							
Command = NOTIFICATION_SET (0x06)							
Notification Type							
Notification Status							

Notification Type (8 bits)

This field is used to specify a Notification Type. Assigned values are defined in [36].

CC:0071.03.06.11.001 A sending node MUST specify a Notification Type that is supported by a receiving node.

CC:0071.03.06.11.002 A receiving node MUST ignore the command if this field is set to a non-supported Notification Type or if this field set to 0xFF.

Notification Status (8 bits)

Push nodes:

This field is used to set the Status of a Notification Type.

This field MUST comply with Table 2.430.

Table 2.430: Notification Set::Notification Status (push mode)

Value	Description	Version
0x00	Unsolicited messages MUST be disabled for the specified Notification Type	2
0xFF	Unsolicited messages MUST be enabled for the specified Notification Type	2

CC:0071.03.06.11.004
CC:0071.03.06.11.005

CC:0071.03.06.11.006 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0071.03.06.13.001
CC:0071.03.06.11.007
CC:0071.03.06.11.008

A receiving node MAY deny the deactivation of a specific Notification Type. In that case, the receiving node MUST respond to this command with an Application Rejected Request Command. Thus, if a node can deny the deactivation of a Notification Type, the node MUST implement the Application Status CC. A sending node should be aware that a receiving node may return a response to the Set Command and thus SHOULD apply a back-off timer before sending a subsequent command.

Pull nodes:

CC:0071.03.06.11.009 A sending node MUST set this field to 0x00 to indicate that a persistent Notification for the specified Notification Type MUST be cleared in the receiving node's queue.

CC:0071.03.06.11.00A All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.74.5 Notification Get Command

Push nodes:

This command is used to request if the unsolicited transmission of a specific Notification Type is enabled.

Some supporting nodes will also advertise a current state in response to this command.

Pull nodes:

This command is used to retrieve the next Notification from the receiving node's queue.

CC:0071.03.04.11.001 The Notification Report Command MUST be returned in response to this command unless this command is to be ignored. Refer to the fields description.

CC:0071.03.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0071.03.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.431: Notification Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NOTIFICATION (0x71)							
Command = NOTIFICATION_GET (0x04)							
V1 Alarm Type							
Notification Type							
Notification Event / State							

V1 Alarm Type (8 bits)

The use of this field depends on the V1 Alarm field advertised in the Alarm Type Supported Report Command.

A sending node MAY specify a V1 Alarm Type if the receiving node supports the actual alarm type.

A sending node MUST specify the value 0x00 if the receiving node does not support V1 alarms.

CC:0071.03.04.12.001 The receiving node behavior SHOULD comply with Table 2.424 and Table 2.425.

Notification Type (8 bits)

This field is used to specify a Notification Type. Assigned values are defined in [36].

CC:0071.03.04.11.004 This field MUST be set to a Notification Type that is supported by the receiving node or to 0xFF.

CC:0071.03.04.11.005 A receiving node MUST ignore the command if this field is different than 0xFF and set to a non-supported Notification Type.

Push nodes:

CC:0071.03.04.12.002

The value 0xFF indicates to the receiving node that it MUST select a supported Notification Type and SHOULD advertise the current state of one of its state variables within the chosen Notification Type.

CC:0071.03.04.11.006 A supported Notification Type value indicates that the receiving node MUST advertise the unsolicited transmission status for the requested Notification Type and MAY advertise the current state of one of its state variables within the Notification Type.

Pull nodes:

CC:0071.03.04.11.007 The value 0xFF indicates to the receiving node that it MUST retrieve the next Notification in its queue.

CC:0071.03.04.12.003 A Notification Type value indicates to the receiving node that it SHOULD retrieve the next Notification in its Notification queue matching the specified Notification Type value.

Notification Event / State (8 bits)

This field is used to optionally specify a Notification Event/State within the Notification Type. Assigned values are defined in [36]. This field allows receiving nodes to differentiate between V2 Alarm Get Command and V3 or newer Notification Get Command.

CC:0071.03.04.11.008 If the Notification Type is set to 0xFF, this field MUST be set to 0x00 by a sending node and SHOULD be ignored by a receiving node.

Push nodes:

CC:0071.03.04.13.001 A sending node MAY set this field to 0x00.

CC:0071.03.04.13.002 A sending node MAY specify a value that is supported by the receiving node within the specified Notification Type. In this case, a receiving node:

- MAY advertise a current state related to the indicated state - MAY return the same value in response to a supported Notification Event.

CC:0071.03.04.11.009 A receiving node MUST return the “Unknown notification” value (0xFE) in response to a non-supported Notification Event.

Pull nodes:

CC:0071.03.04.12.004 This field SHOULD be ignored by a receiving node.

2.2.74.6 Notification Report Command

Push nodes:

This command is used for two purposes:

- If this command is returned in response to a Notification Get Command, this command advertises if the unsolicited transmission of the advertised Notification Type is enabled and optionally advertises a currently active state.
- If this command is sent unsolicited, it advertises an event or state Notification.

The use of the command’s fields is the same in both cases.

Pull nodes:

This command is used by a sending node to return a Notification from its queue or indicate that its queue is empty.

Table 2.432: Notification Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NOTIFICATION (0x71)							
Command = NOTIFICATION_REPORT (0x05)							
V1 Alarm Type							
V1 Alarm Level							
Reserved							
Notification Status							
Notification Type							
Notification Event / State							
Sequence	Reserved			Event / State Parameters Length			
Event /State Parameter 1 (optional)							
...							
Event / State Parameter N (optional)							
Sequence Number (optional)							

Reserved

CC:0071.03.05.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

V1 Alarm Type (8 bits) & V1 Alarm Level (8 bits)

These fields carry the proprietary Alarm Type and Alarm Level fields originally introduced with Alarm
CC:0071.03.05.11.002 Command Class, Version 1. V1 Alarm Type and V1 Alarm Level fields MUST be specified in the product manual.

CC:0071.03.05.11.003 If the V1 Alarm Type is not supported, these fields MUST be set to 0x00.

Notification Status (8 bits)

Push nodes:

This field is used to advertise the status of the Notification Type indicated in this command.

CC:0071.03.05.11.004 This field MUST comply with Table 2.433.

Table 2.433: Notification Report::Notification Status (push nodes)

Value	Description	Version
0x00	Unsolicited transmissions are disabled for the specified Notification Type	2
0xFF	Unsolicited transmissions are enabled for the specified Notification Type	2

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Pull nodes:

This field is used to advertise the status of the sending node's Notification queue.

CC:0071.03.05.11.005 This field MUST comply with Table 2.434.

Table 2.434: Notification Report::Notification Status (pull nodes)

Value	Description	Version
0x00	This command carries a valid Notification returned from the queue. There may be more Notifications queued up. This Notification MAY be persistent.	2
0xFE	This command does not carry any valid Notification and the event queue is empty	3

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Notification Type (8 bits)

This field is used to advertise the Notification Type. Assigned values are defined in [36].

CC:0071.03.05.11.006 A sending node MUST set this field to the Notification Type of the actual Notification.

Notification Event / State (8 bits)

This field is used to specify a Notification Event/State for the advertised Notification Type. Assigned values are defined in [36].

CC:0071.03.05.11.007 A node advertising a Notification MUST set this field to the actual Notification event/state.

Sequence (1 bit)

This field is used to advertise the presence of the "Sequence Number" field.

CC:0071.03.05.11.008 The value 0 MUST indicate that no "Sequence Number" field is appended after the "Event / State Parameter" field.

CC:0071.03.05.11.009 The value 1 MUST indicate that a "Sequence Number" field is appended after the "Event / State Parameter" field.

Event / State Parameters Length (5 bits)

This field is used to advertise the length in bytes of the Event / State Parameters field.

CC:0071.03.05.11.00A The value 0 MUST indicate that no Event / State Parameter field is appended after the Event Parameters Length field.

CC:0071.03.05.11.00B Values in the range 1..31 MUST indicate the length of the Event / State Parameter field appended after the Event / State Parameters Length field.

Event / State Parameter (N bytes)

The Event / State Parameter field is used to specify associated parameters to a Notification.

Parameters for each Event/State Notification are defined in [36].

CC:0071.03.05.13.001 This field MAY carry an encapsulated command. In this case, the field MUST include the complete
CC:0071.03.05.11.00C command structure, i.e. Command Class, Command and all mandatory command fields.

CC:0071.03.05.11.00D If a node encapsulates a command in this field, the corresponding Command Class MUST be supported by the node and advertised in the Node Information Frame (NIF) or S0/S2 Supported Command Report.

CC:0071.03.05.11.00E This field MUST be omitted if the "Event Parameter Length" field is set to 0.

Section 2.2.74.6.1 provides an example of Event Parameter encapsulation.

Sequence Number (8 bits)

CC:0071.03.05.13.002 This field is used to advertise a sequence number for the actual Notification. This command MAY carry a Sequence Number field.

CC:0071.03.05.11.00F This field MUST be omitted if the Sequence flag is set to 0.

CC:0071.03.05.11.010 The first sequence number for each distinct Notification MUST be 1. A sending node MUST increment the sequence number for a Notification each time it issues a Notification Report for that given Notification.

CC:0071.03.05.11.011 The Sequence Number range MUST be in the range 0..255. The value after 255 MUST be 0.

Example:

1. Notification {Smoke Alarm::Smoke Detected}, ..., Seq. Number = 254
2. Notification {Smoke Alarm::Smoke Detected}, ..., Seq. Number = 255
3. Notification {Heat Alarm::Overheat Detected}, ..., Seq. Number = 6
4. Notification {Smoke Alarm::Smoke Detected}, ..., Seq. Number = 0

5. Etc.

2.2.74.6.1 Event / State parameter encapsulation

CC:0071.03.05.11.012 Event / State parameter encapsulation of commands MUST comprise the entire command, starting from the Command Class identifier.

For example, a "User Code Report" is used as Event / State Parameter in a "Keypad Lock/Unlock Operation". The "User Code Report" has the following format:

- Command Class =COMMAND_CLASS_USER_CODE = 0x63
- Command = USER_CODE_REPORT = 0x03
- User Identifier = 0x01
- User ID Status = 0x01
- User Code = 0x30, 0x30, 0x30, 0x30

The complete User Code Report therefore comprises the following Bytes: [0x63, 0x03, 0x01, 0x01, 0x30, 0x30, 0x30, 0x30].

Another example with the Node naming and location Command class is given in Table 2.435.

Table 2.435: Notification Report::Event / State parameter encapsulation (example)

Notification Report Command fields		Value	Explanation
1	COMMAND_CLASS_NOTIFICATION	0x71	Notification CC id
2	NOTIFICATION_REPORT	0x05	Notification Report command id
3	V1 Alarm Type	0x00	Not implemented
4	V1 Alarm Level	0x00	Not implemented
5	Reserved	0x00	Reserved field
6	Notification Status	0xFF	Unsolicited report is activated
7	Notification Type	0x01	Smoke Alarm
8	Notification Event	0x01	Smoke Detected
9	Sequence Number / Event Parameters Length	0x1A	Sequence Number is appended / Event Parm Length = 10
10	Event Parm 1	0x77	Node Naming & Location CC id
11	Event Parm 2	0x06	Node Location Report command id
12	Event Parm 3	0x00	Cmd Parm: Char = ASCII
13	Event Parm 4	0x4B	Cmd Parm: Node Location Char 1 = K
14	Event Parm 5	0x49	Cmd Parm: Node Location Char 2 = I
15	Event Parm 6	0x54	Cmd Parm: Node Location Char 3 = T
16	Event Parm 7	0x43	Cmd Parm: Node Location Char 4 = C
17	Event Parm 8	0x48	Cmd Parm: Node Location Char 5 = H
18	Event Parm 9	0x45	Cmd Parm: Node Location Char 6 = E
19	Event Parm 10	0x4E	Cmd Parm: Node Location Char 7 = N
20	Sequence Number	0x0F	Sequence Number = 15

2.2.74.7 Notification Supported Get Command

This command is used to request supported Notification Types.

CC:0071.03.07.11.001 The Notification Supported Report Command MUST be returned in response to this command.

CC:0071.03.07.11.002 This command MUST NOT be issued via multicast addressing.

CC:0071.03.07.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.436: Notification Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NOTIFICATION (0x71)							
Command = NOTIFICATION_SUPPORTED_GET (0x07)							

2.2.74.8 Notification Supported Report Command

This command is used to advertise supported Notification Types.

Table 2.437: Notification Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NOTIFICATION (0x71)							
Command = NOTIFICATION_SUPPORTED_REPORT (0x08)							
V1 Alarm	Reserved		Number of Bit Masks				
Bit Mask 1							
...							
Bit Mask N							

V1 Alarm (1 bit)

This field is used to indicate if the node implements proprietary Alarms from version 1.

CC:0071.03.08.11.001 The value 0 MUST indicate that the device implements only Notification CC V2 or newer Notification Types.

CC:0071.03.08.11.002 The value 1 MUST indicate that the device implements Notification CC V2 Notification Types as well as proprietary Alarm CC V1 Alarm Types and Alarm Levels.

Number of Bit Masks (5 bits)

CC:0071.03.08.11.003 This field MUST advertise the length in bytes of the Bit Mask field.

The value MUST be in the range 1..31.

Bit Mask (N bytes)

The Bit Mask field describes the supported Notification Types by the node. The length of this field CC:0071.03.08.11.004 in bytes MUST match the value advertised in the Number of Bit Masks field.

- Bit 0 in Bit Mask 1 is not allocated to any Notification Type and MUST be set to zero.
- Bit 1 in Bit Mask 1 indicates if Notification Type = Smoke Alarm (0x01) is supported.
- Bit 2 in Bit Mask 1 indicates if Notification Type = CO Alarm (0x02) is supported.
- Bit 3 in Bit Mask 1 indicates if Notification Type = CO2 Alarm (0x03) is supported
- ...

For Notification Types, refer to [36].

CC:0071.03.08.11.005 If the Notification Type is supported, the corresponding bit MUST be set to 1.

If the Notification Type is not supported, the corresponding bit MUST be set to 0.

CC:0071.03.08.11.006 The Notification Type values 0x00 and 0xFF are special-purpose values. Reserved values and special-purpose values MUST NOT be advertised in the Bit Mask field by a sending node and MUST be ignored by a receiving node.

2.2.74.9 Event Supported Get Command

This command is used to request the supported Notifications for a specified Notification Type.

CC:0071.03.01.11.001 The Event Supported Report Command MUST be returned in response to this command.

CC:0071.03.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:0071.03.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.438: Event Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NOTIFICATION (0x71)							
Command = EVENT_SUPPORTED_GET (0x01)							
Notification Type							

Notification Type (8 bits)

For Notifications Types, refer to [36].

CC:0071.03.01.11.004 If a node receives an unsupported Notification Type or a special-purpose value, the receiving node MUST respond with Event Supported Report Command with the Notification Type specified in the this command and the Number of Bit Masks field set to 0.

2.2.74.10 Event Supported Report Command

This command is used to advertise supported events/states for a specified Notification Type.

Table 2.439: Event Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NOTIFICATION (0x71)							
Command = EVENT_SUPPORTED_REPORT (0x02)							
Notification Type							
Reserved			Number of Bit Masks				
Bit Mask 1							
...							
Bit Mask N							

Notification Type (8 bits)

For Notifications Types, refer to [36].

Number of Bit Masks (5 bits)

This field is used to advertise the length (in bytes) of the Bit Mask field.

CC:0071.03.02.11.001 The value MUST be in the range 0..31.

The value 0 MUST indicate that the Notification Type is not supported.

Bit Mask (N bytes)

CC:0071.03.02.11.002

This field is used to advertise the supported Events/States for the actual advertised Notification Type. The length of this field (in bytes) MUST match the value advertised in the Number of Bit Masks field. This field MUST be omitted if the "Number of Bit Masks" field is set to 0.

CC:0071.03.02.11.003 The bit value '1' MUST indicate that the actual Event/State is supported.

The bit value '0' MUST indicate that the actual Event/State is not supported.

Example: Notification Type = Heat Alarm (0x04):

- Bit 0 in Bit Mask 1 field is not allocated to any event and must therefore be set to zero.
- Bit 1 in Bit Mask 1 field indicates support for Overheat detected (0x01).
- Bit 2 in Bit Mask 1 field indicates support for Overheat, unknown loc. (0x02).
- Bit 3 in Bit Mask 1 field indicates support for Rapid temp rise (0x03).
- …

For Notification Types and their Events/States, refer to [36].

CC:0071.03.02.11.004 The Event/State values 0x00 and 0xFE are special-purpose values. Reserved values and special-purpose values MUST NOT be advertised in the Bit Mask field by a sending node and MUST be ignored by a receiving node.

2.2.75 Prepayment Command Class, version 1

The Prepayment Command Class defines the Commands necessary to implement a Z-Wave encapsulation of Prepayment data and to distribute prepayment information between devices

2.2.75.1 Prepayment Balance Get Command

This command is used to request the balance of the Prepayment.

The Prepayment Balance Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.440: Prepayment Balance Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PREPAYMENT (0x3F)							
Command = PREPAYMENT_BALANCE_GET (0x01)							
Balance Type		Reserved					

Balance Type (2 bits)

The field specifies which balance type is requested in the response report. The available balance types may be requested using the Prepayment Supported Get Command.

Table 2.441: Prepayment Balance Get::Balance Type encoding

Value	Balance Type
0x00	Utility Balance
0x01	Monetary Balance

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.75.2 Prepayment Balance Report Command

This command is used to report the current balances.

The report includes the following main elements:

- Balance
- Debt
- Emergency Credit

The elements MAY be given in monetary values or in utility units depending on the Balance Type field.

Table 2.442: Prepayment Balance Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PREPAYMENT (0x3F)							
Command = PREPAYMENT_BALANCE_REPORT (0x02)							
Balance Type		Meter type					
Balance Precision			Scale				
Balance Value 1							
Balance Value 2							
Balance Value 3							
Balance Value 4							
Debt Precision			Reserved				
Debt 1							
Debt 2							
Debt 3							
Debt 4							
Emer. Credit Precision			Reserved				
Emer. Credit 1							
Emer. Credit 2							
Emer. Credit 3							
Emer. Credit 4							
Currency 1							
Currency 2							
Currency 3							
Debt Recovery Percentage							

Balance Type (2 bits)

The field specifies which type of balance is given in the report.

All available Balance Types may be found in the section Section 2.2.75.1 Prepayment Balance Get Command.

Meter Type (6 bits)

Meter Type specifies the type of metering device the command originates. This field MUST be encoded according to the Meter Types defined in [26].

Scale (5 bits)

The Scale used to indicate the scale (unit) of the balance, debt and emergency credit value in the report. Scale field only used for a report of type "Utility Balance", for other reports set the scale to 0x1F. The Scale parameter is of the variable type *Meter Scale*; refer to Section 2.2.62.13 Meter Table Current Data Report Command.

Currency (3 bytes)

This field advertises the currency code. Reports of the type "Monetary Balance" MUST advertise currency codes complying with ISO 4217. Other report types MUST advertise a currency code of "XXX".

Table 2.443: Prepayment Balance Report::Currency examples

Currency Code	Currency 1	Currency 2	Currency 3
Pound sterling (ISO 4217)	G	B	P
US Dollar (ISO 4217)	U	S	D
No Currency	X	X	X

Balance, Debt and Emergency Credit Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Balance, Debt and Emergency Credit (32 bits)

The Balance, Debt and Emergency Credit fields MUST be encoded as 32-bit signed integers. The first byte MUST carry most significant byte. Table 2.12 Signed field encoding (two's complement representation) shows signed decimal values together with their hexadecimal equivalents.

Debt Recovery Percentage (8 bits)

The Debt Recovery Percentage indicates the percentage of the payment that is for debt recovery. This can take the value from 0 – 50%, the value is always given with a precision of 0 decimal places. The value 0xFF indicates that this field is unspecified.

This field is only used for only used for a report of type "Monetary Balance". For other reports this field MUST be set to 0xFF (unspecified).

2.2.75.3 Prepayment Supported Get Command

The Prepayment Supported Get Command is used to request type of Balance Reports that are available in the device.

The Prepayment Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.444: Prepayment Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PREPAYMENT (0x3F)							
Command = PREPAYMENT_SUPPORTED_GET (0x03)							

2.2.75.4 Prepayment Supported Report Command

The Prepayment Supported Report Command reports the types of Balance Reports that are available in the device.

Table 2.445: Prepayment Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PREPAYMENT (0x3F)							
Command = PREPAYMENT_SUPPORTED_REPORT (0x04)							
Reserved				Bit Mask Balance Types Supported			

Bit Mask - Balance Types Supported (8 bits)

The Bit Mask - Balance Types Supported byte describes the type of Balance Reports that are available in the device.

Bit 0 in Bit Mask is used to indicate if the Balance Report Type "Utility Balance" is supported, 0 indicating not supported and 1 indicating supported. Bit 1 in the Bit Mask is used to indicate if the Balance Report Type "Monetary Balance" is available in the device, 0 indicating not supported and 1 indicating supported.

All available Balance Types may be found in the section Section 2.2.75.1 Prepayment Balance Get Command.

2.2.76 Prepayment Encapsulation Command Class, version 1 [NEVER CERTIFIED]

Warning: THIS COMMAND CLASS HAS NEVER BEEN CERTIFIED

This command class has never been implemented and certified by a Z-Wave product Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.

Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

The Prepayment Encapsulation Command Class is used to smartcard preinstalled security mechanisms.

2.2.76.1 Prepayment encapsulation command

This command is used to encapsulate Smart card related data communication (e.g. between a Card reader and Meter), allowing the smartcard preinstalled security mechanisms to be applied transparently to Z-Wave.

Table 2.446: Prepayment Encapsulation Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PREPAYMENT_ENCAPSULATION (0x41)							
Command = CMD_ENCAPSULATION (0x01)							
Data 1							
...							
Data N							

Data (N bytes)

This field contains prepayment smart card data.

2.2.77 Proprietary Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST NOT support this Command Class. Consult the Z-Wave Alliance when your application does not seem to fit any existing Command Class.

The Proprietary Command Class is used to transfer data between devices. The data content MUST be vendor specific and commands MUST NOT provide any value-add with respect to the Home Automation application in general.

2.2.77.1 Proprietary set command

This command is used to transfer data to a device.

Table 2.447: Proprietary Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROPRIETARY (0x88)							
Command = PROPRIETARY_SET (0x01)							
Data 1							
...							
Data N							

Data (N bytes)

The data fields may be used to set various data in the device. The number of data fields transmitted MUST be determined from the length field in the frame.

2.2.77.2 Proprietary get command

This command is used to request data from a device.

The Proprietary Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.448: Proprietary Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROPRIETARY (0x88)							
Command = PROPRIETARY_GET (0x02)							
Data 1							
...							
Data N							

Data (N bytes)

Refer to explanation under the Proprietary Set Command.

2.2.77.3 Proprietary report command

This command is used to retrieve various data from a device.

Table 2.449: Proprietary Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROPRIETARY (0x88)							
Command = PROPRIETARY_REPORT (0x03)							
Data 1							
...							
Data N							

Data (N bytes)

Refer to explanation under the Proprietary Set Command.

2.2.78 Protection Command Class, version 1

The Protection Command Class version 1 used to protect a device against unintentional control by e.g. a child.

2.2.78.1 Protection Set Command

This command is used to set the protection state in a device.

Table 2.450: Protection Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_SET (0x01)							
Protection State							

Protection State (8 bits)

The Protection State field used to set the protection state of the device.

Table 2.451: Protection Set::Protection State encoding

Protection State	Description
0x00	Unprotected - The device is not protected, and may be operated normally via the user interface.
0x01	Protection by sequence - The device is protected by altering the way the device normally is operated into a more complicated sequence of actions, e.g. if a device normally is controlled by a single press of a button on the device it might be changed to require 3 rapid presses on a button to control it.
0x02	No operation possible - It is not possible at all to control a device directly via the user interface.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Control via Z-Wave is always possible independently of the protection state.

2.2.78.2 Protection get command

This Command is used to request the protection state from a device.

The Protection Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.452: Protection Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_GET (0x02)							

2.2.78.3 Protection report command

This command is used to report the protection state of a device.

Table 2.453: Protection Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_REPORT (0x03)							
Protection State							

Protection State (8 bits)

Refer to explanation under Protection Get Command.

2.2.79 Protection Command Class, version 2

The Protection Command Class version 2 is extended to specify whether a device may be controlled via RF Commands or not. When a video recorder is powered by an outlet that can be controlled by RF the user would like to prevent the video recorder from being turned off when it is recording her/his favorite show. In this case the Protection Command Class version 2 may be used to protect the outlet from being turned off by setting the outlet in "No RF Control" state.

The following Commands have been added or changed in version 2. The Commands not mentioned remain unchanged.

This Command Class is intended for convenience applications. The Command Class SHOULD NOT be used for safety critical applications.

2.2.79.1 Protection set command

This command is used to set the protection state in a device.

Table 2.454: Protection Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_SET (0x01)							
Reserved				Local Protection State			
Reserved				RF Protection State			

Local Protection State (4 bits)

The Local Protection State field used to set the protection state of the device.

Table 2.455: Protection Set::Local Protection State encoding

Local Protection State	Description
0	Unprotected - The device is not protected, and may be operated normally via the user interface.
1	Protection by sequence - The device is protected by altering the way the device normally is operated into a more complicated sequence of actions, e.g. if a device normally is controlled by a single press of a button on the device it might be changed to require 3 rapid presses on a button to control it.
2	No operation possible - It is not possible at all to control a device directly via the user interface.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Note: Local Protection can only protect a device from "normal operation". This means only the operation that is intended by the application of the device. It is NOT allowed to protect the device from network functionalities. The device cannot be protected from being put into learn mode nor from sending out the NIF.

RF Protection State (4 bits)

The RF Protection State field used to set the RF protection state of the device. In the case where a device set into a RF Protection State which instructs the device not to answer to a "normal operation" Command, the device MUST return the Application Rejected Request Command (Status = 0) of the Application Status Command Class. Refer to the Application Status Command Class.

Table 2.456: Protection Set::RF Protection State

RF Protection State	Description
0	Unprotected - The device MUST accept and respond to all RF Commands.
1	No RF control - all runtime Commands are ignored by the device. The device MUST still respond with status on requests.
2	No RF response at all. The device will not even reply to status requests.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Note: It is only possible to un-protect the device with the Protection Set Command. It is not allowed ignore Protection Commands. If a device is excluded from the network, the protection states MUST be reset.

2.2.79.2 Protection report command

This command is used to report the protection state of a device.

Table 2.457: Protection Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_REPORT (0x03)							
Reserved				Local Protection State			
Reserved				RF Protection State			

For field description, refer to *Protection set command*.

2.2.79.3 Protection supported get command

This command is used to query supported protection capabilities.

The Protection Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.458: Protection Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_SUPPORTED_GET (0x04)							

2.2.79.4 Protection supported report command

This command is used to advertise supported protection capabilities.

Table 2.459: Protection Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_SUPPORTED_REPORT (0x05)							
Reserved						Exclusive Control	Timeout
Local Protection State Byte 1							
Local Protection State Byte 2							
RF Protection State Byte 1							
RF Protection State Byte 2							

Local Protection State Byte (2 bytes)

The list of all Local Protection States may be found in section Section 2.2.79.1. The two bytes MUST be interpreted as bit masks where byte 1 bit 0 represent Local Protection State 0, byte 1 bit 1 represent Protection State 1, byte 2 bit 0 represent Protection State 8 etc.

RF Protection State Byte (2 bytes)

The list of all RF Protection States may be found in section Section 2.2.79.1. The two bytes MUST be interpreted as bit masks where byte 1 bit 0 represent RF Protection State 0, byte 1 bit 1 represent Protection State 1, byte 2 bit 0 represent Protection State 8 etc.

Exclusive Control (1 bit)

When this bit is set to 1 the device support Exclusive Control. When Exclusive Control is supported the device MUST support the Commands Protection Exclusive Control Set, Get and Report described below.

Timeout (1 bit)

When this bit is set to 1 the device supports a timeout for RF Protection State. When the timeout is supported the device MUST support the Commands Protection Timeout Set, Get and Report described below.

2.2.79.5 Protection exclusive control

The Protection Exclusive Control is an optional feature. The Commands in this chapter can only be implemented if the device supporting Protection Command Class version 2 announces support for Exclusive Control in the Protection Supported Report Command.

2.2.79.5.1 Protection exclusive control set command

This command is used to set the NodeID of a Z-Wave device that can override the protection state in a protected device.

Table 2.460: Protection Exclusive Control Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_EC_SET (0x06)							
NodeID							

NodeID

The NodeID that has exclusive control can override the RF protection state of the device and can control it regardless of the protection state. Commands from any other nodes in the network may be restricted by the RF protection state. In that case, the Application Rejected Request Command MUST be returned.

All of the Protection Command Class commands will be accepted and processed regardless of whether or not a node has exclusive control.

Factory default setting of the NodeID for exclusive control MUST be set to 0. To reset the exclusive control state in a device an Exclusive Control Set Command with NodeID 0 as parameter MUST be send to the device.

2.2.79.5.2 Protection exclusive control get command

Table 2.461: Protection Exclusive Control Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_EC_GET (0x07)							

2.2.79.5.3 Protection exclusive control report command

This command is used to return the NodeID of a Z-Wave device that has exclusive control over this device in protection mode.

Table 2.462: Protection Exclusive Control Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_EC_REPORT (0x08)							
NodeID							

NodeID

See description under the Protection Exclusive Control Set Command section Section 2.2.79.5.1.

2.2.79.6 Protection timeout

The Protection Timeout is an optional feature. The Commands in this section MAY be implemented if the device supporting Protection Command Class version 2 announces support for Timeout in the Protection Supported Report Command.

2.2.79.6.1 Protection timeout set command

This command is used to set the timeout for protection mode in a device.

Table 2.463: Protection Timeout Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_TIMEOUT_SET (0x09)							
Timeout							

Timeout

The timeout describes the time that a device MUST remain in RF Protection mode.

Factory default setting for the Timeout parameter MUST be 0x00. This field MUST be encoded according to Table 2.464.

Table 2.464: Protection Timeout Set::Timeout encoding

RF Protection State	Description
0x00	No timer is set. All “normal operation” Commands MUST be accepted.
0x01-0x3C	Timeout is set from 1 second (0x01) to 60 seconds (0x3C) in 1-second resolution.
0x41-0xFE	Timeout is set from 2 minutes (0x41) to 191 minutes (0xFE) in 1-minute resolution.
0xFF	No Timeout – The Device will remain in RF Protection mode infinitely.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.79.6.2 Protection timeout get command

This command is used to request a Protection Timeout Report Command from the device.

The Protection Timeout Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.465: Protection Timeout get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_TIMEOUT_GET (0x0A)							

2.2.79.6.3 Protection timeout report command

This command is used to return the remaining time that a device will remain in protection mode.

Table 2.466: Protection Timeout Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_PROTECTION (0x75)							
Command = PROTECTION_TIMEOUT_REPORT (0x0B)							
Timeout							

Timeout

This field indicates the remaining timeout set in the Node. It MUST be encoded as described in Table 2.464

2.2.80 Pulse Meter Command Class, version 1 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Meter Command Class. If implementing this command class, it is RECOMMENDED that the Meter Command Class is also implemented.

The Pulse Meter Command Class defines the Commands necessary to implement the pulse meter functionality. The Pulse Meter Command Class is intended for all kinds of meters that generate pulses, such as gas and water meters.

2.2.80.1 Pulse meter get command

This command is used to request the number of pulses that has been counted.

The Pulse Meter Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.467: Pulse Meter Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_PULSE (0x35)							
Command = METER_PULSE_GET (0x04)							

2.2.80.2 Pulse meter report command

This command is used to report the number of pulses detected.

Table 2.468: Pulse Meter Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_PULSE (0x35)							
Command = METER_PULSE_REPORT (0x05)							
Pulse Count 1							
Pulse Count 2							
Pulse Count 3							
Pulse Count 4							

Pulse Count (32 bits)

The Pulse Count field contains the number of pulses generated by the meter. The first byte is the most significant byte.

2.2.81 Rate Table Configuration Command Class, version 1

The Rate Table Configuration Command Class defines the parameter sets for a range of rates.

The Rate Table configuration commands are separated for the Rate Table monitoring commands in the rate Table Monitor Command Class, allowing the classes to be optionally supported at different Z-Wave security levels.

Every rate is described as a combination of time, maximum consumption, maximum demand and DCP Rate ID.

- Time: Time of day. E.g. 7.15am to 9.20am
- Maximum Consumption: E.g. 200kWh. This allows the Utility Supplier to provide special rates for 'utility conserving' end users.
- Maximum Demand: E.g. 4000W. This allows the Utility Supplier to provide special rates for end-users which manages to keep the 'peak' demand under 'control'.
- DCP Rate ID: E.g. DCP Rate ID = 16. This allows the Utility Supplier to provide special rates for end-users participating in DCP event with a specific DCP Rate ID.

2.2.81.1 Rate table set command

This command adds a *rate parameter set to a given rate parameter set identifier*.

Table 2.469: Rate Table Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_CONFIG (0x48)							
Command = RATE_TBL_SET (0x01)							
Rate Parameter Set ID							
Reserved	Rate Type		Number of Rate Char.				
Rate Character 1							
...							
Rate Character N							
Start Hour Local Time							
Start Minute Local Time							
Duration Minute 1							
Duration Minute 2							

The following part of the command is OPTIONAL depending on the parameters supported. Use the Rate Table Supported Get Command to obtain supported parameters beside the default above. A not supported parameter is not included into the command layout.

Table 2.470: Rate Table Set Command Optional

7	6	5	4	3	2	1	0
Consumption Precision 1			Consumption Scale 1				
Min. Consumption Value 1							
Min. Consumption Value 2							
Min. Consumption Value 3							
Min. Consumption Value 4							
Max. Consumption Value 1							
Max. Consumption Value 2							
Max. Consumption Value 3							
Max. Consumption Value 4							
Max. Demand Precision 1			Max. Demand Scale 1				
Max. Demand Value 1							
Max. Demand Value 2							
Max. Demand Value 3							
Max. Demand Value 4							
DCP Rate ID							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID indicates the requested parameter set.

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. This field MUST be encoded according to Table 2.345.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Rate Characters (5 bits)

Number of characters defining the rate text (1..32).

Rate Character (N bytes)

The Rate character fields hold the string identifying the rate parameter set. The character presentation uses standard ASCII codes (values in the range 128..255 MUST be ignored).

Start Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00..23) in local time.

Start Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00..59) in local time.

Duration Minute (16 bits)

Specify the duration in minutes (1..1440) since the start in local time. The value 1440 indicates that the parameter set applies the entire day.

Consumption Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Consumption Scale (5 bits)

The Consumption Scale used to indicate the scale (unit) is applicable for the rate. The Consumption Scale parameter is of the variable type Meter Scale; refer to Section 2.2.62.13 Meter Table Current Data Report Command.

Minimum / Maximum Consumption Value (4 * 8 bits)

The Minimum / Maximum Consumption Value are a 32 bit unsigned field. The first byte is the most significant byte.

The Minimum / Maximum Consumption Value are used in the Rate Table when Block Tariffs are used. Block tariffs assign blocks of energy at a set cost. The billing period is defined by the Utility Supplier.

Example of Block Tariff based Rate Table: (Two Block tariff system)

The first block from 0 kWh to 200 kWh is charged at 2 USD/kWh and all other units consumed over 200kWh will be charged at 2.5 USD/kWh

- Rate1: Min Consumption value = 0kWh, Max Consumption Value = 200kWh
- Rate2: Min Consumption value = 200kWh, Max Consumption Value = 2147483648kWh

NOTE: The actual charge is set using the Tariff Table Configuration Command Class, version 1.

NOTICE: The device receiving the Rate Table Report MUST show the value even though the Scale is not supported.

Max. Demand Precision (3 bits)

The Maximum Demand Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Max. Demand Scale (5 bits)

The Maximum Demand Scale field describes what unit is applicable for the rate. The Maximum Demand Scale parameter is of the variable type Meter Scale; refer to Section 2.2.62.13 Meter Table Current Data Report Command.

Max. Demand Value (4 * 8 bits)

The Maximum Demand Value is a 32 bit unsigned field. The first byte is the most significant byte.

The maximum demand value 0xFFFFFFFF is reserved and represents an unlimited maximum demand value.

NOTICE: The device receiving the Rate Table Report MUST show the value even though the Scale is not supported.

DCP Rate ID (8 bits)

The DCP Rate ID addresses the Demand Control Plan parameters, which will overrule other parameter sets defined in the Rate Table Command Class. A DCP Rate ID equal to zero disables Demand Control Plan mapping.

2.2.81.2 Rate table remove command

This command is used to remove rate parameter set(s).

Table 2.471: Rate Table Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_CONFIG (0x48)							
Command = RATE_TBL_REMOVE (0x02)							
Reserved		Rate Parameter Set IDs					
Rate Parameter Set ID 1							
...							
Rate Parameter Set ID N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Parameter Set IDs (6 bits)

The Rate Parameter Set IDs indicates the number of rate parameter set IDs in the command.

Rate Parameter Set ID (N bytes)

These fields contain a list of Rate Parameter Set ID's that should be removed from the Rate Table. All Rate Parameter Set ID's are cleared in case no Rate Parameter Set ID's are supplied.

2.2.82 Rate Table Monitor Command Class, version 1

The Rate Table Monitor Command Class defines the parameter sets for a range of rates.

2.2.82.1 Rate table supported get command

This command is used to request the number of rates and parameter sets supported by the Rate Table Command Class.

The Rate Table Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.472: Rate Table Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_SUPPORTED_GET (0x01)							

2.2.82.2 Rate table supported report command

This command is used to advertise the number of rates and parameter sets supported.

Table 2.473: Rate Table Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_SUPPORTED_REPORT (0x02)							
Rates Supported							
Parameter Set Supported Bit Mask 1							
Parameter Set Supported Bit Mask 2							

Rates Supported (8 bits)

Number of rates supported (1..255).

Parameter Set Supported Bit Mask 1, 2 (16 bits)

The Bit Mask field describes the supported parameter set in addition to the default-supported parameter set. The default parameter set comprises of Rate Parameter Set ID, Rate text, Start time and Duration.

It is possible to extend the default parameter set as follows:

Table 2.474: Rate Table Supported Report::Parameter Set Supported Bit Mask encoding

Parameter Set Supported	Bit Map	Description
1	Bit 0	Reserved
1	Bit 1	Supports *Block tariffs* if the bit is 1 and the opposite if 0. Block tariffs assign blocks of energy at a set cost, for example in a two block tariff the first block say from 0 kWh to 200 kWh is charged at X currency per unit(kWh) all other units consumed over 200kWh will be charged at Y currency per unit for the billing period.
1	Bit 2	Supports *Maximum demand tariffs* if the bit is 1 and the opposite if 0. Maximum demand tariffs are based on the maximum load that is measured for example 20kw over an averaging period. The charge is based on the max load; hence a 30kW maximum demand would be more costly than a 20kW maximum demand.
1	Bit 3	Supports *Subscribed demand tariffs* if the bit is 1 and the opposite if 0. This type of tariff is used in France and Italy primarily; the standing charge is calculated from the maximum load, for example 10A = 10 currency/month, 20A = 20 currency / month.
1	Bit 4	Supports Demand Control Plan mapping (DCP ID) if the bit is 1 and the opposite if 0.

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

2.2.82.3 Rate table get command

This command is used to request the rate parameter set for a given rate parameter set identifier.

The Rate Table Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.475: Rate Table Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_GET (0x03)							
Rate Parameter Set ID							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID addresses the wanted parameter set. The rate parameter set identifier MUST be a sequence starting from 1 to Rates Supported.

2.2.82.4 Rate table report command

This command reports rate parameter set for a given rate parameter set identifier.

Table 2.476: Rate Table Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_REPORT (0x04)							
Rate Parameter Set ID							
Reserved	Rate Type		Number of Rate Char.				
Rate Character 1							
...							
Rate Character N							
Start Hour Local Time							
Start Minute Local Time							
Duration Minute 1							
Duration Minute 2							

The following part of the command is OPTIONAL depending on the parameters supported. Use the Rate Table Supported Get Command to obtain supported parameters beside the default above. A not supported parameter is removed from the command layout.

Table 2.477: Rate Table Report Command Optional

7	6	5	4	3	2	1	0
Consumption Precision 1			Consumption Scale 1				
Min. Consumption Value 1							
Min. Consumption Value 2							
Min. Consumption Value 3							
Min. Consumption Value 4							
Max. Consumption Value 1							
Max. Consumption Value 2							
Max. Consumption Value 3							
Max. Consumption Value 4							
Max. Demand Precision 1			Max. Demand Scale 1				
Max. Demand Value 1							
Max. Demand Value 2							
Max. Demand Value 3							
Max. Demand Value 4							
DCP Rate ID							

Refer to description of fields under the Rate Table Set Command (Section 2.2.81.1).

2.2.82.5 Rate table active rate get command

This command is used to retrieve the rate currently active in the meter.

The Rate Table Active Rate Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.478: Rate Table Activate Rate Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_ACTIVE_RATE_GET (0x05)							

2.2.82.6 Rate table active rate report command

This command is used to advertise the rate parameter set ID of the rate currently active in the meter.

Table 2.479: Rate Table Activate Rate Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_ACTIVE_RATE_REPORT (0x06)							
Rate Parameter Set ID							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID of the rate currently active in the meter.

2.2.82.7 Rate table current data get command

This command is used to request a number of time stamped values (current) in physical units according to the dataset mask.

The Rate Table Current Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.480: Rate Table Current Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_CURRENT_DATA_GET (0x07)							
Rate Parameter Set ID							
Dataset Requested 1							
Dataset Requested 2							
Dataset Requested 3							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID addresses the wanted parameter set. The rate parameter set identifier MUST be a sequence starting from 1 to Rates Supported.

Dataset Requested (24 bits)

The dataset requested parameter indicates which data is requested by the command. This field MUST be encoded according to the Meter Datasets defined in [26].

2.2.82.8 Rate table current data report command

This command is used to report a number of time stamped values (current) in physical units in the device.

Table 2.481: Rate Table Current Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_CURRENT_DATA_REPORT (0x08)							
Reports to Follow							
Rate Parameter Set ID							
Dataset 1							
Dataset 2							
Dataset 3							
Year 1							
Year 2							
Month							
Day							
Hour Local Time							
Minute local Time							
Second Local Time							
Current Precision 1				Current Scale 1			
Current Value 1,1							
Current Value 1,2							
Current Value 1,3							
Current Value 1,4							
...							
Current Precision N				Current Scale N			
Current Value N,1							
Current Value N,2							
Current Value N,3							
Current Value N,4							

Reports to Follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID addresses the wanted parameter set. The rate parameter set identifier MUST be a sequence starting from 1 to Rates Supported.

Dataset (24 bits)

The dataset parameter indicates which data is included in the report. This field MUST be encoded according to the Meter Dataset defined in [26].

Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that a accumulated value is not determined yet.

Day (8 bits)

Specify the day of the month between 01 and 31.

Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00..23) in local time.

Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00..59) in local time.

Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00..59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Precision (N * 3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Current Scale (N*5 bits)

The Current Scale is used to indicate the scale (unit) of the following value. This field MUST be encoded according to the Meter Dataset and Scales defined in [26].

Current Value (N * 32 bits)

The Current Value advertises a value corresponding to the Dataset Requested field of the Get Command. The field MUST be encoded as a 32-bit signed integer. The first byte (Value 1) MUST carry most significant byte. *Signed encoding* shows signed decimal values together with their hexadecimal equivalents.

NOTICE: The device receiving the Rate Table Current Data Report MUST show the value even though the Scale is not supported.

2.2.82.9 Rate table historical data get command

This command is used to request a number of time stamped values (historical) in physical units according to rate type, dataset mask and time interval.

The Rate Table Historical Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 2.482: Rate Table Historical Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_HISTORICAL_DATA_GET (0x09)							
Maximum Reports							
Rate Parameter Set ID							
Dataset Requested 1							
Dataset Requested 2							
Dataset Requested 3							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Start Second Local Time							
Stop Year 1							
Stop Year 2							
Stop Month							
Stop Day							
Stop Hour Local Time							
Stop Minute Local Time							
Stop Second Local Time							

Maximum Reports (8 bits)

The maximum reports parameter is used to indicate the maximum number of reports to return based on the get. Reports are always returned with the most recently recorded value first. If set to 0x00 the meter will return all reports based on the request.

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID addresses the wanted parameter set. The rate parameter set identifier MUST be a sequence starting from 1 to Rates Supported.

Dataset Requested (24 bits)

The dataset requested parameters indicate data requested. This field MUST be encoded according to the Meter Dataset defined in [26].

Start/Stop Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start/Stop Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that an accumulated value is not determined yet.

Start/Stop Day (8 bits)

Specify the day of the month between 01 and 31.

Start/Stop Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00..23) in local time.

Start/Stop Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00..59) in local time.

Start/Stop Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00..59) in local time. The value 60 used to keep UTC from wandering away is not supported.

2.2.82.10 Rate table historical data report command

This command is used to report a number of time stamped values (historical) in physical units in the device.

Table 2.483: Rate Table Historical Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_RATE_TBL_MONITOR (0x49)							
Command = RATE_TBL_HISTORICAL_DATA_REPORT (0x0A)							
Reports to Follow							
Rate Parameter Set ID							
Dataset 1							
Dataset 2							
Dataset 3							
Year 1							
Year 2							
Month							
Day							
Hour Local Time							
Minute local Time							
Second Local Time							
Historical Precision 1				Historical Scale 1			
Historical Value 1,1							
Historical Value 1,2							
Historical Value 1,3							
Historical Value 1,4							
...							
Historical Precision N				Historical Scale N			
Historical Value N,1							
Historical Value N,2							
Historical Value N,3							
Historical Value N,4							

Reports to follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID addresses the wanted parameter set. The rate parameter set identifier MUST be a sequence starting from 1 to Rates Supported.

Dataset (24 bits)

The dataset parameter indicates which data is included in the report. This field MUST be encoded according to the Meter Dataset defined in [26].

Year 1, 2 (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that a accumulated value is not determined yet.

Day (8 bits)

Specify the day of the month between 01 and 31.

Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00..23) in local time.

Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00..59) in local time.

Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00..59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Historical Precision (N * 3 bits)

The Historical Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Historical Scale (N * 5 bits)

The Historical Scale used to indicate the scale (unit) of the following value. The Historical Scale parameter is of the variable type Meter Scale; refer to Section 2.2.62.13.

Historical Value (N * 32 bits)

The Historical Value is a 32 bit signed field defined by dataset requested field. The first byte (Value 1) is the most significant byte. shows signed decimal values together with their hexadecimal equivalents.

NOTICE: The device receiving the Rate Table Historical Data Report MUST show the value even though the Scale is not supported.

2.2.83 Scene Activation Command Class, version 1

The Scene Activation Command Class used for launching scenes in a number of actuator nodes e.g. another scene-controlling unit, a multilevel switch, a binary switch etc.

2.2.83.1 Compatibility Considerations

A node supporting this Command Class MUST also support the Scene Actuator Configuration Command Class.

This command class requires an initial configuration of the scenes to be launched by the Scene Actuator Configuration Set.

Since a common identifier that is sent out, multicast addressing may be used. Multicast addressing may eliminate the potential popping effect which could be the result if individual Set Commands were send out to a large number of nodes distributed over a vast area. The multicast transmission MUST be followed by individual singlecast transmissions to ensure all the nodes have received the command.

2.2.83.2 Scene Activation Set Command

This command is used to activate the setting associated to the scene ID.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCENE_ACTIVATION (0x2B)							
Command = SCENE_ACTIVATION_SET (0x01)							
Scene ID							
Dimming Duration							

Scene ID (8 bits)

This field is used to specify the Scene ID that the receiving node MUST activate

This field MUST be in the range 1..255.

Dimming Duration (8 bits)

This field MUST specify the time that the transition from the current level to the target level (indicated by the Scene ID) should take.

Supporting actuator nodes without duration capabilities MUST ignore this field.

Supporting actuator nodes with a duration capabilities SHOULD respect the indicated dimming duration to reach the target level.

This field MUST be encoded according to Table 2.484.

Table 2.484: Scene Activation Set::Dimming Duration encoding

Value	Description
0x00	Instantly
0x01..0x7F	Obtain dimming durations from 1 second (0x01) to 127 seconds (0x7F) in 1-second resolution
0x80..0xFE	Specify dimming durations from 1 minute (0x80) to 127 minutes (0xFE) in 1-minute resolution.
0xFF	Specify dimming duration configured by the Scene Actuator Configuration Set and Scene Controller Configuration Set Command depending on device used.

2.2.84 Scene Actuator Configuration Command Class, version 1

The Scene Actuator Configuration Command Class is used to configure scenes settings for a node supporting an actuator Command Class, e.g. a multilevel switch, binary switch etc.

2.2.84.1 Compatibility Considerations

A node supporting this Command Class MUST support 255 Scene IDs (Scene ID 1 to 255). The Scene ID 0 MUST NOT be supported..

2.2.84.2 Scene Actuator Configuration Set Command

This command is used to associate the specified scene ID to the defined actuator settings.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCENE_ACTUATOR_CONF (0x2C)							
Command = SCENE_ACTUATOR_CONF_SET (0x01)							
Scene ID							
Dimming Duration							
Override	Reserved						
Level							

Scene ID (8 bits)

This field is used to specify the Scene ID to be associated with the current actuator settings.

This field MUST be in the range 1..255.

Dimming Duration (8 bits)

This field MUST specify the time it MUST take to reach the target level associated to the actual Scene ID.

A receiving node always starts from current level. The dimming duration MUST be the same independently of the number of levels to be changed.

Supporting actuator nodes without duration capabilities MUST ignore this field.

Supporting actuator nodes with duration capabilities SHOULD respect the indicated duration to reach the target level.

This field MUST be encoded according to Table 2.9.

Override (1 bit)

This field is used to specify if the current actuator settings MUST be used as settings for the actual Scene ID.

If this field is set to 0, the current actuator settings at the supporting node MUST be associated to the actual Scene ID. In this case, the Level field MUST be ignored.

If this field is set to 1, the value indicated by the Level field MUST be associated to the actual Scene ID.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Level (8 bits)

This field is used to specify the actuator setting that MUST be used as setting for the actual Scene ID. This field MUST correspond and be interpreted as the Value field of the Basic Set Command. i.e. the supporting node receiving a Scene Activation Set for the actual Scene ID specified in this command MUST react as if it had received a Basic Set Command with the Value field set to the value indicated by this field.

2.2.84.3 Scene Actuator Configuration Get Command

This command is used to request the settings for a given scene identifier or for the scene currently active.

The Scene Actuator Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCENE_ACTUATOR_CONF (0x2C)							
Command = SCENE_ACTUATOR_CONF_GET (0x02)							
Scene ID							

Scene ID (8 bits)

This field is used to specify the requested Scene ID settings.

Values in the range 1..255 MUST indicate that the receiving node MUST return settings associated to the actual Scene ID..

The value 0 MUST indicate that the current active scene (if any).

2.2.84.4 Scene Actuator Configuration Report Command

This command is used to advertise the settings associated to a scene identifier.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCENE_ACTUATOR_CONF (0x2C)							
Command = SCENE_ACTUATOR_CONF_REPORT (0x03)							
Scene ID							
Level							
Dimming Duration							

Scene ID (8 bits)

This field is used to specify the actual Scene ID for which the settings are being advertised.

Values in the range 1..255 MUST indicate an actual Scene ID.

The value 0 MUST indicate that no scene is currently active at the sending node and no scene setting is advertised in this command. In this case, the Level and Dimming Duration fields SHOULD be ignored.

Level (8 bits)

This field is used to advertise the actuator setting that MUST be used by the node when activating the actual Scene ID.

The value advertised by the sending node MUST correspond to the format of the Value field when returning a Basic Report Command.

Dimming Duration (8 bits)

This field MUST specify the time it MUST take to reach the target level associated to the actual Scene ID..

Supporting actuator nodes without duration capabilities MUST ignore this field and SHOULD set it to 0x00.

Supporting actuator nodes with duration capabilities SHOULD respect the indicated duration to reach the target level.

This field MUST be encoded according to Table 2.485.

Table 2.485: Scene Actuator Configuration Report::Dimming Duration encoding

Dimming Duration	Description
0x00	Instantly
0x01..0x7F	Dimming durations from 1 second (0x01) to 127 seconds (0x7F) in 1-second resolution.
0x80-0xFE	Specify dimming durations from 1 minute (0x80) to 127 minutes (0xFE) in 1-minute resolution.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.85 Scene Controller Configuration Command Class, version 1

The Scene Controller Configuration Command Class is used to configure nodes launching scenes using their association groups

2.2.85.1 Compatibility Considerations

A node supporting this Command Class MUST support 255 scene IDs (Scene ID 1 to 255). The Scene ID 0 MUST NOT be supported..

A node supporting this Command Class MUST support the Association Command Class, version 1 or newer.

A node supporting this Command Class MUST issue Scene Activation Set Command via its association groups.

2.2.85.2 Scene Controller Configuration Set Command

This command is used to configure settings for a given physical item on the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCENE_CONTROLLER_CONF (0x2D)							
Command = SCENE_CONTROLLER_CONF_SET (0x01)							
Group ID							
Scene ID							
Dimming Duration							

Group ID (8 bits)

This field is used to specify an Association Group Identifier.

The grouping identifier is mapped into a physical item e.g. a push button on the device in question. The grouping identifier values MUST be a sequence starting from 1. The Association Supported Groupings Get Command may be used to request the number of groupings that a node supports.

A sending node SHOULD NOT specify Group ID 1 as it is reserved for the Lifeline association group.

Scene ID (8 bits)

This field is used to specify the Scene ID that MUST be associated with the actual grouping identifier.

Values in the range 1..255 MUST indicate that the receiving node MUST associate the Scene ID and Dimming Duration values to the actual Association Group ID. The supporting node MUST set the Scene ID value indicated by this field in the Scene Activation Set Command when issuing it via the actual Group ID.

The value 0 MUST indicate that the receiving node MUST disable an associated scene for the specified Group ID.

Dimming Duration (8 bits)

This field is used to specify what value the supporting node MUST set in the Dimming Duration field of the Scene Activation Set Command when issuing it via the actual Group ID.

2.2.85.3 Scene Controller Configuration Get Command

This command is used to request the settings for a given association grouping identifier or the active settings.

The Scene Controller Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCENE_CONTROLLER_CONF (0x2D)							
Command = SCENE_CONTROLLER_CONF_GET (0x02)							
Group ID							

Group ID (8 bits)

This field is used to request scene settings configured for an actual Association Group Identifier.

Values in the range 1..255 MUST indicate the requested Group Identifier.

The value 0 MUST indicate that the receiving node MUST return the currently active Group Identifier and Scene ID (last activated group/scene).

A node receiving a non-supported group or having no currently active Group Identifier SHOULD return a report for Group ID 0.

2.2.85.4 Scene Controller Configuration Report Command

This command is used to advertise the current scene controller settings.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCENE_CONTROLLER_CONF (0x2D)							
Command = SCENE_CONTROLLER_CONF_REPORT (0x03)							
Group ID							
Scene ID							
Dimming Duration							

Group ID (8 bits)

This field is used to advertise the actual Association Group Identifier for which the settings are being advertised.

Scene ID (8 bits)

This field is used to specify the Scene ID that is associated with the actual grouping identifier.

Value in the range 1..255 MUST indicate the actual Scene ID setting for the specified grouping identifier.

The value 0 MUST indicate that the Group ID/Scene ID is disabled.

Dimming Duration (8 bits)

This field is used to specify the Dimming Duration that is associated with the actual grouping identifier.

If the actual Group ID/Scene ID is disabled, a sending node SHOULD set this field to 0.

2.2.86 Schedule Command Class, version 1

The Schedule Command Class allows scheduling the execution of commands for a given duration in a supporting device. It is a generic Command Class that may be used to schedule commands of any other Command Class.

2.2.86.1 Terminology

A **schedule** or **regular schedule** is a delayed execution of one or more commands for a given duration. The commands used in a schedule are typically "Set" type commands, affecting actuating resources or states.

A schedule is first **created** or **set**, meaning that an available **Schedule ID** is used and has been assigned a set of commands, a starting time and a duration.

A schedule can be **removed**, meaning that a previously set Schedule ID is freed and the assigned commands, starting time and duration are erased.

A schedule is **active** during the configured duration after its starting time.

When a schedule is active, the states affected by the schedule commands are temporarily changed.

When a schedule becomes inactive, the states affected by the schedule commands are restored to their previous values.

Several schedules can overlap and be active simultaneously.

Supporting nodes can optionally support **enabling** and **disabling** schedules. By default, a schedule is enabled when being created. When a schedule is disabled, it will never become active, even if the start time condition is met. It allows a controlling node to quickly enable/disable schedules without erasing the schedules configuration from the supporting nodes.

When all schedules are inactive, the supporting node is said to be operating in **Normal Mode**. Regardless of whether any schedule is active, Normal Mode's states are permanently affected by **direct commands** (issued without Schedule encapsulation). State changes triggered by schedules do not affect Normal Mode.

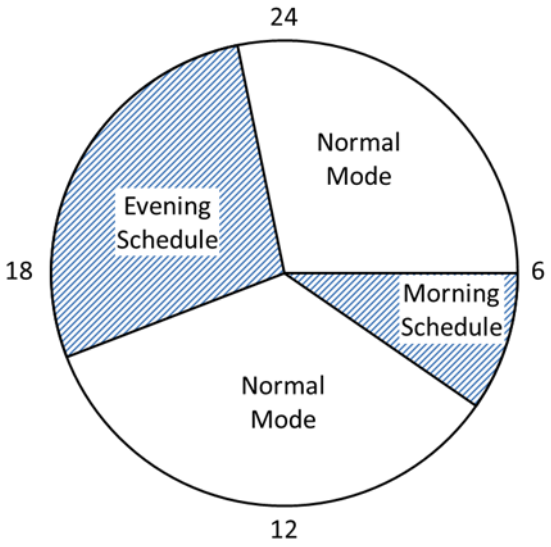


Figure 2.24: Simple daily schedules (example)

Direct commands affecting the states of a currently active schedule will cause those states to go back to normal mode and the corresponding normal mode states/values to be permanently changed.

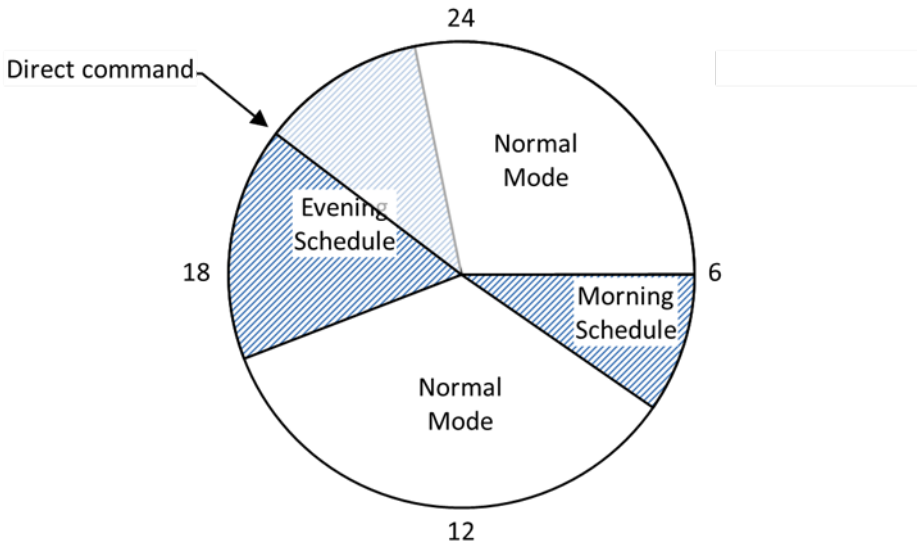


Figure 2.25: Receiving a direct command during daily schedules (example)

There are 2 types of special schedules.

If a **Fall Back Schedule** is created, it takes over the role of the Normal Mode. The Fall Back Schedule is activated when all other schedules are inactive. Fall Back Schedule is used to define default settings or states for the normal mode

If an **Override Schedule** is created, it **suspends** all other schedules. An override schedule may have a start time or start immediately. An override schedule may have a duration, run until stopped or run until another regular schedule starts. An example is given in the figure below

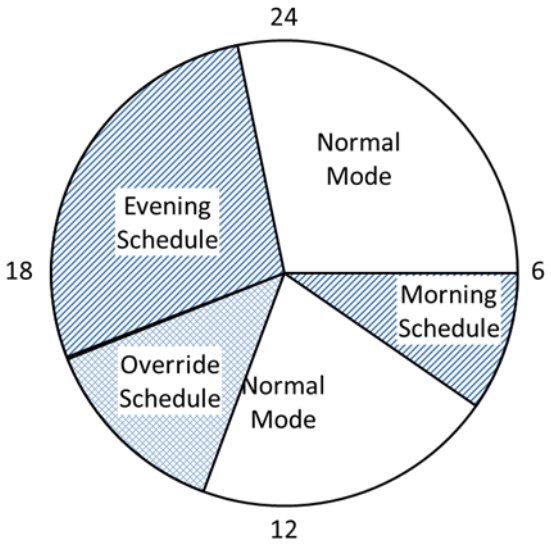


Figure 2.26: Daily schedules and an “Advance” Override Schedule (example)

An overview of the Schedule types and priorities is given in Table 2.486.

Table 2.486: Schedule CC terminology and priority

Priority	Term	Description
(highest)	Direct command	Affects device state immediately. Permanently affects Normal Mode.
(higher)	Override Schedule (Schedule ID = 0xFF)	All other schedules are suspended as long as the override schedule is active.
(normal)	Regular Schedule(s) (Schedule ID in [1..Supported number of schedules])	One or more schedules defining intended behavior.
(lower)	Fall Back Schedule (Schedule ID = 0xFE)	Fall Back Schedule becomes active when no other schedule is currently active. If the Fall Back Schedule is defined, Normal Mode is never reached, unless receiving direct commands.
(lowest)	Normal Mode	No schedules are currently active.

2.2.86.2 Interoperability Considerations

CC:0053.01.00.31.001 A schedule causes a temporary state change that only applies to the duration of the schedule. The state value(s) of the Normal Mode MUST NOT be affected by any command executed as part of a schedule.

CC:0053.01.00.31.002 The state value(s) of the Normal Mode MUST be restored when the affecting Schedule become inactive.

CC:0053.01.00.32.001 Schedules are set with a start time, which requires the supporting node to be aware of the current date and/or time. It is RECOMMENDED to support the Clock Command Class or the Time Command Class in order to allow a controlling node to verify or configure correct date/time settings at the supporting node.

2.2.86.2.1 Override Schedule

CC:0053.01.00.31.003 When the Override Schedule is active, all regular schedules MUST be suspended.

CC:0053.01.00.31.004 When the Override Schedule ends, all suspended schedules MUST resume the state they would have had if the Override Schedule had not been activated.

CC:0053.01.00.32.002 If the Override Schedule is created with a duration type not set to Override, a receiving node SHOULD respect the specified duration for the Override Schedule.

CC:0053.01.00.31.005 A supporting node MUST respect the specified duration for all regular schedules.

2.2.86.2.2 Fall Back Schedule

CC:0053.01.00.31.006 If no Fall Back Schedule is set, a supporting node MUST return to Normal Mode when no schedule is currently active.

CC:0053.01.00.31.007 If a Fall Back Schedule is set, a supporting node MUST:

- Activate the Fall Back Schedule when no other schedule is active.
- De-activate the Fall Back Schedule if any other schedule becomes active.
- Direct commands state changes MUST be active until the start of a new schedule affecting the same state values.

2.2.86.2.3 Multi Channel considerations

A Multi Channel device may support different schedule types and command classes for each Multi Channel End Point. An example is a central heating boiler with a thermostat for room heating and another thermostat for water heating. Each system may implement regular weekday+time schedules as well as an override schedule for manual intervention.

CC:0053.01.00.32.003 In any case, it is RECOMMENDED that multi-function devices support the Multi Channel Command Class, so that each End Point supports its own scheduling functionality.

2.2.86.3 Schedule Supported Get Command

This command is used to query the schedule functionalities supported by a node.

CC:0053.01.01.11.001 The Schedule Supported Report Command MUST be returned in response to this command.

CC:0053.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:0053.01.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_GET (0x01)							

2.2.86.4 Schedule Supported Report Command

This command is used to advertise the schedule functionalities supported by a node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_REPORT (0x02)							
Number of Supported Schedule IDs							
Support Enable/Disable	Fallback Support	Start Time Support					
Number of Supported CC							
Supported CC 1							
Reserved							Supported Command 1
...							
Supported CC N							
Reserved							Supported Command N
Override Support	Supported Override Types						

Number of Supported Schedule IDs (8 bits)

CC:0053.01.02.11.001 This field MUST advertise the number of regular Schedule IDs supported by the node.

CC:0053.01.02.11.002 The implemented Schedule IDs MUST be in the range 1..[Number of Supported Schedule IDs]. i.e. a node supporting 10 regular schedules MUST accept Schedule ID values in the range 1..10.

CC:0053.01.02.13.001 The following special Schedule IDs MAY also be supported by a device:

- Schedule ID = 0xFF (Override Schedule)
- Schedule ID = 0xFE (Fall Back schedule)

CC:0053.01.02.11.003 Special Schedule IDs MUST NOT be included in the number advertised in this field.

Start Time Support (6 bits)

This field is used to advertise the start time options supported for regular schedules by a node.

CC:0053.01.02.11.004 Regular schedules MUST support the advertised start time options.

CC:0053.01.02.12.001 The Override Schedule SHOULD support the advertised start time options.

CC:0053.01.02.11.005 This field MUST be treated as a bitmask and MUST be encoded according to Table 2.487.

Table 2.487: Start Time Support encoding

Bit	Indicates support for	Version
0	Start now. Refer to Section 2.2.86.5.	1
1	Start Hour and Minute. Refer to Section 2.2.86.6.	1
2	Calendar time. Refer to Section 2.2.86.7.	1
3	Weekdays. Refer to Section 2.2.86.8.	1
4..5	Reserved	N/A

Each bit indicates the support for a given Start Time option.

CC:0053.01.02.11.006 The value 1 MUST indicate that the node supports the corresponding start time option.

CC:0053.01.02.11.007 The value 0 MUST indicate that the node does not support the corresponding start time option.

While support is advertised as a bitmask, the actual functionality is triggered by different combinations of Schedule Set start time fields. The following subsections outline the mandatory supported combinations when advertising support for the corresponding option.

2.2.86.5 Start Time Option: Start Now

The start now option is used to make a configured schedule start immediately (upon reception of the Schedule Set Command).

CC:0053.01.02.11.008 A node that supports the start now option, MUST support the creation of a schedule starting immediately when all start time fields are left unspecified.

2.2.86.6 Start Time Option: Hour and Minute

The start hour and minute option is used to make a schedule start at a specified time of the day.

CC:0053.01.02.11.009 If Start Hour and Start Minute are specified, node supporting this option MUST activate the schedule at the specified start time.

CC:0053.01.02.11.00A A node supporting the *Hour and Minute* start option MUST support the creation of a schedule with the following start time fields' combination:

- Every day @ Hour:Minute

  (YYMMDD = 0xFF, 0x00, 0x00, Weekdays = 0x00, HH:MM = **Hour:Minute**)

CC:0053.01.02.11.00B A node that also supports the *Weekdays* start option MUST support the creation of a schedule with the following start time fields' combination:

- Same weekday(s) every week @ Hour:Minute

  (YYMMDD = 0xFF, 0x00, 0x00, Weekdays = **weekdays**, HH:MM = **Hour:Minute**)

CC:0053.01.02.11.00C A node that also supports the *Calendar Time* start option MUST support the creation of a schedule with the following start time fields' combination:

- Same day every month @ Hour:Minute

  (YYMMDD = 0xFF, 0x00, **day**, Weekdays = 0x00, HH:MM = **Hour:Minute**)

- Same day every year @ Hour:Minute

  (YYMMDD = 0xFF, **month**, **day**, Weekdays = 0x00, HH:MM = **Hour:Minute**)
- One specific date in a specific year @ Hour:Minute

  (YYMMDD = **year**, **month**, **day**, Weekdays = 0x00, HH:MM = **Hour:Minute**)

2.2.86.7 Start Time Option: Calendar Time

The calendar time option is used to make a schedule start at a specified date.

CC:0053.01.02.11.00D If Start Year, Start Month and Start Day are specified, node supporting this option MUST activate the schedule at the specified date(s).

CC:0053.01.02.11.00E A node that supports the *Calendar* Time option MUST support the creation of a schedule with the following start time fields' combination:

- Same day every month @ 00:00

  (YYMMDD = 0xFF, 0x00, **day**, Weekdays = 0x00, HH:MM = 0x1F, 0x3F)
- Same day every year @ 00:00

  (YYMMDD = 0xFF, **month**, **day**, Weekdays = 0x00, HH:MM = 0x1F, 0x3F)
- One specific date in a specific year @ 00:00

  (YYMMDD = **year**, **month**, **day**, Weekdays = 0x00, HH:MM = 0x1F, 0x3F)

2.2.86.8 Start Time Option: Weekday

The start weekday option is used to make a schedule start at on specified weekdays.

CC:0053.01.02.11.00F If weekdays are specified, node supporting this option MUST activate the schedule on the specified weekdays.

A node that supports the *Weekdays* option MUST support the creation of a schedule with the following start time fields' combination:

- Same weekday(s) every week @ 00:00

  (YYMMDD = 0xFF, 0x00, 0x00, Weekdays = **weekdays**, HH:MM = 0x1F, 0x3F)

Fall Back Support (1 bit)

This bit is used to advertise the support of the Fall Back Schedule (Schedule ID = 0xFE).

CC:0053.01.02.11.011 The bit MUST be set to 1 if the node supports the Fall Back Schedule.

CC:0053.01.02.11.012 The bit MUST be set to 0 if the node does not support the Fall Back Schedule.

Support Enable/Disable (1 bit)

This bit is used to advertise the support for enabling/disabling set/used schedules via the Schedule State Set Command.

CC:0053.01.02.11.013 The bit MUST be set to 0 if the node supports enabling/disabling schedules.

CC:0053.01.02.11.014 The bit MUST be set to 1 if the node does not support enabling/disabling schedules.

CC:0053.01.02.11.015 A node that does not support enabling/disabling schedules MUST consider every set/used schedule ID as enabled.

Number of supported CC (8 bits)

CC:0053.01.02.11.016 This field MUST advertise the number of supported Command Classes advertised in this command with the Supported CC and Supported Command fields.

CC:0053.01.02.11.017

The value 0xFF MUST be used if all the command classes supported by the node are also supported for scheduling. If this field is set to 0xFF:

CC:0053.01.02.11.018 • This command MUST NOT carry any Command Class entries (Supported CC and Supported Command fields).

CC:0053.01.02.11.019 • The node MUST support "Get" as well as "Set" commands. (i.e. Supported Commands fields set to 0x00)

Supported CC (N * 8 bits)

This field is used to advertise the list of Command Classes that can be scheduled by the node.

CC:0053.01.02.11.01A The length of this field MUST be according to the Number of supported CC field value.

CC:0053.01.02.11.01B A sending node MUST support every Command Class advertised in this field.

CC:0053.01.02.12.002 A sending node SHOULD NOT advertise the Supervision Command Class in this field.

CC:0053.01.02.11.01C A sending node MUST NOT advertise the following encapsulation Command Classes:

- S0/S2 Command Class
- Transport Service Command Class
- Multi Channel Command Class
- Multi Command Command Class

Supported Command (N * 2 bits)

CC:0053.01.02.11.01D This field MUST advertise the supported commands for the actual command class entry. This field MUST comply with Table 2.488.

Table 2.488: Supported Command

Value	Description	Version
0x00	Both Set and Get Commands are supported	1
0x01	Only Set Commands are supported	1
0x02	Only Get Commands are supported	1
0x03	Reserved	N/A

CC:0053.01.02.11.01E Certain command classes commands do not contain the string "Set" or "Get" in their name. Nodes MUST consider all commands mandating to return a response to be of type "Get" and all other commands to be of type "Set".

CC:0053.01.02.11.01F The length of this field MUST be according to the Number of supported CC field value.

Override Support (1 bit)

This field is used to advertise support for the Override Schedule (Schedule ID = 0xFF).

CC:0053.01.02.11.020 This bit MUST be set to 1 if the node supports the Override Schedule.

CC:0053.01.02.11.021 This bit MUST be set to 0 if the node does not support the Override Schedule.

Supported Override Types (7 bits)

This field is used to advertise the supported Override Schedule duration types.

CC:0053.01.02.11.022 A receiving node MUST ignore this bit mask if the Override Schedule is not supported by the sending node.

CC:0053.01.02.11.023 This field MUST be treated as a bitmask and MUST be encoded according to Table 2.489.

Table 2.489: Schedule Supported Report::Supported Override Types encoding

Bit	Name	Description	Version
CC:0053.01.02.11.024 0	Advance	The override schedule MUST run until the start of the next regular schedule.	1
CC:0053.01.02.11.025 1	Run forever	The override schedule MUST run until the schedule is removed.	1
2..6	Reserved	Reserved	N/A

2.2.86.9 Schedule Set Command

This command is used to create a new schedule or modify an existing schedule.

This command MUST enable the schedule if a new schedule is created.

This command MUST NOT change the enabled/disabled state of existing schedules.

If the receiving node supports S0 or S2 Command Class, it MUST NOT process the schedule creation if any of the scheduled command classes is not supported at the security level of the received Schedule Set Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_SET (0x03)							
Schedule ID							
Reserved							
Start Year							
Reserved				Start Month			
Reserved			Start Day of Month				
Reserved	Start Weekday						
Duration Type		Start Hour					
Reserved	Start Minute						
Duration Byte 1 MSB							
Duration Byte 2 LSB							
Reports to Follow							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Reserved

CC:0053.01.03.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Schedule ID (8 bits)

This field is used to indicate which Schedule ID is being set.

CC:0053.01.03.11.005 Values in the range 1..[Number of supported Schedule] MUST indicate regular Schedule IDs

CC:0053.01.03.11.006 The value 0xFE MUST indicate the Fall Back Schedule.

CC:0053.01.03.11.007 The value 0xFF MUST indicate the Override Schedule.

CC:0053.01.03.11.008 A receiving node MUST ignore this command if a non-supported Schedule ID is specified.

If this field is set to 0xFF:

CC:0053.01.03.12.001 • The sending node SHOULD set the start time fields to the Start Now pattern.

CC:0053.01.03.11.009 • The receiving node MUST respond immediately to an Override Schedule .

If this field is set to 0xFE:

CC:0053.01.03.12.002 • The sending node SHOULD set the start time fields and duration fields to 0x00.

CC:0053.01.03.11.00A • The receiving node MUST ignore the start time fields and duration fields

2.2.86.9.1 Start Time fields

The schedule start time is defined by the following fields:

- Start Year
- Start Month
- Start Day of Month
- Start Weekday
- Start Hour
- Start Minute

CC:0053.01.03.11.00B A receiving node MUST support the start time fields combinations indicated in Section 2.2.86.5, Section 2.2.86.6, Section 2.2.86.7 and Section 2.2.86.8 if the corresponding start time option is supported.

CC:0053.01.03.11.00C If none of the start time fields are specified, the schedule MUST start immediately if the start now
CC:0053.01.03.11.00D option is supported. In this case, the schedule MUST NOT be activated again at a later time unless receiving another Schedule Set Command.

Start Year (8 bits)

CC:0053.01.03.11.00E This field is used to set the year for which the schedule is to start. This field MUST be in the range 0..99 or set to 0xFF.

CC:0053.01.03.11.00F Values in the range 0..99 MUST indicate the actual year last 2 digits. A node MUST accept a value lower than the current year to allow schedules starting in the next century.

CC:0053.01.03.11.010 The value 0xFF MUST be used to indicate that the start year is unspecified.

CC:0053.01.03.11.011 This field MUST be ignored and treated as unspecified by a receiving node advertising no support for the Calendar time start time option in the Schedule Supported Report Command.

Start Month (4 bits)

CC:0053.01.03.11.012 This field is used to set the month for which the schedule is to start. This field MUST be in the range 0..12.

CC:0053.01.03.11.013 Values in the range 1..12 MUST indicate the actual start month.

CC:0053.01.03.11.014 The value 0x00 MUST be used to indicate that the start month is unspecified.

CC:0053.01.03.11.015 This field MUST be ignored and treated as unspecified by a receiving node advertising no support for the Calendar time start time option in the Schedule Supported Report Command.

Start Day of Month (5 bits)

CC:0053.01.03.11.016 This field is used to set the day of month for which the schedule is to start. This field MUST be in the range 0..31.

CC:0053.01.03.11.017 Values in the range 1..31 MUST indicate the actual start day of the month. If the specified value does
CC:0053.01.03.13.001 not exist in the actual Year/Month tuple (28/29/30 day month), a receiving node MAY ignore this command.

CC:0053.01.03.11.018 The value 0x00 MUST be used to indicate that the day of the month is unspecified.

CC:0053.01.03.11.019

This field MUST be ignored and treated as unspecified by a receiving node advertising no support for the Calendar time start time option in the Schedule Supported Report Command.

Start Weekday (7 bits)

CC:0053.01.03.11.01A This field is used to set one or more weekdays for which the schedule is to start. This field MUST be treated a as bitmask and MUST be encoded according to Table 2.490.

Table 2.490: Weekday bitmask encoding

Bit	Description	Version
0	Monday	1
1	Tuesday	1
2	Wednesday	1
3	Thursday	1
4	Friday	1
5	Saturday	1
6	Sunday	1

CC:0053.01.03.11.01B The value 1 MUST indicate that the specified Schedule ID MUST start on the corresponding day.

CC:0053.01.03.11.01C The value 0 MUST indicate that the specified Schedule ID MUST NOT start on the corresponding day.

CC:0053.01.03.11.01D If any of the Start Year, Start Month or Start Day of Month field is specified, this field MUST be set to 0x00 by a sending node and MUST be ignored by a receiving node.

CC:0053.01.03.11.01E This field MUST be ignored and treated as unspecified by a receiving node advertising no support for the Weekday start time option in the Schedule Supported Report Command.

Start Hour (5 bits)

CC:0053.01.03.11.01F This field is used to set the hour for which the schedule is to start. This field MUST be in the range 0..23 or set to 0x1F.

CC:0053.01.03.11.020 Values in the range 0..23 MUST indicate the start hour of the day.

CC:0053.01.03.11.021 The value 0x1F (31) MUST be used to indicate that the start hour is unspecified.

CC:0053.01.03.11.022 This field MUST be ignored and treated as unspecified by a receiving node advertising no support for the hour and minute start time option in the Schedule Supported Report Command.

Start Minute (6 bits)

CC:0053.01.03.11.023 This field is used to set the minute for which the schedule is to start. This field MUST be in the range 0..59 or set to 0x3F.

CC:0053.01.03.11.024 Values in the range 0..59 MUST indicate the starting minute of the schedule.

CC:0053.01.03.11.025 The value 0x3F (63) MUST be used to indicate that the start minute is unspecified.

CC:0053.01.03.11.026 This field MUST be ignored and treated as unspecified by a receiving node advertising no support for the hour and minute start time option in the Schedule Supported Report Command.

2.2.86.9.2 Duration fields

The schedule duration is defined by the following fields:

- Duration Type
- Duration

Duration Type (3 bits)

CC:0053.01.03.11.027 This field is used to indicate how to interpret the duration field. This field MUST be encoded according to Table 2.491.

Table 2.491: Duration Type encoding

Value	Description	Version
0x00	The duration field is expressed in Minutes.	1
0x01	The duration field is expressed in Hours.	1
0x02	The duration field is expressed in Days.	1
0x03	Override: The duration field is indicating the Override Type. This value MUST NOT be used if the Schedule ID field is not set to 0xFF	1

CC:0053.01.03.11.028

CC:0053.01.03.11.029 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Duration (16 bits)

CC:0053.01.03.11.02A This field is used to specify the duration of the schedule being set. The Duration Type field MUST be inspected before interpreting this field.

CC:0053.01.03.11.02B If the Duration Type field is not set to Override (0x03), this field MUST indicate the duration of the actual schedule. The Duration Type field indicates the unit of this field.

CC:0053.01.03.11.02C If the Duration Type field is set to Override (0x03), the LSB of this field MUST be encoded according to Table 2.492 and the MSB of this field MUST be set to 0x00.

Table 2.492: Override Types duration

Name	Duration value (LSB)	Description	Version
Advance		The override schedule MUST run until the start of the next regular schedule..	1
Run forever	0x02	The override schedule MUST run until it is removed.	1
Reserved	0x03..0x07	Reserved	N/A

CC:0053.01.03.11.02D

CC:0053.01.03.11.02E

2.2.86.9.3 Command and other fields

Reports to Follow (8 bits)

CC:0053.01.03.11.02F This field MUST be used if multiple Schedule Set commands are used to define a single schedule.

CC:0053.01.03.11.030 The value MUST indicate the remaining number of Schedule Set Commands. The header bytes (Schedule ID, Start time, Duration, etc.) MUST be the same for all Schedule Set Commands.

Number of Cmd to Follow (8 bits)

This field is used to advertise the number of command blocks (Cmd Length/Cmd Byte fields units) included within the actual Schedule Set Command (represented by P in the command structure).

CC:0053.01.03.11.031 Each command block MUST comprise the Cmd Length and Cmd Byte fields.

CC:0053.01.03.11.032 CC:0053.01.03.11.033 If the list of scheduled commands and their payload is longer than the maximum available Z-Wave MAC frame size, multiple Schedule Set commands MUST be used to send the complete list of scheduled commands. In this case, this field MUST advertise how many commands are included in the actual/current Schedule Set Command.

Cmd Length (8 bits)

This field is used to indicate the length in byte of the corresponding Cmd Byte field.

Cmd Byte (N bytes)

This field is used to set the commands to be executed during the actual Schedule.

CC:0053.01.03.11.034 This field MUST carry a complete command including Command Class identifier, Command identifier and the required parameter bytes.

CC:0053.01.03.11.035

The length of each Cmd Byte entry MUST be according to the corresponding Cmd Length field entry.

CC:0053.01.03.11.036 The number of Cmd Byte entries MUST be according to the Number of Cmd to Follow field.

CC:0053.01.03.11.037 If a schedule includes commands mandating to return a response, responses MUST be returned to the node that has set the schedule.

CC:0053.01.03.12.003 A receiving node SHOULD ignore the entire Schedule Set Command if this field carries a command not advertised as supported for scheduling in the Schedule Supported Report Command. In any

CC:0053.01.03.11.038 case, the receiving node MUST NOT accept commands non-supported for scheduling as part of the Schedule.

CC:0053.01.03.12.004 A controlling node SHOULD avoid creating conflicting schedules and a supporting node SHOULD discard a new conflicting schedule, even if the conflict is with a temporarily disabled schedule.

2.2.86.10 Schedule Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_GET (0x04)							
Schedule ID							

Schedule ID (8 bits)

CC:0053.01.04.11.004 This field MUST carry the requested Schedule ID.

2.2.86.11 Schedule Report Command

This command is used to report the configuration for a specific schedule.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_REPORT (0x05)							
Schedule ID							
Reserved							
Start Year							
Active_ID				Start Month			
Reserved			Start Day of Month				
Reserved	Start Weekday						
Duration Type			Start Hour				
Reserved		Start Minute					
Duration Byte 1							
Duration Byte 2							
Report to Follow							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Fields not described below are described in the Schedule Set Command; refer to Section 2.2.86.9.

Schedule ID (8 bits)

This field is used to indicate the advertised Schedule ID.

CC:0053.01.05.11.001 If this command is advertising an unused or unsupported Schedule ID, all parameters until Number of Cmd to Follow MUST be set to 0x00. The Cmd Length and Cmd Byte fields MUST be omitted.

Active_ID (4 bits)

CC:0053.01.05.11.002 This field is used to advertise the status for the requested Schedule ID. This field MUST be encoded according to Table 2.494.

Duration (16 bits)

This field is used to advertise either the configured duration for a schedule or how much time is left before the schedule becomes inactive.

CC:0053.01.05.11.003 The Duration Type, Start Time and Schedule ID fields MUST be inspected before interpreting this field. This field MUST advertise a value according to Table 2.493.

Table 2.493: Duration field usage

Duration Type	Schedule ID	Start Time	Advertised duration
Time (0x00..0x02)	Regular	Specified start time	Configured duration
Time (0x00..0x02)	Regular of Override	Start now	Remaining time
Override (0x03)	Override (0xFF)	Start now	Configured override type (Table 2.492)

2.2.86.12 Schedule Remove Command

This command is used to request the removal of one or all Schedules in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_REMOVE (0x06)							
Schedule ID							

Schedule ID (8 bits)

This field is used to indicate the schedule that is to be removed/erased.

CC:0053.01.06.11.001 If this field is set to 0x00, all configured schedules MUST be removed.

CC:0053.01.06.11.002 If this field is set to 0xFE, the Fall Back Schedule MUST be removed (if defined).

CC:0053.01.06.11.003 If this field is set to 0xFF, the Override Schedule MUST be removed (if defined).

CC:0053.01.06.11.004 If this field is in the range 0x01..[Number of supported Schedules], the receiving node MUST remove the corresponding schedule.

2.2.86.13 Schedule State Set Command

This command is used to enable or disable a schedule.

CC:0053.01.07.11.001 This command MUST be ignored by a node advertising no support for Enabling/Disabling schedules in the Schedule Supported Report Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_STATE_SET (0x07)							
Schedule ID							
Schedule State							

Schedule ID (8 bits)

This field is used to indicate the schedule that is to be enabled or disabled.

CC:0053.01.07.11.002 The value 0x00 MUST cause the receiving node to change the state for all schedules, except for the Override Schedule.

CC:0053.01.07.11.003 Values in the range 0x01..[number of supported schedules] and 0xFE MUST indicate the actual schedule ID which state is to be updated.

CC:0053.01.07.11.004 The value 0xFF MUST be ignored by a receiving node. Override schedules cannot be enabled/disabled and may only be removed using the Schedule Remove Command.

Schedule state (8 bits)

This field is used to indicate the new state of one or more schedules.

CC:0053.01.07.11.005 The value 0xFF MUST indicate that the actual schedule MUST be enabled.

CC:0053.01.07.11.006 The value 0x00 MUST indicate that the actual schedule MUST be disabled.

CC:0053.01.07.11.007 Disabling a schedule MUST NOT cause the schedule settings to be permanently removed. Schedules may be permanently removed via the Schedule Remove Command.

CC:0053.01.07.11.008 If a schedule is enabled, the receiving node MUST assess if the actual schedule should have been active and activate it accordingly.

CC:0053.01.07.11.009 If a schedule is disabled while being active, the receiving node MUST treat the disabling operation as if the schedule ended.

2.2.86.14 Schedule State Get Command

This command is used to request the status of all schedules supported by a node.

CC:0053.01.08.11.001 The Schedule State Report Command MUST be returned in response to this command.

CC:0053.01.08.11.002 This command MUST NOT be issued via multicast addressing.

CC:0053.01.08.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_STATE_GET (0x08)							

2.2.86.15 Schedule State Report Command

This command is used to advertise the status of all schedules supported by a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_STATE_REPORT (0x09)							
Number of Supported Schedule ID							
Reports to Follow							Override
Active_ID 2				Active_ID 1			
...							
Active_ID N				Active_ID N-1			

Number of Supported Schedule ID (8 bits)

CC:0053.01.09.11.001 This field is used to advertise the number of supported regular schedule IDs. This field's value MUST be the same as the value advertised in the Schedule Supported Report Command.

Override (1 bit)

CC:0053.01.09.11.002 This field MUST be used to advertise the status of the Override Schedule (ID = 0xFF).

CC:0053.01.09.11.003 If the Override Schedule is active, this bit MUST be set to 1.

CC:0053.01.09.11.004 If the Override Schedule is inactive, this bit MUST be set to 0.

Reports to Follow (7 bits)

The length of the Active_ID field may be larger than the maximum available Z-Wave MAC frame size.

CC:0053.01.09.11.005 This field MUST be used to advertise the number of commands following this command in order to advertise the status of all Schedules IDs.

CC:0053.01.09.12.001 A controlling node SHOULD use this value to detect missing reports.

Active_ID (N * 4 bits)

This field is used to advertise the status for each supported schedule ID. Four bits units are used for each Schedule ID to indicate the status. It means that:

- Bits 0 to 3 in Active_ID byte 1 represent Schedule ID = 1
- Bits 4 to 7 in Active_ID byte 1 represent Schedule ID = 2
- Bits 0 to 3 in Active_ID byte 2 represent Schedule ID = 3
- ...

CC:0053.01.09.11.006 The size of this field MUST be the smallest number of bytes needed to advertise the number of supported Schedule IDs advertised in this command.

CC:0053.01.09.11.007 If several Reports are returned (using the Reports to follow field), the first Report advertises Schedule ID 1 to Schedule ID N and the subsequent report MUST be interpreted as representing Schedule ID starting from N+1.

CC:0053.01.09.11.008 Each 4 bits unit MUST be encoded according to Table 2.494.

Table 2.494: Active_ID encoding

Hex	Description	Detailed description	Version
0x00	Not used	The Schedule ID is not used. (not set/configured) or unsupported	1
0x01	[DEPRECATED] Override + Not used	[DEPRECATED] A sending node SHOULD NOT use this status. It is RECOMMENDED to use 0x00 instead. A receiving node MUST interpret this status as Active_ID = 0x00 (and the override field set to 1 for the Schedule State Report Command).	1
0x02	Not Active	The Schedule ID is used, enabled and currently not active.	1
0x03	Active	The Schedule ID is used, enabled and currently active.	1
0x04	Disabled	The Schedule ID is used and disabled.	1
0x05	Override + Active	The Schedule ID is used, enabled and should currently be active but it is suspended by the Override Schedule The override field MUST be set to 1 when using this status in the Schedule State Report Command.	1
0x06	[DEPRECATED] Override + Not Active	[DEPRECATED] A sending node SHOULD NOT use this status. It is RECOMMENDED to use 0x02 instead. A receiving node MUST interpret this status as Active_ID = 0x02 (and the override field set to 1 for the Schedule State Report Command.	1
0x07	[DEPRECATED] Override + Disabled	[DEPRECATED] A sending node SHOULD NOT use this status. It is RECOMMENDED to use 0x04 instead. A receiving node MUST interpret this status as Active_ID = 0x04 (and the override field set to 1 for the Schedule State Report Command).	1

CC:0053.01.09.11.00D All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.87 Schedule Command Class, version 2

The Schedule Command Class version 2 allows scheduling the execution of commands for a given duration in a supporting node. It is a generic command class that may be used to schedule commands from any other Command Class.

2.2.87.1 Compatibility Considerations

The Schedule Command Class, version 2 is backwards compatible with version 1.

CC:0053.02.00.21.001 A node supporting the Schedule Command Class, version 2 MUST also support version 1.

Schedule Command Class, version 2 introduces the following:

- Schedule ID Blocks
- Clarification on using Schedule Command Class with Security

All commands or fields not described in this version remain unchanged from version 1.

2.2.87.1.1 Schedule ID Blocks

In Schedule Command Class version 1, all Schedule IDs have to support the same Start Time Options, Override Types and supported commands.

The Schedule ID Blocks allows a node to implement several Schedule ID pools sharing the same scheduling functionalities.

CC:0053.02.00.21.002 Each Schedule ID Block MUST have its own range of Schedule ID's. The ID for each Block MUST be in the range 1..[Number of Supported Schedule IDs].

CC:0053.02.00.21.003 Schedule ID Block = 1 is the default Block. A version 1 node MUST be assumed to be using Schedule ID block 1.

2.2.87.1.2 Schedule Command Class with Security

The Schedule Command Class may be implemented by secure devices. Depending on the type of device, a given Command Class may be supported only via secure communication or via secure as well as unsecure communication.

The following command class categories may be considered:

- **Always unsecure**:

  Examples of command classes which are always supported via unsecure communication - as well as via secure communication if the device is securely included (e.g Z-Wave Plus Info Command Class).

- **Migrate to secure**:

  Examples of command classes which are supported via unsecure communication if the device is not securely included - but **only via secure** communication if the device is securely included. (e.g. Multilevel Switch or Association Command Class)

- **Secure only**:

  Examples of command classes which are supported **only via secure** communication. (e.g. Door Lock Command Class)

CC:0053.02.00.21.004 To reflect the above mentioned dynamic support scenarios, a device MUST advertise the supported Command Classes for scheduling in a way that matches the current secure/non-secure supported Command Class lists found in the NIF and S0/S2 Commands Supported Report Commands.

Clarification is added to the Schedule Supported Report Command and Schedule Set Command for proper reporting and configuration of unsecure and secure command classes scheduling.

CC:0053.02.00.22.001 It is RECOMMENDED to support the Schedule Command Class only at the highest security level in order to avoid dynamic listing.

2.2.87.2 Schedule Supported Get Command

This command is used to query the properties of a node.

CC:0053.02.01.11.001 The Schedule Supported Report Command MUST be returned in response to this command.

CC:0053.02.01.11.002 If the receiving node supports S0 or S2 Command Class and this command is issued via non-secure communication, the receiving node MUST advertise Command Classes that can be scheduled but also supported at the received security level.

CC:0053.02.01.11.003 This command MUST NOT be issued via multicast addressing.

CC:0053.02.01.11.004 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_GET (0x01)							
Schedule ID Block							

Schedule ID Block (1 byte)

This field is used to request a particular Schedule ID Block.

CC:0053.02.01.11.005 If this field is set to 0x00, the receiving node MUST return the default Schedule ID Block (ID = 1).

CC:0053.02.01.11.006 A node receiving a version 1 Schedule Supported Get Command (without the Schedule ID Block field) MUST return a response for the default Schedule ID Block (ID = 1).

CC:0053.02.01.12.001 A node receiving a non-supported Schedule ID Block SHOULD return a report for the default Schedule ID Block.

2.2.87.3 Schedule Supported Report Command

This command is used to advertise the scheduling functionalities supported by a node for a given Schedule ID Block.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_REPORT (0x02)							
Number of Supported Schedule IDs							
Support Enable/Disable	Fallback Support	Start Time Support					
Number of Supported CC							
Supported CC 1							
Reserved							Supported Command 1
...							
Supported CC N							
Reserved							Supported Command N
Override Support	Supported Override Types						
Schedule ID Block							
Number of Supported Schedule Blocks							

Fields not described below remain unchanged from version 1.

All fields contained in this command advertise the scheduling supported functionalities for the actual Schedule ID Block.

Schedule ID Block (8 bits)

This field is used to indicate the actual Schedule ID Block for which the supported scheduling functionalities are advertised.

CC:0053.02.02.12.001 The advertised supported functionalities SHOULD be different for every Schedule ID Block.

Number of Schedule ID Blocks (8 bits)

CC:0053.02.02.11.001 This field MUST advertise the total number of Schedule ID blocks supported by the node.

CC:0053.02.02.11.002 This field MUST be in the range 1..255.

CC:0053.02.02.11.003 The implemented Schedule ID Blocks MUST be in the range 1..[Number of Supported Schedule ID Blocks]. i.e. a node supporting 10 Schedules ID Blocks MUST accept Schedule ID Blocks values in the range 1..10.

2.2.87.4 Schedule Set Command

This command is used to create a new schedule or modify an existing schedule.

CC:0053.02.03.11.001 This command MUST enable the schedule if a new schedule is created.

CC:0053.02.03.11.002 This command MUST NOT change the enabled/disabled state of existing schedules

CC:0053.02.03.11.003 If the receiving node supports S0 or S2 Command Class, it MUST NOT process the schedule creation if any of the scheduled command classes is not supported at the security level of the received Schedule Set Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_SET (0x03)							
Schedule ID							
Schedule ID Block							
Start Year							
Reserved				Start Month			
Reserved			Start Day of Month				
Reserved	Start Weekday						
Duration Type		Start Hour					
Reserved	Start Minute						
Duration Byte 1 MSB							
Duration Byte 2 LSB							
Reports to Follow							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Fields not described below remain unchanged from version 1.

Schedule ID Block (8 bits)

This field is used to specify the Schedule ID Block in which the specified Schedule ID is being set.

CC:0053.02.03.11.004 A node receiving a version 1 Schedule Set Command (with the Schedule ID Block field set to 0x00) MUST assume the default Schedule ID Block (ID = 0x01).

2.2.87.5 Schedule Get Command

This command is used to request the configuration for a specific schedule ID.

CC:0053.02.04.11.001 The Schedule Report Command MUST be returned in response to this command.

CC:0053.02.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0053.02.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_GET (0x04)							
Schedule ID							
Schedule ID Block							

Fields not described below remain unchanged from version 1.

Schedule ID Block (8 bits)

This field is used to specify the requested Schedule ID Block.

CC:0053.02.04.11.004 A node receiving a version 1 Schedule Get Command (without the Schedule ID Block field) MUST return a response for the default Schedule ID Block (ID = 1).

2.2.87.6 Schedule Report Command

This command is used to advertise the configuration for a specific schedule.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_REPORT (0x05)							
Schedule ID Block							
Reserved							
Start Year							
Active_ID				Start Month			
Reserved				Start Day of Month			
Reserved			Start Weekday				
Duration Type			Start Hour				
Reserved		Start Minute					
Duration Byte 1							
Duration Byte 2							
Report to Follow							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Fields not described below remain unchanged from version 1.

Schedule ID Block (8 bits)

This field is used to specify the advertised Schedule ID Block.

2.2.87.7 Schedule Remove Command

This command is used to request the removal of one or all Schedules in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_REMOVE (0x06)							
Schedule ID							
Schedule ID Block							

Schedule ID (8 bits)

This field is used to indicate the schedule that is to be removed.

CC:0053.02.06.11.001 If this field is set to 0x00, all schedules within the Schedule ID Block MUST be removed.

CC:0053.02.06.11.002 If this field is set to 0xFE, the Fall Back Schedule MUST be removed (if defined for the Schedule ID Block).

CC:0053.02.06.11.003 If this field is set to 0xFF, the Override Schedule MUST be removed (if defined for the Schedule ID Block).

CC:0053.02.06.11.004 If this field is in the range 0x01..[Number of supported Schedules], the receiving node MUST remove the corresponding schedule in the Schedule ID Block.

Schedule ID Block (8 bits)

This field is used to indicate the Schedule ID Block in which a schedule is being removed.

CC:0053.02.06.11.005 A node receiving a version 1 Schedule Set Command (without the Schedule ID Block field) MUST assume the default Schedule ID Block (ID = 0x01).

CC:0053.02.06.11.006 A node receiving this command with **Schedule ID = 0x00** and **Schedule ID Block = 0x00** MUST remove all Schedule IDs in all Schedule ID Blocks

CC:0053.02.06.11.007 A node receiving this command with **Schedule ID = 0x00** and **Schedule ID Block 0x00** MUST remove all Schedule IDs from the specified Schedule ID Block.

CC:0053.02.06.11.008 A node receiving this command with **Schedule ID 0x00** and **Schedule ID Block = 0x00** MUST ignore the command

2.2.87.8 Schedule State Set Command

This command is used to enable or disable one or more schedules.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_STATE_SET (0x07)							
Schedule ID							
Schedule State							
Schedule ID Block							

Fields not described below remain unchanged from version 1.

Schedule ID Block (8 bits)

This field is used to specify the Schedule ID Block in which one or more schedule are being enabled or disabled.

CC:0053.02.07.11.001 This command MUST be ignored by a node advertising no support for Enabling/Disabling schedules in the Schedule Supported Report Command for the given Schedule ID Block.

CC:0053.02.07.11.002 A node receiving a version 1 Schedule Set Command (without the Schedule ID Block field) MUST assume the default Schedule ID Block (ID = 0x01).

CC:0053.02.07.11.003 A node receiving this command with **Schedule ID = 0x00** and **Schedule ID Block = 0x00** MUST enable/disable all Schedule IDs from all Schedule ID Blocks.

CC:0053.02.07.11.004 A node receiving this command with **Schedule ID = 0x00** and **Schedule ID Block 0x00** MUST enable/disable all Schedule IDs from the specified Schedule ID Block.

CC:0053.02.07.11.005 A node receiving this command with **Schedule ID 0x00** and **Schedule ID Block = 0x00** MUST ignore the command.

2.2.87.9 Schedule State Get Command

This command is used to request the status of all schedules within a Schedule ID Block supported by a node.

CC:0053.02.08.11.001 The Schedule State Report Command MUST be returned in response to this command.

CC:0053.02.08.11.002 This command MUST NOT be issued via multicast addressing.

CC:0053.02.08.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_STATE_GET (0x08)							
Schedule ID Block							

Schedule ID Block (8 bits)

This field is used to request a particular Schedule ID Block.

CC:0053.02.08.11.004 A node receiving a version 1 Schedule Supported Get Command (without the Schedule ID Block field) MUST return a response for the default Schedule ID Block (ID = 1).

2.2.87.10 Schedule State Report Command

This command is used to advertise the status of all schedules supported by a device in a given Schedule ID block.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_STATE_REPORT (0x09)							
Number of Supported Schedule ID							
Reports to Follow							Override
Active_ID 2				Active_ID 1			
...							
Active_ID N				Active_ID N-1			
Schedule ID Block							

Fields not described below remain unchanged from version 1.

Number of Supported Schedule ID (8 bits)

CC:0053.02.09.11.001 This field is used to advertise the number of supported regular schedule IDs for the actual Schedule ID Block. This field's value MUST be the same as the value advertised in the Schedule Supported Report Command.

Schedule ID Block (8 bits)

This field is used to specify the Schedule ID Block in which all schedule states are advertised.

CC:0053.02.09.11.002 If several Reports are returned (using the Reports to Follow field), this field MUST be present in every Report.

2.2.88 Schedule Command Class, version 3

The Schedule Command Class, version 3 allows scheduling the execution of commands for a given duration in a supporting node. It is a generic command class that may be used to schedule commands of any other Command Class.

2.2.88.1 Terminology

The Schedule Command Class version 3 introduces new terminology, complementing the terminology introduced in The Schedule Command Class version 1:

Regular schedules may be triggered repeatedly by two mechanisms.

From version 1, a **Repeating Schedule** can be created by setting parts of the start time to unspecified values, e.g. if month is unspecified, the schedule can start the first day of each month.

From version 3, **Recurrence** settings may specify additional periodical triggers, e.g. every second day measured from the most recent trigger of the regular schedule.

2.2.88.2 Compatibility Considerations

The Schedule Command Class, version 3 is backwards compatible with version 2.

CC:0053.03.00.21.001 A node supporting the Schedule Command Class, version 3 MUST also support version 2.

The Schedule Command Class version 3 introduces the “Recurring Mode” start mode and “Time from now” start time option.

The “Time from now” start time option allows to set a schedule to start a after a given time from the reception of the command.

The “Recurring Mode” start mode allows to set a schedule to restart repeatedly at fixed intervals since the last activation, e.g. every second day or every 6 months since the last activation.

The Active_ID advertised in the Schedule Report Command of versions 1 and 2 is now overloaded with the Recurrence Offset field. A new AID_RO_CTL flag in the Schedule Get Command controls which value is actually returned.

All commands or fields not described in this version remain unchanged from version 2.

2.2.88.3 Schedule Supported Report Command

This command is used to advertise the scheduling functionalities supported by a node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_REPORT (0x02)							
Number of Supported Schedule IDs							
Support Enable/Disable	Fallback Support	Start Time Support					
Number of Supported CC							
Supported CC 1							
Reserved						Supported Command 1	
...							
Supported CC N							
Reserved						Supported Command N	
Override Support	Supported Override Types						
Schedule ID Block							
Number of Supported Schedule Blocks							

Fields not described below remain unchanged from version 2.

Start Time Support (6 bits)

This field is used to advertise the start time options supported by the sending node.

CC:0053.03.02.11.001 Regular schedules MUST support the advertised start time options.

CC:0053.03.02.12.001 The Override Schedule SHOULD support the advertised start time options.

CC:0053.03.02.11.002 The Override Schedule MUST NOT support the Recurring Mode.

CC:0053.03.02.11.003 This field MUST be treated as a bitmask and MUST be encoded according to Table 2.495.

Table 2.495: Start Time Support

Category	Bit	Indicates support for	Version
Start Time option	0	Start now. Refer to Section 2.2.86.5.	1
	1	Start Hour and Minute. Refer to Section 2.2.86.6.	1
	2	Calendar time. Refer to Section 2.2.86.7.	1
	3	Weekdays. Refer to Section 2.2.86.8.	1
	4	Time from now. Refer to Section 2.2.88.3.1.	3
Start Time mode	5	Recurring Mode. Refer to Section 2.2.88.3.2.	3

Each bit indicates the support for a given Start Time option or mode.

CC:0053.03.02.11.004 The value 1 MUST indicate that the node supports the corresponding start time option/mode.

CC:0053.03.02.11.005 The value 0 MUST indicate that the node does not support the corresponding start time option/mode.

While support is advertised as a bitmap, the actual functionality is triggered by different combinations of Schedule Set start time fields. The following subsections outline the mandatory supported combinations when advertising support for the corresponding option.

2.2.88.3.1 Start Time Option: Start from now

The start now option is used to make a configured schedule start after the indicated time has elapsed from the reception of the Schedule Set Command.

CC:0053.03.02.11.006 The *Start from now* option MUST cause a schedule to be activated at the specified relative time measured from the reception of the *Schedule Set* Command and run for the specified duration.

CC:0053.03.02.11.007 A node supporting the *Time from now* option MUST support the creation of a schedule with the following start time fields' combination:

Relative = '1' and

YYMMDD = 0xFF, 0x00, **Days**, Weekdays = 0x00, HH:MM = **Hours:Minutes**

CC:0053.03.02.11.008 A receiving node MUST ignore all combinations which do not comply with the one above if the Relative flag is set.

CC:0053.03.02.11.009 A node supporting this option is NOT REQUIRED to support the Start hour and Minute option or Calendar time option even though it MUST be able to read the start day, start hour and start minutes fields when using this option.

2.2.88.3.2 Start Time Mode: Recurring Mode

This mode is used to trigger a schedule repeatedly at fixed intervals. When using this mode, a schedule will start at the specified start time and will restart after a given time has elapsed since the last activation.

The recurring mode is configured with the Recurrence Offset and Recurrence Mode fields of the Schedule Set Command.

CC:0053.03.02.11.00A A node supporting the Recurring Mode MUST support at least one start time option.

2.2.88.4 Schedule Set Command

This command is used to create a new schedule or modify an existing schedule

CC:0053.03.03.11.001 This command MUST enable the schedule if a new schedule is created.

CC:0053.03.03.11.002 This command MUST NOT change the enabled/disabled state of existing schedules.

CC:0053.03.03.11.003 If the receiving node supports S0 or S2 Command Class, it MUST NOT process the schedule creation if any of the scheduled command classes is not supported at the security level of the received Schedule Set Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_SET (0x03)							
Schedule ID							
Schedule ID Block							
Start Year							
Recurrence Offset				Start Month			
Reserved	Recurrence Mode		Start Day of Month				
Reserved	Start Weekday						
Duration Type		Start Hour					
Reserved	Relative	Start Minute					
Duration Byte 1 MSB							
Duration Byte 2 LSB							
Reports to Follow							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Fields not described below remain unchanged from version 2.

Reserved

CC:0053.03.03.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.88.4.1 Recurrence fields

The recurrence settings of a schedule are set by the following fields:

- **Recurrence Mode**
- **Recurrence Offset**

CC:0053.03.03.11.005 These fields MUST be ignored by a node advertising no support for the Recurring Mode start time mode in the Schedule Supported Report Command.

Recurrence mode (2 bits)

This field is used to specify the unit of the Recurrence Offset field.

This field MUST comply with Table 2.496.

Table 2.496: Recurrence Mode encoding

Value	Description	Version
0x00	The recurrence Offset is expressed in Hours.	3
0x01	The recurrence Offset is expressed in Days.	3
0x02	The recurrence Offset is expressed in Weekd	3

CC:0053.03.03.11.007 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0053.03.03.11.008 This field MUST be ignored if the Recurrence Offset field is set to 0.

Recurrence Offset (4 bits)

This field is used to specify the interval at which the recurring schedule will be triggered.

CC:0053.03.03.11.009 This field MUST be encoded according to Table 2.497 and MUST be interpreted together with the Recurrence Mode field.

Table 2.497: Recurrence Offset encoding

Offset	Description
0	Recurrence Disabled
1..15	Repeat every 1..15 hour, day or week. The unit (hour, day, or week) is indicated by the Recurrence Mode field.

CC:0053.03.03.11.00A CC:0053.03.03.11.00B CC:0053.03.03.11.00C When a schedule starts due to its start time options, it MUST restart the current recurrence timer. It means that a schedule MUST start every time the start time condition is met and the recurrence offset MUST trigger the schedule to restart based on the last time it started.

CC:0053.03.03.11.00D For example, a schedule set to start on the 1st of each month with a 2 day recurrence offset MUST start on the 1st, 3rd, 5th, ..., 27th, 29th, 31st and will start again the next month on the 1st, 3rd, etc.

CC:0053.03.03.11.00E A receiving node MUST accept a schedule set to start in the past if recurrence is enabled. In this case, the recurrence MUST trigger the schedule to restart based on the time it should have been started last.

2.2.88.4.2 Start time fields

The Schedule start time is defined by the following fields:

- Start Year
- Start Month
- Start Day of Month
- Start Weekday
- Start Hour
- Start Minute
- Relative

CC:0053.03.03.11.00F A receiving node MUST support the start time fields combinations indicated in sections Section 2.2.86.5, Section 2.2.86.6, Section 2.2.86.7, Section 2.2.86.8 and Section 2.2.88.3.1 if the corresponding start time option is supported.

CC:0053.03.03.11.010 CC:0053.03.03.11.011 If none of the start time fields are specified, the schedule MUST start immediately if the start now option is supported. In this case, the schedule MUST NOT be activated again at a later time unless receiving another Schedule Set Command.

Relative (1 bit)

This field is used to indicate if the schedule start time is using the start from now start option.

CC:0053.03.03.11.012 CC:0053.03.03.11.013 CC:0053.03.03.11.014 The value 1 MUST indicate that the start time fields MUST be interpreted as the time left before the schedule starts. In this case, the start time fields MUST be set and interpreted as indicated in Section 2.2.88.3.1 and the schedule MUST NOT be activated again at a later time unless receiving another Schedule Set Command.

CC:0053.03.03.11.015 The value 0 MUST indicate that the start time fields MUST be interpreted as an absolute start time for the schedule.

2.2.88.5 Schedule Get Command

This command is used to request the configuration for a specific schedule ID.

CC:0053.03.04.11.001 The Schedule Report Command MUST be returned in response to this command.

CC:0053.03.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0053.03.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_GET (0x04)							
Schedule ID							
Schedule ID Block							
AID_RO_CTL	Reserved						

Fields not described below remain unchanged from version 2.

Active_ID / Recurrence Offset control (AID_RO_CTL) (1 bit)

This field is used to request either the Active_ID or the Recurrence Offset value in the Schedule Report Command which is returned in response to this command.

CC:0053.03.04.11.004 A receiving node MUST return a Schedule Report Command advertising the Active_ID of the Schedule if this field is set to 0.

CC:0053.03.04.11.005 A receiving node MUST return a Schedule Report Command advertising the Recurrence Offset of the schedule if this field is set to 1.

2.2.88.6 Schedule Report Command

The Schedule Report Command is used to advertise the configuration for a specific schedule.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_REPORT (0x05)							
Schedule ID Block							
Reserved							
Start Year							
Active_ID / Recurrence Offset (AID_RO)				Start Month			
AID_RO_CTL			Start Day of Month				
Reserved	Start Weekday						
Duration Type		Start Hour					
Reserved	Start Minute						
Duration Byte 1							
Duration Byte 2							
Report to Follow							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Fields not described below remain unchanged from version 2.

Active_ID/Recurrence Offset (AID_RO) (4 bits)

This field is used to advertise the Active_ID or the Recurrence Offset of the advertised schedule.

CC:0053.03.05.11.001 If this field carries the Active_ID value, this field MUST be encoded according to Table 2.494.

CC:0053.03.05.11.002 If this field carries the Recurrence Offset, this field MUST be encoded according to Table 2.497 and MUST be interpreted together with the Recurrence Mode field.

Active_ID / Recurrence Offset control (AID_RO_CTL) (1 bit)

This field is used to indicate whether the Active_ID or the Recurrence Offset value is advertised in the command.

CC:0053.03.05.11.003 The value 0 MUST indicate that the Active_ID/Recurrence Offset (AID_RO) field carried the Active_ID value.

CC:0053.03.05.11.004 The value 1 MUST indicate that the Active_ID/Recurrence Offset (AID_RO) field carried the Recurrence Offset value.

Recurrence mode (2 bits)

This field is used to specify the Recurrence Offset unit.

CC:0053.03.05.11.005 This field MUST be ignored if the AID_RO_CTL field is set to 0.

CC:0053.03.05.11.006 This field MUST comply with Table 2.496 if the AID_RO_CTL field is set to 1.

2.2.89 Schedule Command Class, version 4

The Schedule Command Class, version 4 allows supporting nodes to advertise the list of commands they support for scheduling.

2.2.89.1 Compatibility Considerations

The Schedule Command Class version 4 introduces the possibility to advertise the list of command supported for scheduling and explicitly allows scheduling extended command classes. The following commands are introduced:

- Schedule Supported Commands Get Command
- Schedule Supported Commands Report Command

The following functionalities become obsoleted:

- Redundant values in the Active_ID field encoding
- The Report to Follow field.

A supporting node no longer needs to be able to parse and handle several commands in order to set or report schedules or schedules states. Supporting nodes MUST use the Transport Service Command Class, version 2 in order to transmit long Z-Wave payloads.

CC:0053.04.00.21.001 A node supporting the Schedule Command Class, version 4 MUST also support version 3. Commands and fields not mentioned in this version MUST remain unchanged from version 3.

2.2.89.2 Schedule Supported Report Command

This command is used to advertise the scheduling properties of a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_REPORT (0x02)							
Number of Supported Schedule IDs							
Support Enable/Disable	Fallback Support	Start Time Support					
Number of Supported CC							
Supported CC 1							
Reserved						Supported Command 1	
...							
Supported CC N							
Reserved						Supported Command N	
Override Support	Supported Override Types						
Schedule ID Block							
Number of Supported Schedule Blocks							

All fields not described below remain unchanged from version 3.

Supported Command (N * 2 bits)

CC:0053.04.02.11.001 This field MUST advertise the supported commands for the command class entry. This field MUST comply with Table 2.498.

Table 2.498: Schedule Supported Report version 4::Supported Command

Value	Description	Version
0x00	All Commands within the Command Class are supported.	1
0x01	Only Set Commands are supported.	1
0x02	Only Get Commands are supported.	1
0x03	Custom list of commands are supported.	4

CC:0053.04.02.11.002 Certain command classes commands do not contain the string “Set” or “Get” in their name. Nodes MUST consider all commands mandating to return a response to be of type “Get” and all other commands to be of type “Set”.

CC:0053.04.02.11.003 The length of this field in bytes MUST be according to the Number of supported CC field value.

CC:0053.04.02.11.004 A supporting node MUST NOT advertise the 0x03 value if the list of supported command can be advertised with the other values.

CC:0053.04.02.52.001 If a sending node advertises 0x03 for a given Schedule ID Block and Command Class, a controlling node SHOULD retrieve the exact list of supported commands using the Schedule Supported Commands Get Command.

2.2.89.3 Schedule Supported Commands Get Command

This command is used to query the list of commands that can be scheduled within for a given command class and Schedule ID Block.

CC:0053.04.0A.11.001 The Schedule Supported Commands Report Command MUST be returned in response to this command.

CC:0053.04.0A.51.001 This command MUST NOT be issued via multicast addressing.

CC:0053.04.0A.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_COMMANDS_GET (0x0A)							
Schedule ID Block							

Schedule ID Block (8 bits)

This field is used to request a particular Schedule ID Block.

CC:0053.04.0A.12.001 If a non-supported Schedule ID Block is specified, a receiving node SHOULD return a response for the Schedule ID block 1.

2.2.89.4 Schedule Supported Commands Report Command

This command is used to advertise the list of commands that can be scheduled within a given Schedule ID Block

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_SUPPORTED_COMMANDS_REPORT (0x0B)							
Schedule ID Block							
Command Class List Length							
Command Class 1 (1 or 2 bytes)							
Supported Command List Length 1							
Supported Command 1 - 1							
...							
Supported Command 1 - K							
...							
Command Class N (1 or 2 bytes)							
Supported Command List Length N							
Supported Command N - 1							
...							
Supported Command N - K							

Schedule ID Block (8 bits)

This field is used to specify the requested Schedule ID Block.

Command Class List Length (8 bit)

This field is used to advertise the number of command classes contained in the command.

CC:0053.04.0B.11.001 This field MUST be set to the number of Command Class entries present in this command (represented by N in the command structure).

Command Class (8 or 16 bits)

CC:0053.04.0B.13.001 This field is used to indicate the command class to which the advertised commands belong. This field MAY carry extended Command Classes

Supported Command List Length (8 bits)

This field is used to advertise how many commands are advertised in the current Command Class' list.

CC:0053.04.0B.11.002 This field MUST be set to the number of Supported Commands present in the current Command Class entry (represented by K in the command structure).

Supported Command (K bytes)

This field is used to advertise the list of commands for the given Command Class entry that can be scheduled for the actual Schedule ID Block.

CC:0053.04.0B.11.003 The length of this field in bytes MUST be according to the corresponding Supported Command List Length field value.

2.2.89.5 Schedule Set Command

This command is used to create a new schedule or modify an existing schedule.

CC:0053.04.03.11.001 This command MUST enable the schedule if a new schedule is created.

CC:0053.04.03.11.002 This command MUST NOT change the enabled/disabled state of existing schedules

CC:0053.04.03.11.003 If the receiving node supports S0 or S2 Command Class, it MUST NOT process the schedule creation if any of the scheduled command classes is not supported at the security level of the received Schedule Set Command.

CC:0053.04.03.11.004 A controlling node using previous versions may try to schedule unsupported commands. In this case, a receiving node MUST ignore the Schedule Set Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_SET (0x03)							
Schedule ID							
Schedule ID Block							
Start Year							
Recurrence Offset				Start Month			
Reserved	Recurrence Mode		Start Day of Month				
Reserved	Start Weekday						
Duration Type		Start Hour					
Reserved	Relative	Start Minute					
Duration Byte 1 MSB							
Duration Byte 2 LSB							
Reports to Follow [OBSOLETED]							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Fields not described below remain unchanged from version 3.

Reports to Follow (8 bits) [OBSOLETED]

CC:0053.04.03.11.005 This field has been OBSOLETED: a sending node MUST use the Transport Service Command Class in order to set a schedule containing commands payload too large to fit in the Z-Wave MAC frame size.

CC:0053.04.03.11.006 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.89.6 Schedule Report Command

The Schedule Report Command is used to advertise the configuration for a specific schedule.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = COMMAND_SCHEDULE_REPORT (0x05)							
Schedule ID Block							
Reserved							
Start Year							
Active_ID / Recurrence Offset (AID_RO)				Start Month			
AID_RO_CTL				Start Day of Month			
Reserved			Start Weekday				
Duration Type				Start Hour			
Reserved		Start Minute					
Duration Byte 1							
Duration Byte 2							
Report to Follow [OBSOLETED]							
Number of Cmd to Follow							
Cmd Length 1							
Cmd Byte 1 - 1							
...							
Cmd Byte 1 - N							
...							
Cmd Length P							
Cmd Byte P - 1							
...							
Cmd Byte P - N							

Fields not described below remain unchanged from version 3.

Active_ID/Recurrence Offset (4 bits)

This field is used to advertise the Active_ID or the Recurrence Offset of the advertised schedule.

CC:0053.04.05.11.001 If this field carries the Active_ID value, this field MUST be encoded according to Table 2.499.

CC:0053.04.05.11.002 If this field carries the Recurrence Offset, this field MUST be encoded according to Table 2.497 and MUST be interpreted together with the Recurrence Mode field.

Reports to Follow (8 bits) [OBSOLETED]

CC:0053.04.05.11.003 This field has been OBSOLETED: a sending node MUST use the Transport Service Command Class in order to report a schedule containing commands payload too large to fit in the Z-Wave MAC frame size.

CC:0053.04.05.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.89.7 Schedule State Report Command

This command is used to advertise the status of all schedules supported by the specified schedule block.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE (0x53)							
Command = SCHEDULE_STATE_REPORT (0x09)							
Number of Supported Schedule ID							
Reports to Follow [OBSOLETED]							Override
Active_ID 2				Active_ID 1			
...							
Active_ID N				Active_ID N-1			
Schedule ID Block							

Fields not described below remain unchanged from version 3.

Reports to Follow (7 bits) [OBSOLETED]

CC:0053.04.09.11.001 This field has been OBSOLETED: a sending node MUST use the Transport Service Command Class in order to advertise an Active_ID field too large to fit in the Z-Wave MAC frame size.

CC:0053.04.09.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Active_ID (N * 4 bits)

CC:0053.04.09.11.003 The *Active_ID* field MUST be used to advertise the status for each supported schedule ID. Four bits are used for each Schedule ID to indicate the status. This means that :

- Bits 0 to 3 in Active_ID byte 1 represent Schedule ID = 1,.
- Bits 4 to 7 in Active_ID byte 1 represent Schedule ID = 2
- Bits 0 to 3 in Active_ID byte 2 represent Schedule ID = 3
- …

CC:0053.04.09.11.004 The size of this field MUST be the smallest number of bytes needed to advertise the number of supported Schedule IDs advertised in this command.

CC:0053.04.09.11.005 Each 4 bits unit MUST be encoded according to Table 2.499.

Table 2.499: Active_ID encoding, version 4

Hex	Description	Detailed description	Version
0x00	Not used	The Schedule ID is not used (not set/configured) or unsupported	1
0x01	[OBSOLETED] Override + Not used	[OBSOLETED] A sending node MUST NOT use this status and MUST use 0x00 instead (and the override field set to 1 for the Schedule State Report Command).	4
0x02	Not Active	The Schedule ID is used, enabled and currently not active.	1
0x03	Active	The Schedule ID is used, enabled and currently active.	1
0x05	Override + Active	The Schedule ID is used, enabled and should currently be active but it is suspended by the Override Schedule The override field MUST be set to 1 when using this status in the Schedule State Report Command	1
0x06	[OBSOLETED] Override + Not Active	[OBSOLETED] A sending node MUST NOT use this status and MUST use 0x02 instead. (and the override field set to 1 for the Schedule State Report Command).	4
0x07	[OBSOLETED] Override + Disabled	[OBSOLETED] A sending node MUST NOT use this status and MUST use 0x04 instead. (and the override field set to 1 for the Schedule State Report Command).	4

CC:0053.04.09.11.006 CC:0053.04.09.11.007 CC:0053.04.09.11.008 CC:0053.04.09.11.009

CC:0053.04.09.11.00A All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.90 Schedule Entry Lock Command Class, version 1 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Schedule Command Class.

If implementing this command class, it is RECOMMENDED that the Schedule Command Class is also implemented.

The Schedule Entry Lock Command Class provides Z-Wave devices the capability to exchange scheduling information. The Schedule Entry Lock Type Commands are for controlling the schedules of an Entry Lock using schedule based user code Ids. The Entry Lock supports two types of schedules for each user ID supported in the device. The two schedule types are a time-fenced weekly schedule and a time-fenced one-time range schedule. When these schedules are configured and enabled, it allows the specified user ID's code to be active during the time intervals configured in the scheduling slots.

The Week Day schedule is a day-to-day schedule that will repeat weekly for the enabled user ID. A single schedule slot cannot span days.

Example: A homeowner has a Secure Keypad Door Lock and a dog that needs walking three times a week. The dog walker can be given access to the house using this schedule. The homeowner would give the dog walker a keypad code that would be active M, W, F from 1pm - 2pm.

The Year Day schedule is an extended schedule that allows two points in time to be specified that is beyond a daily schedule. A particular slot can span weeks, months or years. Once the end point is reached that schedule slot is no longer valid because it is out of range.

Example: A homeowner is going away on vacation for two weeks. The homeowner could give the neighbor a keypad code to the neighbor that would be active from April 2nd, 2008 to April 16th' 2008. The code would be invalid after April 16th 2008.

2.2.90.1 Schedule Entry Lock Enable Set Command

This command enables or disables schedules for a specified user code ID. It affects only the schedules associated with the specific user ID.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_ENABLE_SET (0x01)							
User Identifier							
Enabled							

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored.

Enabled (8 bits)

Table 2.500: Schedule Entry Lock Enable Set::Enabled encoding

Value	Description
0x00	Schedule for the user identified is disabled.
0x01	Schedule for the user identified is enabled.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.90.2 Schedule Entry Lock Enable All Set Command

This command enables or disables all schedules for type Entry Lock.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_ENABLE_ALL_SET (0x02)							
Enabled							

Enabled (8 bits)

See description in Schedule Entry Lock Enable Set Command.

2.2.90.3 Schedule Entry Lock Supported Get Command

This command is used to request the number of schedule slots each type of schedule the device supports for every user.

The Schedule Entry Lock Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_TYPE_SUPPORTED_GET (0x09)							

2.2.90.4 Schedule Entry Lock Supported Report Command

This command is used to report the number of supported schedule slots an Entry Lock schedule device supports for each user in the system. It lists how many schedule slots there are for Week Days type and how many slots for the Year Day type.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_TYPE_SUPPORTED_REPORT (0x0A)							
Number of Slots Week Day							
Number of slots Year Day							

Number of Slots Week Day (8 bits)

A number from 0 - 255 that represents how many different schedule slots are supported each week for every user in the system for type Week Day.

Number of Slots Year Day (8 bits)

A number from 0 - 255 that represents how many different schedule slots are supported for every user in the system for type Year Day.

2.2.90.5 Schedule Entry Lock Week Day Schedule Set Command

This command set or erase a weekday schedule for a identified user who already has valid user access code.

When setting, the week day schedule is automatically enabled and the identified user if it is not already. The start parameters of the time fence needs to occur prior to the stop parameters. When erasing the schedule slot ID, the user code ID will continue to use week day type scheduling.

Note: Each user can only use one type of scheduling at a time.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_WEEK_DAY_SET (0x03)							
Set Action							
User Identifier							
Schedule Slot ID							
Day of Week							
Start Hour							
Start Minute							
Stop Hour							
Stop Minute							

Set Action (8 bits)

Table 2.501: Schedule Entry Lock Week Day Schedule Set::Set Action encoding

Set Action	Description
0	Erase the schedule slot.
1	Modify the schedule slot for the identified user.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored.

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Week Day Supported*.

Day of Week (8 bits)

A value from 0 to 6 where 0 is Sunday.

Start Hour (8 bits)

A value from 0 to 23 representing the starting hour of the time fence.

Start Minute (8 bits)

A value from 0 to 59 representing the starting minute of the time fence.

Stop Hour (8 bits)

A value from 0 to 23 representing the stop hour of the time fence.

Stop Minute (8 bits)

A value from 0 to 59 representing the stop minute of the time fence

2.2.90.6 Schedule Entry Lock Week Days Schedule Get Command

This command gets a week day schedule slot for a identified user and specified schedule slot ID.

The Schedule Entry Lock Week Days Schedule Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_WEEK_DAY_GET (0x04)							
User Identifier							
Schedule Slot ID							

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored.

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Week Day Supported*.

2.2.90.7 Schedule Entry Lock Week Day Schedule Report Command

This command returns week day schedule report for the requested schedule slot ID for identified user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_WEEK_DAY_REPORT (0x05)							
User Identifier							
Schedule Slot ID							
Day of Week							
Start Hour							
Start Minute							
Stop Hour							
Stop Minute							

Refer to the description under the Schedule Set Week Day Schedule.

Note: If a requested schedule slot is erased/empty, the time fields SHOULD be set to 0xFF.

2.2.90.8 Schedule Entry Lock Year Day Schedule Set Command

This command sets or erases a schedule slot for a identified user who already has valid user access code. The year day schedule represents two days, any time apart, where the specified user ID's code is valid. When setting the schedule slot, the start parameters of the time fence needs to occur prior to the stop parameters and the year day schedule is automatically enabled for the identified user. When erasing, the user code does not change from year day scheduling.

Note: Each user can only use one type of scheduling at a time.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_YEAR_DAY_SET (0x06)							
Set Action							
User Identifier							
Schedule Slot ID							
Start Year							
Start Month							
Start Day							
Start Hour							
Start Minute							
Stop Year							
Stop Month							
Stop Day							
Stop Hour							
Stop Minute							

Set Action (8 bits)

Table 2.502: Schedule Entry Lock Year Day Schedule Set::Set Action Encoding

Value	Description
0x00	Erase the schedule slot.
0x01	Modify the schedule slot for the identified user.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Year Day Supported*.

Start Year (8 bits)

A value from 0 to 99 that represents the 2 year in the century.

Start Month (8 bits)

A value from 1 to 12 that represents the month in a year.

Start Day (8 bits)

A value from 1 to 31 that represents the date of the month.

Start Hour (8 bits)

A value from 0 to 23 representing the starting hour of the time fence.

Start Minute (8 bits)

A value from 0 to 59 representing the starting minute of the time fence.

Stop Year (8 bits)

A value from 0 to 99 that represents the 2 year in the century.

Stop Month (8 bits)

A value from 1 to 12 that represents the month in a year.

Stop Day (8 bits)

A value from 1 to 31 that represents the date of the month.

Stop Hour (8 bits)

A value from 0 to 23 representing the stop hour of the time fence.

Stop Minute (8 bits)

A value from 0 to 59 representing the stop minute of the time fence

2.2.90.9 Schedule Entry Lock Year Day Schedule Get Command

This command gets a year/day schedule slot for an identified user and specified schedule slot ID.

The Schedule Entry Lock Year Day Schedule Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_YEAR_DAY_GET (0x07)							
User Identifier							
Schedule Slot ID							

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Year Day Supported*.

2.2.90.10 Schedule Entry Lock Year Day Schedule Report Command

This command returns year/day schedule report for the requested schedule slot ID for the identified user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_YEAR_DAY_REPORT (0x08)							
User Identifier							
Schedule Slot ID							
Start Year							
Start Month							
Start Day							
Start Hour							
Start Minute							
Stop Year							
Stop Month							
Stop Day							
Stop Hour							
Stop Minute							

Refer to the description under Schedule Set Year Day Schedule command.

Note: If a requested schedule slot is erased/empty the time fields SHOULD be set to 0xFF.

2.2.91 Schedule Entry Lock Command Class, version 2 [DEPRECATED]

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Schedule Command Class.

If implementing this command class, it is RECOMMENDED that the Schedule Command Class is also implemented.

The Schedule Entry Lock Command Class provides Z-Wave devices the capability to exchange scheduling information. The Schedule Entry Lock Type Commands are for controlling the schedules of an Entry Lock using schedule based user code Ids. The Entry Lock supports two types of schedules for each user ID supported in the device. The two schedule types are a time-fenced weekly schedule and a time-fenced one-time range schedule. When these schedules are configured and enabled, it allows the specified user ID's code to be active during the time intervals configured in the scheduling slots.

In Version 2 local time is used instead of UTC time, and Time Offset commands are added.

The commands not mentioned here remain the same as in version 1.

2.2.91.1 Schedule Entry Lock Time Offset Get Command

This command is used to request time zone offset and daylight savings parameters.

The Schedule Entry Lock Time Offset Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_TIME_OFFSET_GET (0x0B)							

2.2.91.2 Schedule Entry Lock Time Offset Set Command

This command is used to set the current local TZO and DST offsets into an Entry Lock Device. Any schedules that are already in the device before or after issuing the Schedule Entry Time Offset Set command are now assumed to be programmed in the Local time set by this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_TIME_OFFSET_SET (0x0D)							
Sign TZO	Hour TZO						
Minute TZO							
Sign Offset TZO	Minute Offset DST						

Sign TZO (1 bit)

Plus (0) or minus (1) sign to indicate a positive or negative offset from UTC.

Hour TZO (7 bits)

Specify the number of hours that the originating time zone deviates from UTC. Refer to the DST field regarding daylight savings handling.

Minute TZO (7 bits)

Specify the number of minutes that the originating time zone deviates UTC. Refer to the DST field regarding daylight savings handling.

Sign Offset DST (1 bit)

Plus (0) or minus (1) sign to indicate a positive or negative offset from UTC.

Minute Offset DST (7 bits)

This field MUST specify the number of minutes the time is to be adjusted when daylight savings mode is enabled.

2.2.91.3 Schedule Entry Lock Time Offset Report Command

This command is used to advertise the time zone offset and daylight savings parameters.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_TIME_OFFSET_REPORT (0x0C)							
Sign TZO	Hour TZO						
Minute TZO							
Sign Offset TZO	Minute Offset DST						

Refer to description under the Schedule Entry Lock Time Offset Set command.

2.2.92 Schedule Entry Lock Command Class, version 3

Warning: THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the *Schedule Entry Lock Command Class, version 4 [NEVER CERTIFIED]*.

The Schedule Entry Lock Command Class provides a scheduling type alongside the existing types Week Day and Year Day. The new type is similar to Week Day functionality but provides a simpler implementation to repeat a time slot daily (selected days) and repeat those days weekly. The commands not mentioned here remain the same as in Version 2.

2.2.92.1 Schedule Entry Type Supported Report Command

This command is used to report the number of supported schedule slots an Entry Lock schedule device supports for each user in the system. It lists how many schedule slots there are for Week Day, Year Day, and Daily Repeating types.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_TYPE_SUPPORTED_REPORT (0x0A)							
Number of Slots Week Day							
Number of Slots Year Day							
Number of Slots Daily Repeating							

Number of Slots Week Day (8 bits)

A number from 0 to 255 that represents how many different schedule slots are supported each week for every user in the system for type Week Day.

Number of Slots Year Day (8 bits)

A number from 0 to 255 that represents how many different schedule slots are supported for every user in the system for type Year Day.

Number of Slots Daily Repeating (8 bits)

A number from 0 to 255 that represents how many different schedule slots are supported for every user in the system for type Daily Repeating Day.

2.2.92.2 Schedule Entry Lock Daily Repeating Set Command

This command is used to set or erase a daily repeating schedule for an identified user who already has valid user access code.

When setting; the daily repeating schedule is automatically enabled for the identified user if it is not already. The start parameters of the time fence needs to occur prior to the stop parameters. When erasing the schedule slot ID, the user code ID will continue to use daily repeating type scheduling.

Note: Each user can only use one type of scheduling at a time.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_DAILY_REPEATING_SET (0x10)							
Set Action							
User Identifier							
Schedule Slot ID							
Week Day Bitmask							
Start Hour							
Start Minute							
Duration Hour							
Duration Minute							

Set Action (8 bits)

Table 2.503: Schedule Entry Lock Daily Repeating Set::Set Action encoding

Value	Description
0x00	Erase the schedule slot.
0x01	Modify the schedule slot for the identified user.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to Number of Slots Daily Repeating Supported.

Week Day Bitmask (8 bits)

A bitmask of the days of the week for this schedule entry is active.

Table 2.504: Schedule Entry Lock Daily Repeating Set::Week Day Bitmask encoding

Bit	7	6	5	4	3	2	1	0
Value	Res	Sat	Fri	Thr	Wed	Tue	Mon	Sun

The 'Res' bit is reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Start Hour (8 bits)

A value from 0 to 23 representing the starting hour of the time fence.

Start Minute (8 bits)

A value from 0 to 59 representing the starting minute of the time fence.

Duration Hour (8 bits)

A value from 0 to 23 representing how many hours the time fence will last. Duration hour will be maxed at the documented capability of the specific device since this scheduling type is memory conscious.

Duration Minute (8 bits)

A value from 0 to 59 representing how many minutes the time fence will last past the Duration Hour field.

2.2.92.3 Schedule Entry Lock Daily Repeating Get Command

This command is used to request a daily repeating schedule slot for a identified user and specified schedule slot ID.

The Schedule Entry Lock Daily Repeating Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_DAILY_REPEATING_GET (0x0E)							
User Identifier							
Schedule Set ID							

User Identifier (8 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Daily Repeating Supported*.

2.2.92.4 Schedule Entry Lock Daily Repeating Report Command

This command is used to return the requested schedule slot ID for identified user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = SCHEDULE_ENTRY_LOCK_DAILY_REPEATING_REPORT (0x0F)							
User Identifier							
Schedule Set ID							
Week Day Bitmask							
Start Hour							
Start Minute							
Duration Hour							
Duration Minute							

Refer to the description under the Schedule Set Daily Repeating Schedule.

2.2.93 Schedule Entry Lock Command Class, version 4 [NEVER CERTIFIED]

Warning: THIS COMMAND CLASS VERSION HAS NEVER BEEN CERTIFIED

This command class version has never been implemented and certified by a Z-Wave product. Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.

Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

The Schedule Entry Lock Command Class provides a scheduling type alongside the existing types Week Day and Year Day. The new type is similar to Week Day functionality but provides a simpler implementation to repeat a time slot daily (selected days) and repeat those days weekly. The commands not mentioned here remain the same as in Version 3.

The Schedule Entry Command Class, version 4 is backwards compatible with *Schedule Entry Lock Command Class, version 3*.

The *Extended User Code Set Command* permits to configure an extended number of users (credentials) with 16 bits for its "User Identifier". Also the *User Credential Command Class, version 1 [NEVER CERTIFIED]* uses 16 bits for its "User Unique Identifier".

In this Schedule Entry Command Class, version 4 the following commands have been introduced to match the extended number of supported users:

- Extended Schedule Entry Lock Enable Set Command
- Extended Schedule Entry Lock Week Day Schedule Set Command
- Extended Schedule Entry Lock Week Day Schedule Get Command
- Extended Schedule Entry Lock Week Day Schedule Report Command
- Extended Schedule Entry Lock Year Day Schedule Set Command
- Extended Schedule Entry Lock Year Day Schedule Get Command
- Extended Schedule Entry Lock Year Day Schedule Report Command
- Extended Schedule Entry Lock Daily Repeating Set Command
- Extended Schedule Entry Lock Daily Repeating Get Command
- Extended Schedule Entry Lock Daily Repeating Report Command

2.2.93.1 Extended Schedule Entry Lock Enable Set Command

This command enables or disables schedules for a specified User ID. It only affects the schedules associated with the specific User ID.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_ENABLE_SET (0x11)							
User Identifier (MSB)							
User Identifier (LSB)							
Reserved							Enabled

User Identifier (16 bits)

The User Identifier (User ID) is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command MUST be ignored.

Enabled (1 bit)

Table 2.505: Schedule Entry Lock Enable Set::Enabled encoding

Value	Description
0	Schedule for the user identified is disabled.
1	Schedule for the user identified is enabled.

2.2.93.2 Extended Schedule Entry Lock Week Day Schedule Set Command

This command set or erase a weekday schedule for a identified user who already has valid user access code.

When setting, the week day schedule is automatically enabled and the identified user if it is not already. The start parameters of the time fence needs to occur prior to the stop parameters. When erasing the schedule slot ID, the user code ID will continue to use week day type scheduling.

Note: Each user can only use one type of scheduling at a time.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_WEEK_DAY_SCHEDULE_SET (0x12)							
Set Action							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							
Day of Week							
Start Hour							
Start Minute							
Stop Hour							
Stop Minute							

Set Action (8 bits)

Table 2.506: Schedule Entry Lock Week Day Schedule Set::Set Action encoding

Set Action	Description
0	Erase the schedule slot.
1	Modify the schedule slot for the identified user.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Identifier (16 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored.

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Week Day Supported*.

Day of Week (8 bits)

A value from 0 to 6 where 0 is Sunday.

Start Hour (8 bits)

A value from 0 to 23 representing the starting hour of the time fence.

Start Minute (8 bits)

A value from 0 to 59 representing the starting minute of the time fence.

Stop Hour (8 bits)

A value from 0 to 23 representing the stop hour of the time fence.

Stop Minute (8 bits)

A value from 0 to 59 representing the stop minute of the time fence

2.2.93.3 Extended Schedule Entry Lock Week Day Schedule Get Command

This command gets a week day schedule slot for a identified user and specified schedule slot ID.

The Extended Schedule Entry Lock Week Days Schedule Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_WEEK_DAY_SCHEDULE_GET (0x13)							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							

User Identifier (16 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored.

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Week Day Supported*.

2.2.93.4 Extended Schedule Entry Lock Week Day Schedule Report Command

This command returns week day schedule report for the requested schedule slot ID for identified user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_WEEK_DAY_SCHEDULE_REPORT (0x14)							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							
Day of Week							
Start Hour							
Start Minute							
Stop Hour							
Stop Minute							

Refer to the description under the Schedule Set Week Day Schedule.

Note: If a requested schedule slot is erased/empty, the time fields SHOULD be set to 0xFF.

2.2.93.5 Extended Schedule Entry Lock Year Day Schedule Set Command

This command sets or erases a schedule slot for a identified user who already has valid user access code. The year day schedule represents two days, any time apart, where the specified user ID's code is valid. When setting the schedule slot, the start parameters of the time fence needs to occur prior to the stop parameters and the year day schedule is automatically enabled for the identified user. When erasing, the user code does not change from year day scheduling.

Note: Each user can only use one type of scheduling at a time.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_YEAR_DAY_SCHEDULE_SET (0x15)							
Set Action							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							
Start Year							
Start Month							
Start Day							
Start Hour							
Start Minute							
Stop Year							
Stop Month							
Stop Day							
Stop Hour							
Stop Minute							

Set Action (8 bits)

Table 2.507: Schedule Entry Lock Year Day Schedule Set::Set Action Encoding

Value	Description
0x00	Erase the schedule slot.
0x01	Modify the schedule slot for the identified user.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Identifier (16 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Year Day Supported*.

Start Year (8 bits)

A value from 0 to 99 that represents the 2 year in the century.

Start Month (8 bits)

A value from 1 to 12 that represents the month in a year.

Start Day (8 bits)

A value from 1 to 31 that represents the date of the month.

Start Hour (8 bits)

A value from 0 to 23 representing the starting hour of the time fence.

Start Minute (8 bits)

A value from 0 to 59 representing the starting minute of the time fence.

Stop Year (8 bits)

A value from 0 to 99 that represents the 2 year in the century.

Stop Month (8 bits)

A value from 1 to 12 that represents the month in a year.

Stop Day (8 bits)

A value from 1 to 31 that represents the date of the month.

Stop Hour (8 bits)

A value from 0 to 23 representing the stop hour of the time fence.

Stop Minute (8 bits)

A value from 0 to 59 representing the stop minute of the time fence

2.2.93.6 Extended Schedule Entry Lock Year Day Schedule Get Command

This command gets a year/day schedule slot for an identified user and specified schedule slot ID.

The Extended Schedule Entry Lock Year Day Schedule Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_YEAR_DAY_SCHEDULE_GET (0x16)							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							

User Identifier (16 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Year Day Supported*.

2.2.93.7 Extended Schedule Entry Lock Year Day Schedule Report Command

This command returns year/day schedule report for the requested schedule slot ID for the identified user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_YEAR_DAY_SCHEDULE_REPORT (0x17)							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							
Start Year							
Start Month							
Start Day							
Start Hour							
Start Minute							
Stop Year							
Stop Month							
Stop Day							
Stop Hour							
Stop Minute							

Refer to the description under Schedule Set Year Day Schedule command.

Note: If a requested schedule slot is erased/empty the time fields SHOULD be set to 0xFF.

2.2.93.8 Extended Schedule Entry Lock Daily Repeating Set Command

This command is used to set or erase a daily repeating schedule for an identified user who already has valid user access code.

When setting; the daily repeating schedule is automatically enabled for the identified user if it is not already. The start parameters of the time fence needs to occur prior to the stop parameters. When erasing the schedule slot ID, the user code ID will continue to use daily repeating type scheduling.

Note: Each user can only use one type of scheduling at a time.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_DAILY_REPEATING_SET (0x18)							
Set Action							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							
Week Day Bitmask							
Start Hour							
Start Minute							
Duration Hour							
Duration Minute							

Set Action (8 bits)

Table 2.508: Schedule Entry Lock Daily Repeating Set::Set Action encoding

Value	Description
0x00	Erase the schedule slot.
0x01	Modify the schedule slot for the identified user.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Identifier (16 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to Number of Slots Daily Repeating Supported.

Week Day Bitmask (8 bits)

A bitmask of the days of the week for this schedule entry is active.

Table 2.509: Schedule Entry Lock Daily Repeating Set::Week Day Bitmask encoding

Bit	7	6	5	4	3	2	1	0
Value	Res	Sat	Fri	Thr	Wed	Tue	Mon	Sun

The 'Res' bit is reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Start Hour (8 bits)

A value from 0 to 23 representing the starting hour of the time fence.

Start Minute (8 bits)

A value from 0 to 59 representing the starting minute of the time fence.

Duration Hour (8 bits)

A value from 0 to 23 representing how many hours the time fence will last. Duration hour will be maxed at the documented capability of the specific device since this scheduling type is memory conscious.

Duration Minute (8 bits)

A value from 0 to 59 representing how many minutes the time fence will last past the Duration Hour field.

2.2.93.9 Extended Schedule Entry Lock Daily Repeating Get Command

This command is used to request a daily repeating schedule slot for a identified user and specified schedule slot ID.

The Extended Schedule Entry Lock Daily Repeating Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_DAILY_REPEATING_GET (0x19)							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							

User Identifier (16 bits)

The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. If the user identifier is out of range, the command will be ignored

Schedule Slot ID (8 bits)

A value from 1 to *Number of Slots Daily Repeating Supported*.

2.2.93.10 Extended Schedule Entry Lock Daily Repeating Report Command

This command is used to return the requested schedule slot ID for identified user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (0x4E)							
Command = EXTENDED_SCHEDULE_ENTRY_LOCK_DAILY_REPEATING_REPORT (0x1A)							
User Identifier (MSB)							
User Identifier (LSB)							
Schedule Slot ID							
Week Day Bitmask							
Start Hour							
Start Minute							
Duration Hour							
Duration Minute							

Refer to the description under the Schedule Set Daily Repeating Schedule.

2.2.94 Screen Attributes Command Class, version 1

The Screen Attribute Command Class is used to retrieve screen attributes from the device hosting the screen. This allows another device to send data formatted according to the screen attributes to the device hosting the screen. The screen may be located on any device in the network.

2.2.94.1 Screen Attributes Get Command

This command is used to request the screen attributes.

The Screen Attributes Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCREEN_ATTRIBUTES (0x93)							
Command = SCREEN_ATTRIBUTES_GET (0x01)							

2.2.94.2 Screen Attributes Report Command

This command is used to advertise the screen attributes.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCREEN_ATTRIBUTES (0x93)							
Command = SCREEN_ATTRIBUTES_REPORT (0x02)							
Reserved			Number of Lines				
Characters per Line							
Line Buffer Size							
Character Encoding							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Lines (5 bits)

Number of lines the screen supports (1..16).

Characters per Line (8 bits)

Number of characters the screen supports on each line (1..255).

Line Buffer Size (8 bits)

Number of characters the line buffer supports for each line (1..255). Size of line buffer will always be equal or larger than the number of visual characters per line. The text will typically scroll in case it is larger than the number of visual characters.

Character Encoding (8 bits)

The screen supports the following numerical representations of a character:

Table 2.510: Screen Attributes Report::Character Encoding encoding

Bit Map	Description
Bit 0	Supports ASCII codes if the bit is 1 and the opposite if 0. See *ASCII Codes* (values 128-255 are ignored)
Bit 1	Supports ASCII codes and Extended ASCII codes if the bit is 1 and the opposite if 0. See *ASCII Codes*.
Bit 2	Supports Unicode UTF-16 if the bit is 1 and the opposite if 0.
Bit 3	Supports ASCII codes and Player codes, see *ASCII Codes* (undefined values are ignored)

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

2.2.95 Screen Attributes Command Class, version 2

The Screen Attribute Command Class, version 2 introduces the Screen Timeout of the Screen Attributes Command.

Details not mentioned remain the same as in version 1.

2.2.95.1 Screen Attributes Report Command

This command is used to advertise the screen attributes.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCREEN_ATTRIBUTES (0x93)							
Command = SCREEN_ATTRIBUTES_REPORT (0x02)							
Reserved		Escape Sequence	Number of Lines				
Characters per Line							
Line Buffer Size							
Character Encoding							
Screen Timeout							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Escape Sequence (1 bit)

If set to 0, escape sequences are not supported by the device.

If set to 1, escape sequences are supported by the device.

Number of Lines (5 bits)

Number of lines the screen supports (1..16).

Characters per Line (8 bits)

Number of characters the screen supports on each line (1..255).

Line Buffer Size (8 bits)

Number of characters the line buffer supports for each line (1..255). Size of line buffer will always be equal or larger than the number of visual characters per line. The text will typically scroll in case it is larger than the number of visual characters.

Character Encoding (8 bits)

This field MUST be encoded according to Table 2.510.

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Screen Timeout (8 bits)

If Screen Timeout is set to 0, the display is always on. A value larger than 0 MUST specify the display timeout in seconds.

2.2.96 Screen Meta Data Command Class, version 1

The Screen Meta Data Command Class is used to streaming data containing user related information to a screen located on a device in a Z-Wave network. The screen can request single or multiple data packets. The device having the data containing user related information to the screen can also initiate the data streaming.

In order not to congest the Z-Wave network, large data transfers MUST leave transmit opportunities for other nodes in the network. If sending a command longer than two frames, a node MUST implement a delay between every transmitted frame. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 2 frames back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

2.2.96.1 Screen Meta Data Get Command

This command is used to request the Screen Meta Data Report Command. The Screen Meta Data Get Command is used as handshake to avoid buffer overflow in the receiving node. The Screen Meta Data Get Command will optionally be able to request multiple Screen Meta Data Report Commands to improve the effective bandwidth.

The Screen Meta Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCREEN_MD (0x92)							
Command = SCREEN_MD_GET (0x01)							
Number of Reports							
NodeID							

Number of Reports (8 bits)

Number of Screen Meta Data Report Commands to be received without requesting each Screen Meta Data Report Command (1..255). Be aware of overflow when requesting multiple reports.

NodeID (8 bits)

The NodeID (1..232) specifies the device to receive the requested reports. In case NodeID is equal to 0x00, the information is requested by the source NodeID of the Screen Meta Data Get Command.

2.2.96.2 Screen Meta Data Report Command

This command is used to send data to the device hosting the screen.

The size of the payload SHOULD NOT be bigger than 48 bytes. It is possible to write characters to multiple lines in the same frame.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCREEN_MD (0x92)							
Command SCREEN_MD_REPORT							
More Data	Reserved	Screen Settings			Character Encoding		
Line Settings A			Clear A	Line Number A			
Character Position A							
Number of Characters A							
Character 1,A							
...							
Character N,A							
...							
Line Settings B			Clear B	Line Number B			
Character Position B							
Number of Characters B							
Character 1,B							
...							
Character N,B							

More Data (1 bit)

The more data bit indicates if additional reports are expected before the whole data streaming is completed. If the more data bit is set to 1 then additional reports are expected and the opposite if 0.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Screen Settings (3 bits)

This field MUST comply with Table 2.511:

Table 2.511: Screen Meta Data Report::Screen Settings encoding

Screen Settings	Description
0	Whole screen is cleared before lines are written.
1	Current content on screen is scrolled one line down.
2	Current content on screen is scrolled one line up.
7	Do not change the current content on the screen.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Character Encoding (3 bits)

This field MUST comply with Table 2.512:

Table 2.512: Screen Meta Data Report::Character Encoding encoding

Character Encoding	Description
0	Using standard ASCII codes, see *ASCII Codes* (values 128-255 are ignored)
1	Using standard ASCII codes and OEM Extended ASCII codes, see *ASCII Codes*
2	Unicode UTF-16
3	Using standard ASCII codes and Player codes, see *ASCII Codes* (undefined values are ignored)

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Note: Devices supporting Unicode UTF-16 characters are described by a 2 byte long decimal

representation. The first byte is the most significant byte. E.g. if there is one Unicode character in the set frame the char 1 will be MSB and char 2 will be LSB of the Unicode character.

Line Settings (3 bits)

This field MUST comply with Table 2.513:

Table 2.513: Screen Meta Data Report::Line Settings encoding

Line Settings	Description
0	Characters are written in selected font
1	Characters are written as highlighted
2	Characters are written using a larger font compared to line settings equal to 0

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Clear (1 bit)

Determine if the characters are written directly or line is cleared first.

Table 2.514: Screen Meta Data Report::Clear encoding

Clear	Description
0	Characters are written directly
1	Line is cleared before characters are written.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Line Number (4 bits)

The line number field indicates the line to write the characters to counting from zero (0..15).

Character Position (8 bits)

The character position field indicates where on the line to write the characters counting from zero (0..255). The character position may be larger than the display size in case the line buffer is bigger (See the Screen Attributes Report Command).

Number of Characters (8 bits)

The number of characters field indicates how many characters to be written on the screen for the specified line number, counting from 1.

Character (N bytes)

The character fields hold the string to output in specified character representation. Characters will be ignored in case there is no room left in the line buffer.

2.2.97 Screen Meta Data Command Class, version 2

The Screen Meta Data Command Class, version 2 introduces a Screen Timeout bit. The support for the Screen Timeout bit may be advertised by the Screen Attribute Command Class, version 2.

Details not mentioned remain the same as in version 1.

2.2.97.1 Screen Meta Data Report Command

This command is used to transfer data to the device hosting the screen. The size of the payload MUST NOT be bigger than 48 bytes. It is possible to write characters to multiple lines in the same frame.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SCREEN_MD (0x92)							
Command SCREEN_MD_REPORT							
More Data	Extended Setup	Screen Settings			Character Encoding		
Line Settings A			Clear A	Line Number A			
Character Position A							
Number of Characters A							
Character 1,A							
...							
Character N,A							
...							
Line Settings B			Clear B	Line Number B			
Character Position B							
Number of Characters B							
Character 1,B							
...							
Character N,B							

More Data (1 bit)

The more data bit indicates if additional reports are expected before the whole data streaming is completed. If the more data bit is set to 1 then additional reports are expected and the opposite if 0.

Extended Setup (1 bit)

If set to true, the last byte of the payload defines an extended setup.

Screen Settings (3 bits)

The screen settings identifier MUST be encoded according to Table 2.511.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Character Encoding (3 bits)

The Character Encoding identifier MUST be encoded according to Table 2.512.

Note: Devices supporting Unicode UTF-16 characters are described by a 2 byte long decimal representation. The first byte is the most significant byte. E.g. if there is one Unicode character in the set frame the char 1 will be MSB and char 2 will be LSB of the Unicode character.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.98 Sensor Configuration Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND HAS BEEN OBSOLETED

New implementations MUST NOT use the Sensor Configuration Command Class. Refer to the Configuration Command Classes.

The Sensor Configuration Command Class adds the possibility for sensors to act on either a measured value or on a preconfigured value. With this command class an application can act on a specific event. It is up to the application to implement the actual event. This could e.g. be implementation of the Association Command Class where the application would activate a group based on a trigger from the sensor.

The trigger types that may be configured are the same types as the values specified in the Multilevel Sensor Command Class

Most movement sensors may be configured to "ignore" movement if it is not dark. Typically this is done manually. With the Sensor Configuration Command Class this may be configured remotely in a standardised way.

A device supporting the Sensor Configuration Command Class may be configured via the trigger level, but the decision on what the level change should trigger is up to the application. For the movement sensor this trigger level could be an input parameter to the logic that controls the light.

2.2.98.1 Sensor Trigger Level Set Command

This command is used to set different triggers to either a specified value or to the current measured value. The Command also supports to restore a factory default value.

All configurable trigger types and values MUST be mapped directly from the Multilevel Sensor Command Class.

It is RECOMMENDED that all combinations of precision, scale and size parameters are supported. The Set command MUST support same format of precision, scale and size parameters as can be returned in the report command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SENSOR_CONFIGURATION (0x9E)</td></tr>
<tr><td colspan="8">Command = SENSOR_TRIGGER_LEVEL_SET (0x01)</td></tr>
<tr><td>Default</td><td>Current</td><td colspan="6">Reserved</td></tr>
<tr><td colspan="8">Sensor Type</td></tr>
<tr><td colspan="3">Precision</td><td colspan="2">Scale</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Trigger Value 1</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">Trigger Value N</td></tr>
</table>

Default (1 bit)

Reset level of trigger type to factory default when this bit is set to 1. If any value is set in this frame when the Default bit is 1 this value will be ignored.

Current (1 bit)

The current measured value will be stored as trigger value when this bit is set to 1. The trigger value in this frame will be ignored when the Current bit is set to 1.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Precision (3 bits)

The precision field describes what the precision of the trigger value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Scale (2 bits)

The Scale used to indicate what unit the trigger uses. Refer to the table in the Multilevel Sensor Command Class with respect to defined scales for the relevant triggers. Scales are defined by the Z-Wave Alliance.

Size (3 bits)

The size field indicates the number of bytes used for the Trigger Value field. This field can take values from 1 (001b), 2 (010b) or 4 (100b).

Sensor Type (8 bits)

The Sensor Type specifies what type of trigger this Command will set. Refer to the Multilevel Sensor Command Class specification, where Sensor Type is defined in the Multilevel Sensor Report Command.

Trigger Value (N bytes)

Refer to the Multilevel Sensor Report Command for information on what trigger values to set.

2.2.98.2 Sensor Trigger Level Get Command

This command can request the stored trigger level.

The Sensor Trigger Level Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_CONFIGURATION (0x9E)							
Command = SENSOR_TRIGGER_LEVEL_GET (0x02)							

2.2.98.3 Sensor Trigger Level Report Command

This command returns the stored trigger value.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_CONFIGURATION (0x9E)							
Command = SENSOR_TRIGGER_LEVEL_REPORT (0x03)							
Precision			Scale		Size		
Trigger Value 1							
...							
Trigger Value N							

For fields' description, refer to the Sensor Trigger Level Set Command.

2.2.98.4 Mapping Example

The report structure of the Multilevel Sensor Command Class can be mapped direct into the Sensor Configuration Set Command Class. This example frame below will set the trigger level in the receiving node to 10.25 degree Celsius.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_CONFIGURATION (0x9E)							
Command = SENSOR_TRIGGER_LEVEL_SET (0x01)							
Default (0)	Current (0)	Reserved					
Precision (010b)			Celsius (00b)		Size (010b)		
0x04							
0x01							

2.2.99 Simple AV Control Command Class, version 1-4

The Simple AV Control Command Class is used to control an AV device in a Z-Wave network. The Simple AV Control Command Class is suited for IR remote replacement. Furthermore, this command class supports Windows Vista Media Center and Media Center 2005 remote controls.

2.2.99.1 Simple AV Control Set Command

This command is used to control an AV device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SIMPLE_AV_CONTROL (0x94)							
Command = SIMPLE_AV_CONTROL_SET (0x01)							
Sequence Number							
Reserved					Key Attributes		
Reserved [OBSOLETED]							
Reserved [OBSOLETED]							
Command MSB,1							
Command LSB,1							
...							
Command MSB,N							
Command LSB,N							

Sequence Number (8 bits)

The sequence number is incremented each time a Simple AV Control Set Command is issued. The receiving node uses the sequence number to ignore duplicates.

Key Attributes (3 bits)

The key attributes specifies the state of the key. Currently the following key attribute definitions exist:

Table 2.515: Simple AV Control Set::Key Attributes encoding

Key Attribute	Description
0x00	Key Down - Sent when a new key is pressed. It is mandatory to send a Simple AV Control Set Command when this event occurs.
0x01	Key Up - Sent when the key is released. It is optional to send a Simple AV Control Set Command when this event occurs. Only the sequence number and key attribute parameter is changed in the Command.
0x02	Keep Alive - Sent every 100-200ms while the key is still held down. Event used as a failsafe feature for the ramping function, e.g. avoid volume jumps to maximum in case a key up event is not received. The keep alive event can also be used to control the speed of the ramping function, e.g. the first few seconds of the key held down is the speed slow and afterwards will it gradually accelerate. It is optional to send a Simple AV Control Set Command when this event occurs. Only the sequence number and key attribute parameter is changed in the Command.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Un-supported key attribute values MUST be ignored.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Command MSB, Command LSB (N * 16 bits)

This field is used to carry the AV Control command.

Each 16 bits-unit MUST carry a defined AV Control code. Refer to [31] for the defined AV Control codes. Values not defined in [31] are reserved and MUST NOT be used.

It is possible to specify a sequence of AV commands in one frame. If an AV control command is not supported, it MUST be ignored by a receiving node.

Command numbers 1 through 40 are the most popular commands used in AV remotes.

Command numbers 41 through 363 are less popular and is sorted in alphanumerical order.

Support for Windows Vista Media Center and Media Center 2005 remote controls is added with command numbers from 364 to 377 including 16, 200 and 231.

2.2.99.2 Simple AV Control Get Command

This command is used to request the number of reports necessary to report the supported AC Commands from the device.

The Simple AV Control Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SIMPLE_AV_CONTROL (0x94)							
Command = SIMPLE_AV_CONTROL_GET (0x02)							

2.2.99.3 Simple AV Control Report Command

This command is used to report the necessary number of reports to report the supported AC Commands from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SIMPLE_AV_CONTROL (0x94)							
Command = SIMPLE_AV_CONTROL_REPORT (0x03)							
Number of reports							

Number of reports (8 bits)

The number of reports necessary to report the entire list of supported AC Commands.

2.2.99.4 Simple AV Control Supported Get Command

This command is used to request the AV Commands supported by the AV device.

The Simple AV Control Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SIMPLE_AV_CONTROL (0x94)							
Command = SIMPLE_AV_CONTROL_SUPPORTED_GET (0x04)							
Report No							

Report No (8 bits)

Report no. field is used to request wanted report number. The report no. values MUST be a sequence starting from 1.

2.2.99.5 Simple AV Control Supported Report Command

This command is used to report the supported AC Commands from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SIMPLE_AV_CONTROL (0x94)							
Command = SIMPLE_AV_CONTROL_SUPPORTED_REPORT (0x05)							
Report No							
Bit Mask 1							
...							
Bit Mask N							

Report No (8 bits)

Report no. field specify the request report number.

Bit Mask (N bytes)

The Bit Mask fields describe the supported AV Control Commands by the device.

- Bit 0 in Bit Mask 1 indicates if Command #1 is supported.
- Bit 1 in Bit Mask 1 indicates if Command #2 is supported.
- ...

If a Command is supported, the bit MUST be set to 1. If a Command is not supported, the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported Command #. Mask fields bigger than 45 bytes not allowed. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

2.2.100 Sound Switch Command Class, version 1

The Sound Switch Command Class is used to manage nodes with a speaker or sound notification capability. It can be used for a doorbell, alarm clock, siren or any device issuing sound notifications.

2.2.100.1 Terminology

A **Sound Switch** is a sound notification device with pre-recorded tones. A tone can be a short sound effect as well as an entire song. The **volume** setting can be configured and the node will subsequently play tones using the configured volume for any tone. A **default tone** can also be configured, allowing the Sound Switch to play same sound when controlled by a node without the ability to select between tones.

A Sound Switch node advertises a **name** and **duration** for each tone. This information is used to guide an end-user in its tone selection and helps controlling nodes to know when the tone is finished playing.

2.2.100.2 Compatibility Considerations

The Sound Switch Command Class is an actuator control Command Class. Refer to Section 2.1.6.

CC:0079.01.00.21.001 A node supporting this Command Class MUST support at least 1 tone.

CC:0079.01.00.21.002 The implemented tone identifier values MUST be in a sequence starting from 1, i.e. a node supporting 10 tones MUST accept Tone Identifiers in the range 1..10.

2.2.100.3 Sound Switch Tones Number Get Command

This command is used to request the number of tones supported by the receiving node.

CC:0079.01.01.11.001 The Sound Switch Tones Number Report Command MUST be returned in response to this command.

CC:0079.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:0079.01.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONES_NUMBER_GET (0x01)							

2.2.100.4 Sound Switch Tones Number Report Command

This command is used to advertise the number of tones supported by the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONES_NUMBER_REPORT (0x02)							
Supported Tones							

Supported Tones (8 bits)

CC:0079.01.02.11.001 This field MUST be set to the total amount of supported tones.

CC:0079.01.02.11.003 This field MUST be in the range 1..254.

2.2.100.5 Sound Switch Tone Info Get Command

This command is used to query the information associated to a tone at a supporting node.

CC:0079.01.03.11.001 The Sound Switch Tone Info Report Command MUST be returned in response to this command.

CC:0079.01.03.11.002 This command MUST NOT be issued via multicast addressing.

CC:0079.01.03.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONE_INFO_GET (0x03)							
Tone Identifier							

Tone Identifier (8 bits)

This field is used to specify the requested Tone Identifier.

CC:0079.01.03.11.004 The implemented tone identifier values MUST be in a sequence starting from 1, i.e. a node supporting 10 tones MUST accept Tone Identifiers in the range 1..10.

CC:0079.01.03.11.005 A sending node MUST specify a Tone Identifier that is supported by a receiving node, i.e. in the range 1..{Total number of supported tones}.

CC:0079.01.03.11.006 A node receiving this command for an unsupported Tone Identifier or for Tone Identifier 0x00 MUST return a zero duration and a zero length name.

2.2.100.6 Sound Switch Tone Info Report Command

This command is used to advertise the information associated to a tone at a supporting node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONE_INFO_REPORT (0x04)							
Tone Identifier							
Tone Duration 1							
Tone Duration 2							
Name Length							
Name 1							
...							
Name N							

Tone Identifier (8 bits)

This field is used to advertise the Tone Identifier for which the associated information is being advertised.

Tone Duration (16 bits)

This field is used to advertise duration of the Tone Identifier.

CC:0079.01.04.11.001 This field MUST indicate the time in seconds it takes to play the actual Tone Identifier.

CC:0079.01.04.11.002 This field MUST be set to 0 if the Tone Identifier is set to 0x00 or a value higher than the total number of supported tones advertised in the Sound Switch Tones Number Report Command.

Name Length (8 bits)

This field indicates the length in bytes of the Name field.

CC:0079.01.04.11.003

This field MUST be in the range 1..255 if the Tone Identifier is supported.

CC:0079.01.04.11.004 This field MUST be set to 0 if the Tone Identifier is set to 0x00 or a value higher than the total number of supported tones advertised in the Sound Switch Tones Number Report Command.

Name (N bytes)

This field is used to indicate the assigned name or label for the actual Tone Identifier.

CC:0079.01.04.11.005 The length of this field in bytes MUST comply with the advertised value in the Name Length field. This field MUST be omitted if the Name Length field is set to 0.

CC:0079.01.04.11.006 The field MUST be formatted as a byte array with no zero termination.

CC:0079.01.04.11.007 The characters MUST be encoded in UTF-8 format.

2.2.100.7 Sound Switch Configuration Set Command

This command is used to set the configuration for playing tones at the supporting node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_CONFIGURATION_SET (0x05)							
Volume							
Default Tone Identifier							

Volume (8 bits)

This field is used to specify the volume at which the node will play tones.

The encoding of the Volume field MUST be according to Table 2.516.

Table 2.516: Sound Switch Configuration Set::Volume encoding

Value	Description
CC:0079.01.05.11.002 0 (0x00)	This value MUST indicate an Off/Mute volume setting (0%)
CC:0079.01.05.11.003 1..100 (0x01..0x64)	These values MUST indicate the actual volume setting from respectively 1% to 100%. A supporting node MAY implement fewer than 100 hardware level. In this case, mapping of hardware levels MUST be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level
CC:0079.01.05.11.004 255 (0xFF)	This value MUST indicate to restore most recent non-zero volume setting. This value MUST be ignored if the current volume is not zero (0x00). This value MAY be used to set the Default Tone Identifier and do not modify the volume setting

CC:0079.01.05.11.005 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Default Tone Identifier (8 bits)

This field is used to specify the Default Tone. This Tone will be played if receiving the Sound Switch Tone Play Set Command for an unsupported value or for the 0xFF value.

CC:0079.01.05.11.006 The value 0x00 MUST indicate that the receiving node MUST NOT update its current default tone and the command is sent to configure the volume only.

CC:0079.01.05.11.007 Values in the range 1..{Total number of supported tones} MUST indicate that the receiving node MUST use the specified Identifier as Default Tone.

CC:0079.01.05.11.008 Values higher than the Total number of supported tones advertised in the Sound Switch Tones Number Report Command MUST be ignored by a receiving node.

2.2.100.8 Sound Switch Configuration Get Command

This command is used to request the current configuration for playing tones at the supporting node.

CC:0079.01.06.11.001 The Sound Switch Configuration Report Command MUST be returned in response to this command.

CC:0079.01.06.11.002 This command MUST NOT be issued via multicast addressing.

CC:0079.01.06.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_CONFIGURATION_GET (0x06)							

2.2.100.9 Sound Switch Configuration Report Command

This command is used to advertise the current configuration for playing tones at the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_CONFIGURATION_REPORT (0x07)							
Volume							
Default Tone Identifier							

Volume (8 bits)

This field is used to advertise the current volume setting at the sending node.

CC:0079.01.07.11.001 This field MUST be in the range 0..100. Values in the range 0..100 MUST indicate the current volume percentage.

Default Tone Identifier (8 bits)

This field is used to advertise the current configured Default Tone. This Tone Identifier will be played if receiving a Sound Switch Tone Play Set Command for an unsupported value or for the 0xFF value.

CC:0079.01.07.11.002 This field MUST be in the range 1..{Total number of supported Tones}.

2.2.100.10 Sound Switch Tone Play Set Command

This command is used to instruct a supporting node to play (or stop playing) a tone.

CC:0079.01.08.11.001 The supporting node MUST play the specified tone immediately when receiving this command.

CC:0079.01.08.11.002 A node already playing a tone MUST interrupt the current tone and start playing the tone specified in this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONE_PLAY_SET (0x08)							
Tone Identifier							

Tone Identifier (8 bits)

CC:0079.01.08.11.003 This field is used to specify the tone that the receiving node MUST play.

CC:0079.01.08.11.004 This field MUST be encoded according to Table 2.517.

Table 2.517: Sound Switch Tone Play Set::Tone Identifier encoding

Value	Description
CC:0079.01.08.11.005 0x00	No Tone MUST be played. A supporting node MUST stop playing any tone when receiving this value.
CC:0079.01.08.11.006 0x01..0xFE	A supporting node MUST play the specified Tone identifier using the configured volume setting. A node receiving a non-supported Tone Identifier (higher than the Total number of supported tones) MUST play the default tone using the configured volume setting.
CC:0079.01.08.11.007 0xFF	The supporting node MUST play the default tone using the configured volume setting.

2.2.100.11 Sound Switch Tone Play Get Command

This command is used to request the current tone being played by the receiving node.

CC:0079.01.09.11.001 The Sound Switch Tone Play Report Command MUST be returned in response to this command.

CC:0079.01.09.11.002 This command MUST NOT be issued via multicast addressing.

CC:0079.01.09.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONE_PLAY_GET (0x09)							

2.2.100.12 Sound Switch Tone Play Report Command

This command is used to advertise the current tone being played by the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONE_PLAY_REPORT (0x0A)							
Tone Identifier							

Tone Identifier (8 bits)

CC:0079.01.0A.11.001 This field MUST indicate which tone is currently being played by the sending node. This field MUST be in the range 0..{Total number of supported tones}.

CC:0079.01.0A.11.002 The value 0 MUST indicate that no Tone is currently being played.

This field MUST be set to 0x00 by a sending node if the configured volume is 0x00 (muted).

CC:0079.01.0A.11.003 Values in the range 1..{Total number of supported tones} MUST indicate the Tone that is currently being played.

CC:0079.01.0A.11.004 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.101 Sound Switch Command Class, version 2

The Sound Switch Command Class is used to manage nodes with a speaker or sound notification capability. It can be used for a doorbell, alarm clock, siren or any device issuing sound notifications.

2.2.101.1 Compatibility Considerations

The Sound Switch Command Class is backwards compatible with the Sound Switch Command Class, version 1.

CC:0079.02.00.21.001 All Commands and fields not mentioned in this version MUST remain unchanged from the Sound Switch Command Class, version 1.

This version extends the following commands to allow a controlling node to modify the volume setting only for a single Tone Play Set Command:

- Sound Switch Tone Play Set Command
- Sound Switch Tone Play Report Command

2.2.101.2 Sound Switch Tone Play Set Command

This command is used to instruct a supporting node to play (or stop playing) a tone.

CC:0079.02.08.11.001 The supporting node MUST play the specified tone immediately when receiving this command.

CC:0079.02.08.11.002 A node already playing a tone MUST interrupt the current tone and start playing the tone specified in this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONE_PLAY_SET (0x08)							
Tone Identifier							
Play Command Tone Volume							

CC:0079.02.08.11.003 All fields not described below MUST remain unchanged from version 1.

Play Command Tone Volume (8 bits)

This field is used to configure the volume for the actual Play Command.

CC:0079.02.08.11.005 The volume setting configured with the Sound Switch Configuration Set Command MUST remain
CC:0079.02.08.11.006 unchanged by this command and this field MUST be used for the volume of the actual tone that will start playing.

CC:0079.02.08.11.007 This field MUST be set to 0x00 and ignored by a receiving node if the Tone identifier field is set to 0x00.

CC:0079.02.08.11.008 The encoding of this field MUST be according to Table 2.518.

Table 2.518: Sound Switch Tone Play Set::Play Command Volume encoding

Value	Description
0 (0x00)	This value MUST indicate to use the configured volume at the node (Sound Switch Configuration Set Command).
1..100 (0x01..0x64)	These values MUST indicate the actual volume setting from respectively 1% to 100% to use for the Play Set Command This value SHOULD be used for critical applications (e.g. security alarm) where it is important to have the sound played even when the device is muted.
255 (0xFF)	This value MUST indicate to use most recent non-zero volume setting if the current volume is muted (0x00). If the current configured volume is not muted (>0x00), this value MUST indicate to use the configured current volume at the node (Sound Switch Configuration Set Command) This value SHOULD be used only for critical application (e.g. security alarm) where it is important to have the sound played even when the device is muted.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.101.3 Sound Switch Tone Play Report Command

This command is used to advertise the current tone being played by the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SOUND_SWITCH (0x79)							
Command = SOUND_SWITCH_TONE_PLAY_REPORT (0x0A)							
Tone Identifier							
Play Command Tone Volume							

All fields not described below MUST remain unchanged from version 1.

Play Command Tone Volume (8 bits)

CC:0079.02.0A.11.001 This field MUST advertise the actual playing volume represented with the value from 0x00 …0x64 (0 … 100%). The values 0x00 and 0x64 MUST represent mute and maximum volume, respectively.

This field MUST be set to 0x00 if the Tone Identifier field is set to 0x00.

2.2.102 Tariff Table Configuration Command Class, version 1

The Tariff Table Configuration Command Class defines the cost for a range of rates.

The Tariff Table configuration commands are separated for the Tariff Table monitor commands in the Tariff Table Monitor Command Class, allowing the classes to be optionally supported at different Z-Wave security levels.

2.2.102.1 Tariff Table Supplier Set Command

This command is used to set the name of the utility supplier in the metering device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_CONFIG (0x4A)							
Command = TARIFF_TBL_SUPPLIER_SET (0x01)							
Utility Timestamp Year 1							
Utility Timestamp Year 2							
Utility Timestamp Month							
Utility Timestamp Day							
Utility Timestamp Hour Local Time							
Utility Timestamp Minute Local Time							
Utility Timestamp Second Local Time							
Currency 1							
Currency 2							
Currency 3							
Standing Charge Precision			Standing Charge Period				
Standing Charge Value 1							
Standing Charge Value 2							
Standing Charge Value 3							
Standing Charge Value 4							
Reserved			Number of Supplier Characters				
Supplier Character 1							
...							
Supplier Character N							

Utility Timestamp Year (16 bits)

Tariff applies from the specified year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Utility Timestamp Month (8 bits)

Tariff applies from the specified month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that a accumulated value is not determined yet.

Utility Timestamp Day (8 bits)

Tariff applies from the specified day of the month between 01 and 31.

Utility Timestamp Hour Local Time (8 bits)

Tariff applies from the specified number of complete hours that have passed since midnight (00-23) in local time.

Utility Timestamp Minute Local Time (8 bits)

Tariff applies from the specified number of complete minutes that have passed since the start of the hour (00-59) in local time.

Utility Timestamp Second Local Time (8 bits)

Tariff applies from the specified number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Currency (3 bytes)

ISO 4217 defines the currency code. In Table 2.519 are some examples of the codes listed:

Table 2.519: Tariff Table Supplier Set::Currency encoding examples

Currency Code	Currency 1	Currency 2	Currency 3
Pound sterling	G	B	P
US Dollar	U	S	D

Standing Charge Period (5 bits)

This field indicates the stated period that standing charge applies e.g. 50p/week.

Table 2.520: Tariff Table Supplier Set::Standing Charge Period encoding

Period	Value
Weekly	0x01
Monthly	0x02
Quarterly	0x03
Yearly	0x04
Reserved	0x05-0x1F

Standing Charge Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Standing Charge Value (32 bits)

The Standing Charge value MUST be encoded as a 32-bit signed integer. The first byte MUST be the most significant byte. Table 2.12 shows signed decimal values together with their hexadecimal equivalents.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Supplier Characters (5 bits)

Number of characters defining the name of the utility supplier ID (1 … 32).

Supplier Character (N bytes)

The supplier character fields hold the string identifying the utility supplier. The character presentation uses standard ASCII codes (values 128-255 are ignored).

2.2.102.2 Tariff Table Set Command

This command adds a tariff to a given rate parameter set identifier.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_CONFIG (0x4A)							
Command = TARIFF_TBL_SET (0x02)							
Rate Parameter Set ID							
Tariff Precision			Reserved				
Tariff Value 1							
Tariff Value 2							
Tariff Value 3							
Tariff Value 4							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID indicates the requested parameter set.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Tariff Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Tariff Value (32 bits)

The Tariff value is a 32 bit signed field. The first byte is the most significant byte. shows signed decimal values together with their hexadecimal equivalents.

2.2.102.3 Tariff Table Remove Command

This command is used to remove rate parameter set(s).

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_CONFIG (0x4A)							
Command = TARIFF_TBL_REMOVE (0x03)							
Reserved		Rate Parameter Set IDs					
Rate Parameter Set ID 1							
...							
Rate Parameter Set ID N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Parameter Set IDs (6 bits)

The Rate Parameter Set ID's indicates the number of Rate Parameter Set ID's in the command.

Rate Parameter Set ID (N bytes)

These fields contain a list of Tariffs to be removed from the Tariff Table. All Tariffs are cleared in case no Rate Parameter Set ID's are supplied.

2.2.103 Tariff Table Monitor Command Class, version 1

The Tariff Table Monitor Command Class defines the cost for a range of rates.

2.2.103.1 Tariff Table Supplier Get Command

This command is used to request the name of the utility supplier.

The Tariff Table Supplier Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_TBL_MONITOR (0x4B)							
Command = TARIFF_TBL_SUPPLIER_GET (0x01)							

2.2.103.2 Tariff Table Supplier Report Command

This command is used to advertise the name of the utility supplier.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_TBL_MONITOR (0x4B)							
Command = TARIFF_TBL_SUPPLIER_REPORT (0x02)							
Utility Timestamp Year 1							
Utility Timestamp Year 2							
Utility Timestamp Month							
Utility Timestamp Day							
Utility Timestamp Hour Local Time							
Utility Timestamp Minute Local Time							
Utility Timestamp Second Local Time							
Currency 1							
Currency 2							
Currency 3							
Standing Charge Precision				Standing Charge Period			
Standing Charge Value 1							
Standing Charge Value 2							
Standing Charge Value 3							
Standing Charge Value 4							
Reserved				Number of Supplier Characters			
Supplier Character 1							
...							
Supplier Character N							

Refer to description of fields under the Tariff Table Supplier Set Command (Section 2.2.102.1).

2.2.103.3 Tariff Table Get Command

This command is used to request the tariff for the corresponding rate parameter set.

The Tariff Table Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_TBL_MONITOR (0x4B)							
Command = TARIFF_TBL_GET (0x03)							
Rate Parameter Set ID							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID addresses the price for the accompanying rate parameter set. The Rate Table Supported Report Command determines the number of supported rate parameter sets.

2.2.103.4 Tariff Table Report Command

This command is used to advertise information relating to a given Rate Parameter Set Identifier.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_TBL_MONITOR (0x4B)							
Command = TARIFF_TBL_REPORT (0x04)							
Rate Parameter Set ID							
Tariff Precision			Reserved				
Tariff Value 1							
Tariff Value 2							
Tariff Value 3							
Tariff Value 4							

Refer to description of fields under the Tariff Table Set Command (Section 2.2.102.2)

2.2.103.5 Tariff Table Cost Get Command

This command is used to request the cost according to rate parameter set ID, rate type, dataset mask and time interval.

The Tariff Table Cost Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_TBL_MONITOR (0x4B)							
Command = TARIFF_TBL_COST_GET (0x05)							
Rate Parameter Set ID							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Stop Year 1							
Stop Year 2							
Stop Month							
Stop Day							
Stop Hour Local Time							
Stop Minute Local Time							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID indicates the requested parameter set. Rate Parameter Set ID equal to 0xFF returns overall accumulated cost.

Start Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Start Day (8 bits)

Specify the day of the month between 01 and 31.

Start Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Stop Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte. Setting the parameter to 0xFFFF indicates now.

Stop Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that an accumulated value is not determined yet. Setting the parameter to 0xFF indicates now.

Stop Day (8 bits)

Specify the day of the month between 01 and 31. Setting the parameter to 0xFF indicates now.

Stop Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time. Setting the parameter to 0xFF indicates now.

Stop Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time. Setting the parameter to 0xFF indicates now.

Stop Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported. Setting the parameter to 0xFF indicates now.

2.2.103.6 Tariff Table Cost Report Command

This command is used to report a number of time stamped values (historical) in physical units in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TARIFF_TBL_MONITOR (0x4B)							
Command = TARIFF_TBL_COST_REPORT (0x06)							
Rate Parameter Set ID							
Reserved						Rate Type	
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Stop Year 1							
Stop Year 2							
Stop Month							
Stop Day							
Stop Hour Local Time							
Stop Minute Local Time							
Currency 1							
Currency 2							
Currency 3							
Cost Precision			Reserved				
Cost Value 1							
Cost Value 2							
Cost Value 3							
Cost Value 4							

Rate Parameter Set ID (8 bits)

The Rate Parameter Set ID indicates the requested parameter set. Rate Parameter Set ID equal to 0xFF returns accumulated cost.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. This field MUST be encoded according to Table 2.345.

Start/Stop Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start/Stop Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that a accumulated value is not determined yet.

Start/Stop Day (8 bits)

Specify the day of the month between 01 and 31.

Start/Stop Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start/Stop Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Currency (3 bytes)

ISO 4217 defines the currency code. Examples are given in Table 2.519.

Cost Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Cost Value (32 bits)

The Cost value is a 32 bit un-signed field. The first byte is the most significant byte.

The value 0xFFFFFFFF is reserved, and SHOULD be used to report that the cost calculation has not yet been performed.

2.2.104 Thermostat Fan Mode Command Class, version 1

The Thermostat Fan Mode Command Class, version 1 used for the HVAC's systems manual fan.

2.2.104.1 Thermostat Fan Mode Set Command

This command is used to set the fan mode in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SET (0x01)							
Reserved				Fan Mode			

Fan Mode (8 bits)

This field MUST comply with the values indicated for version 1 in Table 2.521.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.104.2 Thermostat Fan Mode Get Command

This command is used to request the fan mode in the device.

The Thermostat Fan Mode Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_GET (0x02)							

2.2.104.3 Thermostat Fan Mode Report Command

This command is used to report the fan mode in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_GET (0x02)							
Reserved				Fan Mode			

Fan Mode (8 bits)

Refer to description under Thermostat Fan Mode Set Command (Section 2.2.104.1).

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.104.4 Thermostat Fan Mode Supported Get Command

This command is used to request the supported fan modes from the device.

The Thermostat Fan Mode Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SUPPORTED_GET (0x04)							

2.2.104.5 Thermostat Fan Mode Supported Report Command

This command is used to report the supported fan modes from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields describe the supported fan modes by the thermostat.

- Bit 0 in Bit Mask 1 indicates if Fan Mode = 0 (Auto / Auto Low) is supported.
- Bit 1 in Bit Mask 1 indicates if Fan Mode = 1 (On / On Low) is supported.
- ...

If a Fan Mode is supported the bit MUST be set to 1. If a Fan Mode is not supported the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported fan mode. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

2.2.105 Thermostat Fan Mode Command Class, version 2

The Thermostat Fan Mode Command Class, version 2 is used for the HVAC's systems manual fan.

The commands not mentioned here will remain the same as specified for Thermostat Fan Mode Command Class (Version 1).

2.2.105.1 Thermostat Fan Mode Set Command

This command is used to set the fan mode in the device

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SET (0x01)							
Off	Reserved			Fan Mode			

Off (1 bit)

The "Off bit" set to "1" will switch the fan fully OFF regardless of what fan mode has been set. In order to activate a fan mode the "Off bit" MUST be set to "0".

Fan Mode (4 bits)

This field MUST comply with the values indicated for version 2 and older in Table 2.521.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.105.2 Thermostat Fan Mode Report Command

This command is used to report the fan mode in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SET (0x01)							
Reserved				Fan Mode			

Fan Mode (4 bits)

Refer to description under the Thermostat Fan Mode Set Command.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

2.2.106 Thermostat Fan Mode Command Class, version 3

The Thermostat Fan Mode Command Class, version 3 is used for the HVAC's systems manual fan.

2.2.106.1 Thermostat Fan Mode Set Command

This command is used to set the fan mode in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SET (0x01)							
Off	Reserved			Fan Mode			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Off (1 bit)

The "Off bit" set to "1" will switch the fan fully OFF. In order to activate a fan mode the "Off bit" MUST be set to "0". However, for some applications it is critical that the fan is ON in certain modes. In this case, the application can decide to ignore the Off bit.

Fan Mode (4 bits)

This field MUST comply with the values indicated for version 3 and older in Table 2.521.

2.2.106.2 Thermostat Fan Mode Get Command

This command is used to request the fan mode in the device.

The Thermostat Fan Mode Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_GET (0x02)							

2.2.106.3 Thermostat Fan Mode Report Command

This command is used to report the fan mode in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_REPORT (0x03)							

Fan Mode (4 bits)

Refer to description under Thermostat Fan Mode Set Command (Section 2.2.106.1).

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Off (1 bit)

The “Off bit” set to “1” indicates that the fan is fully OFF. The “Off bit” set to “0” indicates that it is possible to change between Fan Modes.

For some applications, it is critical that the fan is ON in certain modes. In this case, the application can decide to ignore the Off bit. This means that the Off bit in the Report MUST always be set to “0”

2.2.107 Thermostat Fan Mode Command Class, version 4-5

The Thermostat Fan Mode Command Class is an extension to support control and status monitoring functions of air-conditioning devices in order to achieve a global framework that covers the majority of generic functions implemented by world-wide air-conditioning manufacturer. The new features comprises of new Thermostat Fan Modes.

2.2.107.1 Thermostat Fan Mode Set Command

This command is used to set the fan mode in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SET (0x01)							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Off (1 bit)

"Off bit" set to "1" will switch the fan fully OFF. In order to activate a fan mode the "Off bit" MUST be set to "0". However for some applications it is critical that the fan is ON in certain modes. In this case the application may ignore the off bit.

Fan Mode (4 bits)

If the device transmitting the Thermostat Fan Mode Set command attempts to set a non-supported mode, the receiving thermostat device MUST ignore the command.

Table 2.521: Thermostat Fan Mode Set version 4::Fan Mode encoding

Fan Mode (4 bits)		Description	CC Version
0x00	AUTO LOW	Will turn the manual fan operation off unless turned on by the manufacturer specific "auto low" algorithms.	1
0x01	LOW	Will turn the manual fan operation on. Low speed is selected.	1
0x02	AUTO HIGH	Will turn the manual fan operation off unless turned on by the manufacturer specific "auto high" algorithms.	1
0x03	HIGH	Will turn the manual fan operation on. High speed is selected.	1
0x04	AUTO MEDIUM	Will turn the manual fan operation off unless turned on by the manufacturer specific "auto medium" algorithms.	2
0x05	MEDIUM	Will turn the manual fan operation on. Medium speed is selected.	2
0x06	CIRCULATION	Will turn the manual fan operation off unless turned on by the manufacturer specific circumlation algorithms.	3
0x07	HUMIDITY CIRCULATION	Will turn the manual fan operation off unless turned on by the manufacturer specific "humidity circulation" algorithms.	3
0x08	LEFT & RIGHT	Will turn the manual fan operation off unless turned on by the manufacturer specific "left & right" circulation algorithms.	4

continues on next page

Table 2.521 – continued from previous page

Fan Mode (4 bits)		Description	CC Version
0x09	UP & DOWN	Will turn the manual fan operation off unless turned on by the manufacturer specific “up & down” circulation algorithms.	4
0x0A	QUIET	Will turn the manual fan operation off unless turned on by the manufacturer specific “quiet” algorithms.	4
0x0B	EXTERNAL CIRCULATION	Will turn the manual fan operation off unless turned on by the manufacturer specific circulation algorithms. This mode will circulate fresh air from the outside.	5
0x0C..0x0F	Reserved	These values/modes are reserved for future use. The values cannot be supported by any device and will be ignored.	-

2.2.107.2 Thermostat Fan Mode Get Command

This command is used to request the fan mode in the device.

The Thermostat Fan Mode Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_GET (0x02)							

2.2.107.3 Thermostat Fan Mode Report Command

This command is used to report the fan mode in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_REPORT (0x03)							

Refer to Thermostat Fan Mode Set command (Section 2.2.107.1) for parameter/field descriptions.

2.2.107.4 Thermostat Fan Mode Supported Get Command

This command is used to request the supported modes from the device.

The Thermostat Fan Mode Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SUPPORTED_GET (0x04)							

2.2.107.5 Thermostat Fan Mode Supported Report Command

This command is used to report the supported thermostat modes from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields describe the supported modes by the device. Refer to Thermostat Fan Mode Set Command (Section 2.2.107.1).

- Bit 0 in Bit Mask 1 field indicates support for mode = 0 (AUTO LOW)
- Bit 1 in Bit Mask 1 field indicates support for mode = 1 (LOW)
- Bit 2 in Bit Mask 1 field indicates support for mode = 2 (AUTO HIGH)
- ...

The mode is supported if the bit is 1 and the opposite if 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported mode. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

Example:

To indicate the thermostat device supports AUTO LOW, AUTO HIGH and AUTO MEDIUM, the Thermostat Fan Mode Supported Report command MUST be structured as illustrated below.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_MODE (0x44)							
Command = THERMOSTAT_FAN_MODE_SUPPORTED_REPORT (0x05)							
0	0	0	**1**	0	**1**	0	**1**

2.2.108 Thermostat Fan State Command Class, version 1-2

The Thermostat Fan State Command Class is used to obtain the fan operating state of the thermostat.

2.2.108.1 Compatibility Considerations

A device supporting Thermostat Fan State CC, Version 2 MUST support Thermostat Fan State CC, Version 1.

Version 2 adds Fan Operating State identifiers for use in the Thermostat Fan State Report Command.

Commands not described in Version 2 stays unchanged from version 1.

2.2.108.2 Thermostat Fan State Get Command

This command is used to request the fan operating state from the device.

The Thermostat Fan State Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_STATE (0x45)							
Command = THERMOSTAT_FAN_STATE_GET (0x02)							

2.2.108.3 Thermostat Fan State Report Command

This command is used to report the fan operating state of the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_FAN_STATE (0x45)							
Command = THERMOSTAT_FAN_STATE_REPORT (0x03)							
Reserved				Fan Operating State			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Fan Operating State (4 bits)

The fan operating state identifier MUST comply with Table 2.522.

Table 2.522: Thermostat Fan State Report::Fan Operating State encoding

Fan Operating State	Description	CC Version
0	Idle/Off	1
1	Running/Running Low - If device only supports one fan speed, this state is used to report the fan is running. If the device is a multi-speed device, this state is used to report that the fan is running at the low speed.	1
2	Running High	1
3	Running Medium	2
4	Circulation Mode	2
5	Humidity Circulation Mode	2
6	Right-Left Circulation Mode	2
7	Up-Down Circulation Mode	2
8	Quiet Circulation Mode	2

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.109 Thermostat Mode Command Class, version 1-2

The Thermostat Mode Command Class is used to control which mode a thermostat operates.

2.2.109.1 Interoperability Considerations

Most thermostat modes require a supporting node to have an associated setpoint, which can managed with the Thermostat Setpoint Command Class. A node supporting this Command Class SHOULD support the Thermostat Setpoint Command Class.

2.2.109.2 Thermostat Mode Set Command

This command is used to set the thermostat mode at the receiving node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_MODE (0x40)							
Command = THERMOSTAT_MODE_SET (0x01)							
Reserved			Mode				

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Mode (5 bits)

This field is used to set the thermostat mode at the receiving node.

This field MUST be encoded according to Table 2.523.

2.2.109.3 Thermostat Mode Get Command

This command is used to request the current mode set at the receiving node.

The Thermostat Mode Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_MODE (0x40)							
Command = THERMOSTAT_MODE_GET (0x02)							

2.2.109.4 Thermostat Mode Report Command

This command is used to report the mode from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_MODE (0x40)							
Command = THERMOSTAT_MODE_REPORT (0x03)							
Reserved			Mode				

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Mode (5 bits)

This field is used to advertise the current thermostat mode at the receiving node.

This field MUST be encoded according to Table 2.523.

This field MUST NOT be set to 0x05 (RESUME(ON)) in this command.

2.2.109.5 Thermostat Mode Supported Get Command

This command is used to request the supported modes of a node.

The Thermostat Mode Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_MODE (0x40)							
Command = THERMOSTAT_MODE_SUPPORTED_GET (0x04)							

2.2.109.6 Thermostat Mode Supported Report Command

This command is used to advertise which modes are supported by the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_MODE (0x40)							
Command = THERMOSTAT_MODE_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

This field advertises the Modes supported by the supporting node.

A supporting node MUST support at least one mode. Modes and their minimum required version are described in Table 2.523. A node MUST NOT support a mode associated to a newer version than the version it supports.

- Bit 0 in Bit Mask 1 represents Mode = 0x00 (Off).
- Bit 1 in Bit Mask 1 represents Mode = 0x01 (Heat).
- ...

If a Mode is supported, the corresponding bit MUST be set to '1'.

If a Mode is not supported, the corresponding bit MUST be set to '0'.

This length of this field MUST be set to the minimum amount of bytes which allows advertising all supported Modes.

2.2.110 Thermostat Mode Command Class, version 3

The Thermostat Mode Command Class is used to control which mode a thermostat operates.

2.2.110.1 Compatibility Considerations

The Thermostat Mode Command Class, version 3 is backwards compatible with the Thermostat Mode Command Class, version 1-2.

All commands and fields not mentioned in this version MUST remain unchanged from the Thermostat Mode Command Class, version 1-2.

This version introduces:

- A new FULL POWER thermostat mode
- MANUFACTURER SPECIFIC mode

2.2.110.2 Interoperability Considerations

The MANUFACTURER SPECIFIC (proprietary) mode MUST NOT be supported by a node if the functionality can be provided using a thermostat mode defined in this command class.

A node supporting the MANUFACTURER SPECIFIC mode MUST fulfill the following conditions:

- The node MUST support as a minimum two other thermostat modes (e.g. HEAT and COOL).
- If the MANUFACTURER SPECIFIC mode functionality can in part be supported by one or more defined thermostat mode in this command class, the node MUST also support these thermostat modes.
- The MANUFACTURER SPECIFIC mode and all of its associated Manufacturer Data fields MUST be described in the product manual.

2.2.110.3 Thermostat Mode Set Command

This command is used to set the thermostat mode at the receiving node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_MODE (0x40)							
Command = THERMOSTAT_MODE_SET (0x01)							
Number of Manufacturer Data fields			Mode				
Manufacturer Data 1							
...							
Manufacturer Data N							

Number of Manufacturer Data fields (3 bits)

This field is used to advertise the length in bytes of the Manufacturer Data field in the command. This field MUST be in the range 0..7.

This field MUST be set to 0 if the Mode field is not set to 0x1F (MANUFACTURER SPECIFIC).

This field MUST indicate the length in bytes of the Manufacturer Data field if the Mode field is set to 0x1F (MANUFACTURER SPECIFIC).

Mode (5 bits)

This field is used to set the thermostat mode at the receiving node.

This field MUST be encoded according to Table 2.523.

Manufacturer Data (N bytes)

This field is used to provide a configuration for the MANUFACTURER SPECIFIC mode. Refer to Interoperability considerations (Section 2.2.110.2).

Table 2.523: Thermostat Mode Set version 3::Mode encoding

Mode (5 bits)		Description	Requires own Setpoint	CC Version
0x00	OFF	This mode is used to switch off the thermostat.	No	1
0x01	HEAT	This mode is used to use activate heating when the temperature is below the Heating (0x01) setpoint.	Yes	1
0x02	COOL	This mode is used to use activate cooling when the temperature is above the Cooling (0x02) setpoint.	Yes	1
0x03	AUTO	This mode is used to regulate the temperature using heating and cooling when the temperature is outside the range defined by the Heating (0x01) and Cooling (0x02) setpoints.	No	1
0x04	AUXILIARY	This mode is used to activate heating when the temperature is below the Heating (0x01) setpoint, but using an auxiliary or emergency heat source. For example, a heat pump is not more efficient when the outside temperature is too low. The auxiliary heat mode may be activated to use a more efficient secondary heat source.	No	1
0x05	RESUME (ON)	This mode is used to resume the last activate mode (different than OFF 0x00).	No	1
0x06	FAN	This mode is used to activate fans only and circulate air.	No	1
0x07	FURNACE	This mode is used to activate fans to circulate air and heating or cooling will be activated to regulate the temperature at the Furnace (0x07) setpoint.	Yes	1
0x08	DRY	This mode is used to dehumidify and remove moisture. Heating or cooling will be activated to regulate the temperature at the Dry Air (0x08) setpoint.	Yes	1
0x09	MOIST	This mode is used to humidify and add moisture. Heating or cooling will be activated to regulate the temperature at the Moist Air (0x09) setpoint.	Yes	1
0x0A	AUTO CHANGEOVER	This mode is used to regulate the temperature at the Auto Changeover (0x0A) setpoint using heating and cooling.	Yes	1
0x0B	ENERGY HEAT	This mode is used to activate heating when the temperature is below the Energy Save Heating (0x0B) setpoint. The Energy Save Heating (0x0B) setpoint is usually lower than the Heating (0x01) setpoint in order to save energy.	Yes	2

continues on next page

Table 2.523 – continued from previous page

Mode (5 bits)		Description	Requires own Setpoint	CC Version
0x0C	ENERGY COOL	This mode is used to activate cooling when the temperature is below the Energy Save Cooling (0x0C) setpoint. The Energy Save Cooling (0x0C) setpoint is usually higher than the Cooling (0x02) setpoint in order to save energy.	Yes	2
0x0D	AWAY	This mode is used to regulate the temperature using heating and cooling when the temperature is outside the range defined by the Away Heating (0x0D) and Away Cooling (0x0E) setpoints.	Yes	2
0x0E	Reserved		-	3
0x0F	FULL POWER	This mode is used to regulate the temperature at the Full Power (0x0F) setpoint using heating and cooling. This mode is intended to use more energy and speed up the temperature regulation to the desired setpoint.	Yes	3
0x1F	MANUFACTURER SPECIFIC	Reserved for vendor specific thermostat mode.	No	3

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.110.4 Thermostat Mode Report Command

This command is used to report the current mode of the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_MODE (0x40)							
Command = THERMOSTAT_MODE_REPORT (0x03)							
Number of Manufacturer Data fields			Mode				
Manufacturer Data 1							
...							
Manufacturer Data N							

No of Manufacturer Data fields (3 bits)

This field is used to advertise the length in bytes of the Manufacturer Data field in the command. This field MUST be in the range 0..7.

This field MUST be set to 0 if the Mode field is not set to 0x1F (MANUFACTURER SPECIFIC).

This field MUST indicate the length in bytes of the Manufacturer Data field if the Mode field is set to 0x1F (MANUFACTURER SPECIFIC).

Mode (5 bits)

This field is used to advertise the current thermostat mode at the receiving node.

This field MUST be encoded according to Table 2.523.

This field MUST NOT be set to 0x05 (RESUME(ON)) in this command.

Manufacturer Data (N bytes)

This field is used to advertise the current configuration for the MANUFACTURER SPECIFIC mode. Refer to Interoperability considerations (Section 2.2.110.2).

2.2.111 Thermostat Operating State Command Class, version 1

The Thermostat Operating State Command Class is used to obtain the operating state of the thermostat.

2.2.111.1 Thermostat Operating State Get Command

This command is used to request the operating state.

The Thermostat Operating State Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_STATE_GET (0x02)							

2.2.111.2 Thermostat Operating State Report Command

The Thermostat Operating State Report Command is used to report the operating state.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_STATE_REPORT (0x03)							
Reserved				Operating State			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Operating State (4 bits)

The thermostat operating state identifier MUST comply with Table 138.

2.2.112 Thermostat Operating State Command Class, version 2

The Thermostat Operating State Command Class is used to obtain the operating state of the thermostat as well as logged operating runtime times of thermostat.

2.2.112.1 Thermostat Operating State Get

This command gets the operating state of the thermostat.

The Thermostat Operating State Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_STATE_GET (0x02)							

2.2.112.2 Thermostat Operating State Report

This command is used to report the operating state.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_STATE_REPORT (0x03)							
Operating State							

Operating State (8 bits)

The thermostat operating state identifier MUST be set according to Table 2.524.

Table 2.524: Thermostat Operating State Report version 2::Operating State encoding

Operating State	Description	Version
0x00	Idle	1
0x01	Heating	1
0x02	Cooling	1
0x03	Fan Only	1
0x04	Pending Heat. Short cycle prevention feature used in heat pump applications to protect the compressor.	1
0x05	Pending Cool. Short cycle prevention feature used in heat pump applications to protect the compressor.	1
0x06	Vent/Economizer	1
0x07	Aux Heating	2
0x08	2nd Stage Heating	2
0x09	2nd Stage Cooling	2
0x0A	2nd Stage Aux Heat	2
0x0B	2nd Stage Aux Heat	2

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.112.3 Thermostat Operating State Logging Supported Get

This command is used to request the operating state logging supported by the device.

The Thermostat Operating State Logging Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_STATE_LOGGING_SUPPORTED_GET (0x01)							

2.2.112.4 Thermostat Operating State Logging Supported Report

This command is used to report the operating state logging supported by the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_LOGGING_SUPPORTED_REPORT (0x04)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields describe the operating state logging supported by the device.

- Bit 0 in Bit Mask 1 is not allocated to any Operating State and MUST beset to zero.
- Bit 1 in Bit Mask 1 indicates if Operating State = 1 (Heating) log is supported.
- Bit 2 in Bit Mask 1 indicates if Operating State = 2 (Cooling) is supported.
- ...

If the Operating State log is supported the bit MUST be set to 1. If the Operating State log is not supported the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported operating state log. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

2.2.112.5 Thermostat Operating State Logging Get

This command is used to request the operating state logging supported by the device.

The Thermostat Operating State Logging Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_STATE_LOGGING_GET (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields describe the operating state log types to be requested.

- Bit 0 in Bit Mask 1 is not allocated to any Operating State and MUST be set to zero.
- Bit 1 in Bit Mask 1 indicates if Operating State = 1 (Heating) log is supported.
- Bit 2 in Bit Mask 1 indicates if Operating State = 2 (Cooling) is supported.
- ...

If the Operating State log is supported the bit MUST be set to 1. If the Operating State log is not supported the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last requested operating state log type. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

2.2.112.6 Thermostat Operating State Logging Report

This command is used to report the operating state logged for requested operating states.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_OPERATING_STATE (0x42)							
Command = THERMOSTAT_OPERATING_STATE_LOGGING_REPORT (0x06)							
Reports to Follow							
Reserved				Operating State Log Type 1			
Usage Today (Hours)							
Usage Today (Minutes)							
Usage Yesterday (Hours)							
Usage Yesterday (Minutes)							
...							
Reserved				Operating State Log Type N			
Usage Today (Hours)							
Usage Today (Minutes)							
Usage Yesterday (Hours)							
Usage Yesterday (Minutes)							

Reports to Follow (8 bits)

This value indicates how many report frames left before transferring all of the requested thermostat operating state logs.

Operating State Log Type (N * 4 bits)

The Operating State Log Type indicates the operating state type to be requested.

Usage Today Hours (8 bits)

The number of hours (00:24) the thermostat has been in the indicated operating state since 12:00 am of the current day.

Usage Today Minutes (8 bits)

The number of minutes (00-59) the thermostat has been in the indicated operating state since 12:00 am of the current day.

Usage Yesterday Hours (8 bits)

The number of hours (00:24) the thermostat had been in the indicated operating state between 12:00 am and 11:59pm of the previous day.

Usage Yesterday Hours (8 bits)

The number of minutes (00-59) the thermostat had been in the indicated operating state between 12:00 am and 11:59pm of the previous day.

2.2.113 Thermostat Setback Command Class, version 1

The Thermostat Setback Command Class is used to change the current state of a non-schedule setback thermostat.

2.2.113.1 Thermostat Setback Set Command

This command is used to set the state of the thermostat.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETBACK (0x47)							
Command = THERMOSTAT_SETBACK_SET (0x01)							
Reserved						Setback Type	
Setback State							

Setback Type (2 bits)

The setback type field MUST comply with Table 2.525.

Table 2.525: Thermostat Setback Set::Setback Type encoding

Value	Description
0x00	No override
0x01	Temporary override
0x02	Permanent override
0x03	Reserved

Note: The temporary override provides an opportunity to implement a timer or equivalent in the device. A temporary override will, if a timer is implemented, be terminated by the timer. If no timer is implemented the temporary override MUST act as permanent override. If the temporary override is implemented it MUST be documented in the user's manual.

Setback State (8 bits)

The Setback State MUST comply with Table 2.526.

Table 2.526: Thermostat Setback Set::Setback State encoding

Setback State		Description
Hexadecimal	Decimal	
0x80	-128	The setback in 1/10 degrees (Kelvin)
...	...	Example:
0xFF	-1	0 = 0 degrees setback
0x00	0	1 = 0.1 degrees is added to the setpoint
0x01	1	2 = 0.2 degrees is added to the setpoint
...	...	-1 = 0.1 degrees is subtracted from the setpoint
0x78	120	-2 = 0.2 degrees is subtracted from the setpoint
0x79	121	Frost Protection
0x7A	122	Energy Saving Mode
0x7B-0x7E	123-126	Reserved
0x7F	127	Unused State

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

When converting between Celsius and Fahrenheit proper rounding MUST be applied with at least two decimals in the internal calculations of a device to avoid rounding errors.

Note: The implementation of Energy Saving Mode is manufacturer specific, and MUST be documented in the User's Manual.

If the device is set to an unreachable state, the device SHOULD assume the closest possible state.

2.2.113.2 Thermostat Setback Get Command

This command is used to request the current state of the thermostat.

The Thermostat Setback Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETBACK (0x47)							
Command = THERMOSTAT_SETBACK_GET (0x02)							

2.2.113.3 Thermostat Setback Report Command

This command is used to report the current state of the thermostat.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETBACK (0x47)							
Command = THERMOSTAT_SETBACK_REPORT (0x03)							
Reserved						Setback Type	
Setback State							

Setback Type (2 bits)

Refer to description under Thermostat Setback Set Command.

Setback State (8 bits)

Refer to description under Thermostat Setback Set Command.

2.2.114 Thermostat Setpoint Command Class, version 1-2

The Thermostat Setpoint Command Class is used to configure setpoints for the modes supported by a thermostat.

2.2.114.1 Interoperability Considerations

It has been found that early implementations of this Command Class specification apply two non-interoperable interpretations of the bit mask advertising the support for specific Setpoint Types.

As a consequence, one may find thermostat products and controller products in the marketplace which implement either of the two bit mask interpretations found in Table 2.527. The notation x.y indicates Bit Mask byte x, bit y.

Implementations of Thermostat Setpoint Command Class, version 1-2 MUST comply with Interpretation A.

2.2.114.2 Thermostat Setpoint Set Command

This command is used to specify the target value for the specified Setpoint Type at a supporting node.

A supporting node MUST support the same format of Precision, Scale and Size fields values as it sends in the Thermostat Setpoint Supported Report Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_SET (0x01)							
Reserved				Setpoint Type			
Precision			Scale		Size		
Value 1							
...							
Value N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field is used to specify the setpoint to be set at the receiving node .

The value MUST comply with Table 2.528.

If a non-supported setpoint type is specified, a receiving node MUST ignore the command.

Precision (3 bits)

This field is used to indicate how many decimal places are included in the Value field.

For example, the Value field set to 1025 with the Precision field set to 2 MUST be interpreted as 10.25.

Scale (2 bits)

This field is used to specify what temperature scale is used for the setpoint value.

The field MUST be encoded according to Table 2.529.

Size (3 bits)

This field is used to indicate the length in bytes of the Value field.

This field MUST be set to 1, 2 or 4.

Value (N bytes)

This field is used to advertise the actual setpoint value to be set at the receiving node.

The length of this field MUST be according to the Size field value.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12.

2.2.114.3 Thermostat Setpoint Get Command

This command is used to request the target value for a given setpoint type that is currently configured at a supporting node.

The Thermostat Setpoint Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_GET (0x02)							
Reserved				Setpoint Type			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field is used to specify which setpoint is requested.

The value MUST comply with Table 2.528.

A supporting node receiving a supported Setpoint Type value MUST be return the same value in the Thermostat Setpoint Report Command.

A supporting node receiving a non-supported Setpoint Type value MUST return the value 0x00 (N/A) in the Thermostat Setpoint Report Command.

2.2.114.4 Thermostat Setpoint Report Command

This command is used to advertise the target value for a given setpoint type that is currently configured at the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_REPORT (0x03)							
Reserved				Setpoint Type			
Precision			Scale		Size		
Value 1							
...							
Value N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field is used to advertise the setpoint advertised by the sending node.

The value MUST comply with Table 2.528.

If this field is set to 0x00 (N/A), it is RECOMMENDED to set the Size field to 1 and the Value field to 0.

Precision (3 bits)

This field is used to indicate how many decimal places are included in the Value field.

For example, the Value field set to 1025 with the Precision field set to 2 MUST be interpreted as 10.25.

Scale (2 bits)

This field is used to specify what temperature scale is used for the setpoint value.

The field MUST be encoded according to Table 2.529.

Size (3 bits)

This field is used to indicate the length in bytes of the Value field.

This field MUST be set to 1, 2 or 4.

Value (N bytes)

This field is used to advertise the actual setpoint value set at the sending node.

The length of this field MUST be according to the Size field value.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12.

2.2.114.5 Thermostat Setpoint Supported Get Command

This command is used to query the supported setpoint types.

The Thermostat Setpoint Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_SUPPORTED_GET (0x04)							

2.2.114.6 Thermostat Setpoint Supported Report Command

This command is used to advertise the supported setpoint types.

This command is known to cause interoperability issues. Refer to Section 2.2.114.1.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

This field advertises the supported Setpoint Types at the sending node.

A supporting node MUST support at least one setpoint type. Setpoint types and their minimum required version are described in Table 2.528. A node MUST NOT support a setpoint type associated to a newer version than the version it supports.

This field MUST be encoded according to Table 2.527, interpretation A:

- Bit 0 in Bit Mask 1 represents Mode = 0x00 (Off).
- Bit 1 in Bit Mask 1 represents Mode = 0x01 (Heat).
- Bit 2 in Bit Mask 1 represents Mode = 0x02 (Cooling).
- Bit 3 in Bit Mask 1 represents Mode = 0x07 (Furnace).
- …

If a Setpoint Type is supported, the corresponding bit MUST be set to '1'.

If a Setpoint Type is not supported, the corresponding bit MUST be set to '0'.

This length of this field MUST be set to the minimum amount of bytes which allows advertising all supported Setpoint Types.

2.2.115 Thermostat Setpoint Command Class, version 3

The Thermostat Setpoint Command Class is used to configure setpoints for the modes supported by a thermostat.

2.2.115.1 Compatibility Considerations

The Thermostat Setpoint Command Class, version 3 is backwards compatible with *Thermostat Setpoint Command Class, version 1-2*.

This version introduces:

- Setpoint capability discovery (allowed value range).
- New setpoint types.

2.2.115.2 Interoperability Considerations

It has been found that early implementations of this Command Class specification apply two non-interoperable interpretations of the bit mask advertising the support for specific Setpoint Types.

As a consequence, one may find thermostat products and controller products in the marketplace which implement either of the two bit mask interpretations found in Table 2.527. The notation x.y indicates Bit Mask byte x, bit y.

Implementations of Thermostat Setpoint Command Class, version 3 MUST comply with Interpretation A.

Table 2.527: Thermostat Setpoint Types Bit Mask encoding

Support Bit Mask Interpretation A	Support Bit Mask Interpretation B	Setpoint Type Identifier	Description	Version
1.0	1.0	0x00	N/A	-
1.1	1.1	0x01	Heating	1
1.2	1.2	0x02	Cooling	1
	1.3	0x03	N/A	-
	1.4	0x04	N/A	-
	1.5	0x05	N/A	-
	1.6	0x06	N/A	-
1.3	1.7	0x07	Furnace	1
1.4	2.0	0x08	Dry Air	1
1.5	2.1	0x09	Moist Air	1
1.6	2.2	0x0A	Auto changeover	1
1.7	2.3	0x0B	Energy Save Heating	2
2.0	2.4	0x0C	Energy Save Cooling	2
2.1	2.5	0x0D	Away Heating	2
2.2	2.6	0x0E	Away Cooling	3
2.3	2.7	0x0F	Full Power	3

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

2.2.115.3 Thermostat Setpoint Set Command

This command is used to specify the target value for the specified Setpoint Type at a supporting node.

A supporting node MUST support the same format of Precision, Scale and Size fields values as it sends in the Thermostat Setpoint Supported Report Command or Thermostat Setpoint Capabilities Report Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_SET (0x01)							
Reserved				Setpoint Type			
Precision			Scale		Size		
Value 1							
...							
Value N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field is used to specify the setpoint to be set at the receiving node.

The value MUST comply with Table 2.528.

Table 2.528: Thermostat Setpoint set::Setpoint Types

Value	Description	CC Version
0x00	N/A	1
0x01	Heating	1
0x02	Cooling	1
0x03	N/A - reserved	-
0x04	N/A - reserved	-
0x05	N/A - reserved	-
0x06	N/A - reserved	-
0x07	Furnace	1
0x08	Dry Air	1
0x09	Moist Air	1
0x0A	Auto Changeover	1
0x0B	Energy Save Heating	2
0x0C	Energy Save Cooling	2
0x0D	Away Heating	2
0x0E	Away Cooling	3
0x0F	Full Power	3

Values marked as not applicable (N/A) MUST be ignored by a receiving node. All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Precision (3 bits)

This field is used to specify how many decimal places are included in the Value field. For example, the decimal value 1025 with precision 2 MUST be interpreted as 10.25.

Scale (2 bits)

This field is used to specify what temperature scale is used for the setpoint value.

The field MUST be encoded according to Table 2.529.

Table 2.529: Thermostat Setpoint Set::Scale Encoding

Value	Scale used in Value field
0	Celsius
1	Fahrenheit

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Size (3 bits)

This field is used to indicate the length in bytes of the Value field.

This field MUST be set to 1, 2 or 4.

Value (N bytes)

This field is used to advertise the actual setpoint value to be set at the receiving node.

The length of this field MUST be according to the Size field value.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12.

2.2.115.4 Thermostat Setpoint Get Command

This command is used to request the target value for a given setpoint type that is currently configured at a supporting node.

The Thermostat Setpoint Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_GET (0x02)							
Reserved				Setpoint Type			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field is used to specify which setpoint is requested.

The value MUST comply with Table 2.528.

A supporting node receiving a supported Setpoint Type value MUST be return the same value in the Thermostat Setpoint Report Command.

A supporting node receiving a non-supported Setpoint Type value MUST return the value 0x00 (N/A) in the Thermostat Setpoint Report Command.

2.2.115.5 Thermostat Setpoint Report Command

This command is used to advertise the value of a setpoint type.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_REPORT (0x03)							
Reserved				Setpoint Type			
Precision			Scale		Size		
Value 1							
...							
Value N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field is used to advertise the setpoint advertised by the sending node.

The value MUST comply with Table 2.528.

If this field is set to 0x00 (N/A), it is RECOMMENDED to set the *Size* field to 1 and the Value field to 0.

Precision (3 bits)

This field is used to indicate how many decimal places are included in the *Value* field.

For example, the Value field set to 1025 with the *Precision* field set to 2 MUST be interpreted as 10.25.

Scale (2 bits)

This field is used to specify what temperature scale is used for the setpoint value.

The field MUST be encoded according to Table 2.529.

Size (3 bits)

This field is used to indicate the length in bytes of the *Value* field.

This field MUST be set to 1, 2 or 4.

Value (N bytes)

This field is used to advertise the actual setpoint value set at the sending node.

The length of this field MUST be according to the *Size* field value.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12.

A receiving node MUST ignore values outside the range advertised in the Thermostat Setpoint Capabilities Report Command for the actual Setpoint type.

2.2.115.6 Thermostat Setpoint Supported Get Command

This command is used to query the supported setpoint types.

The Thermostat Setpoint Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_SUPPORTED_GET (0x04)							

2.2.115.7 Thermostat Setpoint Supported Report Command

This command is used to advertise the supported setpoint types.

This command is known to cause interoperability issues. Refer to Section 2.2.114.1.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_SUPPORTED_REPORT (0x05)							
Bit Mask 1							
...							
Bit Mask N							

Bit Mask (N bytes)

This field advertises the supported Setpoint Types at the sending node.

A supporting node MUST support at least one setpoint type. Setpoint types and their minimum required version are described in Table 2.528. A node MUST NOT support a setpoint type associated to a newer version than the version it supports.

This field MUST be encoded according to Table 2.527, interpretation A:

- Bit 0 in Bit Mask 1 represents Mode = 0x00 (Off).
- Bit 1 in Bit Mask 1 represents Mode = 0x01 (Heat).
- Bit 2 in Bit Mask 1 represents Mode = 0x02 (Cooling).
- Bit 3 in Bit Mask 1 represents Mode = 0x07 (Furnace).
- ...

If a Setpoint Type is supported, the corresponding bit MUST be set to '1'.

If a Setpoint Type is not supported, the corresponding bit MUST be set to '0'.

This length of this field MUST be set to the minimum amount of bytes which allows advertising all supported Setpoint Types.

2.2.115.8 Thermostat Setpoint Capabilities Get Command

This command is used request the supported setpoint value range for an actual Setpoint Type.

The Thermostat Setpoint Capabilities Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_CAPABILITIES_GET (0x09)							
Reserved				Setpoint Type			

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Setpoint Type (4 bits)

This field is used to specify which setpoint is requested.

The value MUST comply with Table 2.528.

A supporting node receiving a supported Setpoint Type value MUST be return the same value in the Thermostat Setpoint Capabilities Report Command.

A supporting node receiving a non-supported Setpoint Type value MUST return the value 0x00 (N/A) in the Thermostat Setpoint Capabilities Report Command.

2.2.115.9 Thermostat Setpoint Capabilities Report Command

This command is used advertise the supported setpoint value range for an actual Setpoint Type.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_THERMOSTAT_SETPOINT (0x43)							
Command = THERMOSTAT_SETPOINT_CAPABILITIES_REPORT (0x0A)							
Reserved				Setpoint Type			
Min Value Precision			Min Value Scale		Min Value Size		
Min Value 1							
...							
Min Value N							
Max Value Precision			Max Value Scale		Max Value Size		
Max Value 1							
...							
Max Value N							

Setpoint Type (4 bits)

This field is used to advertise the setpoint advertised by the sending node.

The value MUST comply with Table 2.528.

If this field is set to 0x00 (N/A), it is RECOMMENDED to set the *Min Value Size* and *Max Value Size* fields to 1 and the *Min Value* and *Max Value* fields to 0.

Min Value Precision (3 bits)

This field is used to indicate how many decimal places are included in the *Min Value* field.

For example, the *Min Value* field set to 1025 with the *Min Value Precision* field set to 2 MUST be interpreted as 10.25.

Min Value Scale (2 bits)

This field is used to specify what temperature scale is used for the setpoint *Min Value* field.

The field MUST be encoded according to Table 2.529.

Min Value Size (3 bits)

This field is used to indicate the length in bytes of the *Min Value* field.

This field MUST be set to 1, 2 or 4.

Min Value (N bytes)

This field is used to advertise the minimum value that is supported for the actual setpoint at the sending node.

The length of this field MUST be according to the *Min Value Size* field value.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12.

Max Value Precision (3 bits)

This field is used to indicate how many decimal places are included in the *Max Value* field.

For example, the *Max Value* field set to 1025 with the *Max Value Precision* field set to 2 MUST be interpreted as 10.25.

Max Value Scale (2 bits)

This field is used to specify what temperature scale is used for the setpoint *Max Value* field.

The field MUST be encoded according to Table 2.529 and this field MUST be set to the same value as the *Min Value Scale* field.

Max Value Size (3 bits)

This field is used to indicate the length in bytes of the *Max Value* field.

This field MUST be set to 1, 2 or 4.

Max Value (N bytes)

This field is used to advertise the maximum value that is supported for the actual setpoint at the sending node.

The length of this field MUST be according to the *Max Value Size* field value.

The first byte MUST be the most significant byte.

This field MUST be encoded using signed representation and comply with Table 2.12.

This field MUST be set to a value greater than the *Min Value* field.

2.2.116 User Code Command Class, version 1

The User Code Command Class is used to manage User Codes in access control systems.

2.2.116.1 Interoperability Considerations

CC:0063.01.00.32.001 A node supporting the Door Lock Command Class SHOULD reflect user code inputs in the door lock status when relevant. (e.g. when the door becomes unlocked by a User Code input, the Door Lock Mode is also updated to unlocked)

This Command Class can be used in conjunction with Schedule Entry Lock Command Class in order to schedule access for users.

CC:0063.01.00.31.001 A node receiving a User Code Get Command MUST advertise what User Code is set in the User Code Report Command.

It has been found that some version 1 nodes wrongfully report obfuscated User Codes in the User Code Report (e.g. '******').

CC:0063.01.00.32.002 A controlling node SHOULD understand that a code has been set correctly but cannot be read back with such nodes.

2.2.116.2 User Code Set Command

This command is used to set a User Code at the receiving node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_SET (0x01)							
User Identifier							
User ID Status							
User Code 1							
...							
User Code N							

User Identifier (8 bits)

This field is used to specify the actual User Identifier.

CC:0063.01.01.11.001 The implemented User Identifier values MUST be in a sequence starting from 1, i.e. a node supporting 10 User Identifiers MUST accept values in the range 1..10.

CC:0063.01.01.13.001 A receiving node MAY ignore this field if it only supports one User Code.

CC:0063.01.01.11.002 The value 0 MUST indicate that the receiving node MUST set the User Code and User Status for all
CC:0063.01.01.12.001 supported User Identifiers. The value 0 SHOULD be used only with User ID Status set to 0x00.

If a non-existing User Identifier (higher than the advertised supported users number in the Users
CC:0063.01.01.11.003 Number Report Command) is specified in this command, a receiving node MUST ignore the command.

User ID Status (8 bits)

CC:0063.01.01.11.004 The User ID Status field indicates the status of the User Identifier. This field MUST comply with Table 2.530.

CC:0063.01.01.12.002 The User ID Status 0xFE SHOULD NOT be used in this command and SHOULD be ignored by receiving nodes.

Table 2.530: User Code Set::User ID Status encoding

Value	Description
0x00	Available (not set)
0x01	Occupied
0x02	Reserved by administrator
0x0FE	Status not available

CC:0063.01.01.11.005 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Code (N bytes)

This field is used to advertise the User Code to be set for the User Identifier.

CC:0063.01.01.11.006 The length of this field MUST be between 4 and 10 bytes. The field's length MUST be determined using the length of the frame.

CC:0063.01.01.11.007 Each byte in this field MUST be a digit encoded with ASCII representation (from 0x30 to 0x39).

CC:0063.01.01.11.008 A node receiving an invalid User Code MUST ignore the command.

CC:0063.01.01.11.009 The User Code field MUST be set to 0x00000000 (4 bytes) when User ID Status is equal to 0x00.

2.2.116.3 User Code Get Command

This command is used to request the User Code of a specific User Identifier.

CC:0063.01.02.11.001 The User Code Report Command MUST be returned in response to this command.

CC:0063.01.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:0063.01.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_GET (0x02)							
User Identifier							

User Identifier (8 bits)

This field is used to specify the requested User Identifier.

CC:0063.01.02.11.004 The value 0 MUST NOT be specified in the User Code Get Command.

CC:0063.01.02.12.001 If the specified User Identifier is not supported, a responding node SHOULD return a User Code Report Command with the User ID Status set to 0xFE "Status not available".

2.2.116.4 User Code Report Command

This command is used to advertise a User Code and its current status.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_REPORT (0x03)							
User Identifier							
User ID Status							
User Code 1							
...							
User Code N							

For fields' description, refer to the *User Code Set Command*.

2.2.116.5 Users Number Get Command

This command is used to request the number of user codes supported by the receiving node.

CC:0063.01.04.11.001 The Users Number Report Command MUST be returned in response to this command.

CC:0063.01.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0063.01.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USERS_NUMBER_GET (0x04)							

2.2.116.6 Users Number Report Command

This command is used to report the number of User Codes supported by the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USERS_NUMBER_GET (0x04)							
Supported Users							

Supported Users (8 bits)

This field is used to advertise the number of supported User Codes.

CC:0063.01.05.11.001 This field MUST be set to the total amount of supported User Codes. The value '0' MUST indicate that no User Code is supported by the node.

2.2.117 User Code Command Class, version 2

The User Code Command Class is used to manage user codes in access control systems.

User Code Command Class, version 2 extends the number of supported users to 65535, introduces new User ID Statuses, keypad modes and a Admin Code functionality.

2.2.117.1 Terminology

User Codes are configured on a supporting node for different **User Identifiers**. Each User Code is associated to one User Identifier. An actual end user with several User Codes will also have several User Identifiers (one for each User Code).

The entry of a user code can trigger different outcomes, such as opening a door, triggering a notification or ignoring the code. This can be configured with the User ID status.

A supporting node may support one or several **Keypad Modes**. They allow configuring the keypad to accept, ignore or treat differently keypad input.

A node may support the **Admin Code** functionality. The Admin Code is a separate unique code used to get access to the node's administrator functionalities, such as the network settings and/or User Code management. A node may require the Admin Code to be always set and in this case advertise that the Admin Code can be modified but cannot be deactivated using Z-Wave.

A node may support the **User Code Checksum** functionality. A controlling node can request a checksum representing all the user code set at the supporting node to ensure that the user code databases are synchronized.

2.2.117.2 Compatibility Considerations

The User Code Command Class, version 2 is backwards compatible with *User Code Command Class, version 1*.

CC:0063.02.00.21.001 All commands and fields not mentioned in this version MUST remain unchanged *User Code Command Class, version 1*.

The Extended User Code Set Command permits to configure several user codes per command. The following command has been extended to match the extended number of supported users:

- *User Code Report Command*

The following commands are also introduced to address the extended range of users:

- Extended User Code Set Command
- Extended User Code Get Command
- Extended User Code Report Command

Commands for advertising node capabilities, configuring the keypad mode and the Admin Code are introduced:

- User Code Capabilities Get Command
- User Code Capabilities Report Command
- User Code Keypad Mode Get Command
- User Code Keypad Mode Report Command
- Admin Code Set Command
- Admin Code Get Command
- Admin Code Report Command

User Code Command Class, version 2 does not extend the existing User Code Set/Get/Report commands from version 1. The first 255 users that can be addressed by the User Code Set/Get Commands
CC:0063.02.00.21.002 MUST be identical to the first 255 users that can be addressed by the (version 2) Extended User Code Set/Get Commands.

CC:0063.02.00.21.003 Addressing User Codes for extended User Identifiers (in the range 256..65535) MUST be done with the (version 2) Extended User Code Set/Get Commands.

CC:0063.02.00.21.004 A version 1 node MAY advertise support for 0 User Codes in the Users Number Report Command.

A version 2 node MUST advertise support for at least 1 User Code.

CC:0063.02.00.21.005 The implemented User Identifier values MUST be in a sequence starting from 1, i.e. a node supporting 10 Users MUST accept User Identifier values in the range 1..10. However, a controlling node MAY assign user codes non-continuously in the range of supported User Identifier.

2.2.117.3 Interoperability Considerations

This Command Class can be used in conjunction with the Entry Control Command Class to report User Code input to a controlling application.

CC:0063.02.00.31.001 A node supporting the "Messaging" User ID Status MUST support the Entry Control Command Class, version 1 or newer or the Notification Command Class, version 8 or newer.

CC:0063.02.00.31.005 A node supporting this Command Class MUST either:

- Support the Door Lock Command Class.
- Control the Door Lock Command Class via association groups.

CC:0063.02.00.31.003 A supporting node MUST reflect user code inputs in the door lock status when relevant. (e.g. when the door becomes unsecured by a User Code input, the Door Lock Operation mode is updated to unsecure with a timeout)

CC:0063.02.00.33.001 A node supporting this Command Class MAY have an Association group issuing the corresponding Door Lock Operation Set Commands when valid User Codes are input in order to control several door locks simultaneously.

This Command Class can be used in conjunction with Schedule Entry Lock Command Class to schedule access for users.

CC:0063.02.00.31.004 A node receiving a User Code Get Command MUST advertise what User Code is set in the User Code Report Command.

2.2.117.4 Security Considerations

A node identifies users on the background of User Code entry alone without User Identifier, the following security measures should be taken into consideration:

CC:0063.02.00.42.001
- Users SHOULD NOT be allowed to choose their own user codes. Users tend to choose codes such as birthdates and therefore the risk of having duplicate codes is high. A user may learn about another User Code when trying to set his own User Code and receiving a duplicate/rejection message. In such a case, User Codes SHOULD be generated randomly by the administrating company, if applicable.

CC:0063.02.00.42.002
- If a node is configured with a large number of User Codes: (e.g. 500), it is RECOMMENDED to increase the number of digits in each and every user code, making is less likely to randomly guess an assigned user code.

CC:0063.02.00.42.003
- It is RECOMMENDED to implement mechanisms preventing brute force attacks, such as exponential increasing waiting time between attempts or lock out after a number of failed attempts.

CC:0063.02.00.41.001 If a node allows updating User Codes or User ID Statuses locally via a user interface using a Admin Code, the supporting node MUST issue User Code Report Commands via the Lifeline to advertise local updates.

2.2.117.5 User Code Set Command

This command is used to set a User Code at the receiving node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_SET (0x01)							
User Identifier							
User ID Status							
User Code 1							
...							
User Code N							

CC:0063.02.01.11.001 All fields not described below MUST remain unchanged from version 1.

User ID Status (8 bits)

The User ID Status field indicates the state of the User Identifier.

CC:0063.02.01.11.005 This field MUST comply with Table 2.532, however the requirements in Table 2.531 MUST take precedence over the requirements in Table 2.532.

CC:0063.02.01.11.006 If a non-supported User ID Status is specified, a receiving node MUST ignore the command.

User Identifier (8 bits)

This field is used to specify the actual User Identifier.

CC:0063.02.01.11.002 The implemented User Identifier values MUST be in a sequence starting from 1, i.e. a node supporting 10 User Codes MUST accept values in the range 1..10.

CC:0063.02.01.11.003 The value 0 MUST indicate that the receiving node MUST set the User Code and User Status for all supported user identifiers less than 256. A receiving node supporting the User Code Command Class, version 2 MUST NOT apply the status to users from the extended range (User ID > 255) when this field is set to 0.

CC:0063.02.01.12.001 The value 0 SHOULD be used only with User ID Status set to 0x00.

CC:0063.02.01.11.004 If a non-existing User Identifier (higher than the advertised supported users number in the Users Number Report Command) is specified in this command, a receiving node MUST ignore the command

2.2.117.6 Users Number Report Command

This command is used to report the number of users that the sending node supports.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USERS_NUMBER_REPORT (0x05)							
Supported Users							
Extended Supported Users 1 (MSB)							
Extended Supported Users 2 (LSB)							

Supported Users (8 bits)

This field is used to advertise the number of supported users for version 1 controlling nodes.

CC:0063.02.05.11.001

This field MUST be in the range 1..255.

CC:0063.02.05.11.002 If the node implements more than 255 users, this field MUST be set to 255.

If the node implements 255 users or less, this field MUST be set to the total amount of supported users.

Extended Supported Users (16 bits)

This field is used to advertise the number of supported users for version 2 controlling nodes.

CC:0063.02.05.11.003 If the node supports less than 256 users, this field MUST be identical to the Supported Users field.

If the node implements 256 users or more, this field MUST be set to the total amount of supported users.

2.2.117.7 User Code Capabilities Get Command

This command is used to request the User Code capabilities of a node.

CC:0063.02.06.11.001 The User Code Capabilities Report Command MUST be returned in response to this command.

CC:0063.02.06.11.002 This command MUST NOT be issued via multicast addressing.

CC:0063.02.06.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_CAPABILITIES_GET (0x06)							

2.2.117.8 User Code Capabilities Report Command

This command is used to advertise User Code capabilities.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_CAPABILITIES_REPORT (0x07)							
AC Support	ACD Support	Res	Supported User ID Status Bit Mask Length				
Supported User ID Status Bit Mask 1							
...							
Supported User ID Status Bit Mask N							
UCC Support	MUCR Support	MUCS Support	Supported Keypad Modes Bit Mask Length				
Supported Keypad Modes Bit Mask 1							
...							
Supported Keypad Modes Bit Mask M							
Reserved			Supported Keys Bit Mask Length				
Supported Keys Bit Mask 1							
...							
Supported Keys Bit Mask L							

Reserved / Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

AC (Admin Code) Support (1 bit)

This field indicates if the sending node supports the Admin Code functionality.

CC:0063.02.07.11.001 The value 1 MUST indicate that the Admin Code functionality is supported.

The value 0 MUST indicate that the Admin Code functionality is not supported.

ACD (Admin Code Deactivation) Support (1 bit)

This field indicates if the sending node supports reporting the Admin Code can be deactivated using a Admin Code Set Command.

CC:0063.02.07.11.002 The value 1 MUST indicate that the Admin Code may be deactivated by issuing a Admin Code Set with the length field set to 0.

The value 0 MUST indicate that the Admin Code cannot be deactivated and Admin Code Set commands with the length field set to 0 will be ignored.

CC:0063.02.07.11.003 This field MUST be set to 0 by a sending node if the AC Support field is set to 0.

UCC (User Code Checksum) Support (1 bit)

This field indicates if the sending node supports the User Code Checksum functionality.

CC:0063.02.07.11.004 The value 1 MUST indicate that the User Code Checksum functionality is supported.

The value 0 MUST indicate that the User Code Checksum functionality is not supported.

MUCR (Multiple User Code Report) Support (1 bit)

This field indicates if the sending node supports reporting the Multiple User Codes at once in a single Extended User Code Report Command. This functionality should be supported by node supporting large amount of User Codes.

CC:0063.02.07.11.005 The value 1 MUST indicate that the Multiple User Code Report functionality is supported.

The value 0 MUST indicate that the Multiple User Code Report functionality is not supported.

MUCS (Multiple User Code Set) Support (1 bit)

This field indicates if the sending node supports being set Multiple User Codes at once in a single Extended User Code Set Command. This functionality should be supported by node supporting large amount of User Codes.

CC:0063.02.07.11.006 The value 1 MUST indicate that the Multiple User Code Set functionality is supported.

The value 0 MUST indicate that the Multiple User Code Set functionality is not supported.

Supported User ID Status Bit Mask Length (5 bits)

CC:0063.02.07.11.007 This field MUST advertise the length in bytes of the *Supported User ID Status Bit Mask* field carried in the command.

CC:0063.02.07.11.008 This field MUST be set to the minimum value which allows advertising all supported User ID Statuses.

Supported User ID Status Bit Mask (N bytes)

CC:0063.02.07.11.009 This field advertises the supported User ID status values. The length of this field in bytes MUST match the value advertised in the *Supported User ID Status Bit Mask Length* field.

A node MUST support the User ID Status values 0x00, 0x01 and 0x02.

- Bit 0 in Bit Mask 1 represents User ID status 0x00.
- Bit 1 in Bit Mask 1 represents User ID status 0x01.
- …

User ID Status values are described in Table 2.532.

CC:0063.02.07.11.00A If a User ID Status is supported, the corresponding bit MUST be set to '1'.

If a User ID Status is not supported, the corresponding bit MUST be set to '0'.

CC:0063.02.07.11.00B All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Supported Keypad Modes Bit Mask Length (5 bits)

CC:0063.02.07.11.00C This field MUST advertise the length in bytes of the *Supported Keypad Modes Bit Mask* field carried in the command.

CC:0063.02.07.11.00D This field MUST be set to the minimum value which allows advertising all supported keypad modes.

Supported Keypad Mode Bit Mask (M bytes)

CC:0063.02.07.11.00E This field describes the supported keypad mode values. The length of this field in bytes MUST match the value advertised in the *Supported Keypad Modes Bit Mask* Length field.

CC:0063.02.07.11.00F A node MUST support the keypad mode 0x00 (normal mode).

- Bit 0 in Bit Mask 1 represents keypad mode 0x00 (normal mode).
- Bit 1 in Bit Mask 1 represents keypad mode 0x01 (vacation mode).
- …

Keypad mode values are described in Table 2.531.

CC:0063.02.07.11.010 If a keypad mode is supported, the corresponding bit MUST be set to '1'.

If a keypad mode is not supported, the corresponding bit MUST be set to '0'.

CC:0063.02.07.11.011 All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Supported Keys Bit Mask Length (5 bits)

CC:0063.02.07.11.012 This field MUST advertise the length in bytes of the *Supported Keys Bit Mask* field carried in the command.

CC:0063.02.07.11.013 This field MUST be set to the minimum value which allows advertising all supported ASCII codes.

Supported Keys Bit Mask (L bytes)

CC:0063.02.07.11.014 This field describes the supported keys that can be input on the keypad by end users. The length of this field in bytes MUST match the value advertised in the *Supported Keys Bit Mask Length* field.

CC:0063.02.07.11.015 This field MUST advertise the ASCII codes that represent the supported keys.

- Bit 0 in Bit Mask 1 represents ASCII code 0-
- Bit 1 in Bit Mask 1 represents ASCII code 1.
- …
- Bit 7 in Bit Mask 16 represents ASCII code 127

CC:0063.02.07.11.016 If an ASCII code is supported, the corresponding bit MUST be set to '1'.

If an ASCII code is not supported, the corresponding bit MUST be set to '0'.

CC:0063.02.07.11.017 All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

CC:0063.02.07.11.018 A supporting node MUST NOT advertise an ASCII code representing the "enter" button on the keypad, but only ASCII codes that can be part of the user codes.

CC:0063.02.07.12.001 A supporting node SHOULD support the ASCII codes 0x30..0x39 (decimal digits 0..9).

2.2.117.9 User Code Keypad Mode Set Command

This command is used to set the keypad mode at the receiving node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_KEYPAD_MODE_SET (0x08)							

Keypad Mode (8 bits)

This field is used to set the keypad mode at the receiving node.

CC:0063.02.08.11.001 This field MUST be encoded according to Table 2.531.

CC:0063.02.08.11.002 A supporting node being set a keypad mode MUST respect the requirements associated to this mode in Table 2.531. The requirements in Table 2.531 (keypad mode) MUST take precedence over the requirements in Table 2.532 (user code statuses).

CC:0063.02.08.11.003 This command MUST be ignored by a receiving node if the specified keypad mode is not supported.

Table 2.531: User Code Keypad Mode Set::Keypad mode encoding

Value	Description	Version
0x00	**Normal Mode**: This mode MUST be the default mode. When the Normal mode is active, the supporting node: • MUST work normally with all defined User Codes according to Table 2.532 • MUST accept the Admin Code, if supported	1
0x01	**Vacation Mode**: This mode is used to disable/switch off the User Code functionalities for normal users. When the Vacation mode is active, the supporting node: • MUST ignore all input of assigned User Code (i.e. with Status ID different than 0x00 **Available**) • MUST react according to Table 2.532 when being input unknown codes (i.e. with Status ID 0x00 **Available**). • MUST accept the Admin Code, if supported	2
0x02	**Privacy Mode**: This mode is used to completely disable/switch off the User Code functionalities. When the Privacy mode is active, the supporting node: • MUST ignore any keypad input • MUST ignore the Admin Code, if supported This mode can be used e.g. to disallow User Code keypad use completely or to rely only on Entry Control Notifications, if supported.	2
0x03	**Locked Out Mode**: This mode is used as a security measure to prevent brute force attacks. This mode is normally activated by the supporting node itself. If the node activates the Locked Out mode, it MUST issue a User Code Keypad Mode Report Command to the Lifeline destination. When the Locked Out mode is active, the supporting node: • MUST ignore any keypad input • MUST ignore the Admin Code, if supported. A supporting node MUST return to the last active mode automatically after a defined timeout if it activated this mode. A controlling node may unlock the keypad again by issuing this command with a different mode.	2

CC:0063.02.08.11.004

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.117.10 User Code Keypad Mode Get Command

This command is used to request a node about its current keypad mode.

CC:0063.02.09.11.001 The User Code Keypad Mode Report Command MUST be returned in response to this command.

CC:0063.02.09.11.002 This command MUST NOT be issued via multicast addressing.

CC:0063.02.09.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_KEYPAD_MODE_GET (0x09)							

2.2.117.11 User Code Keypad Mode Report Command

This command is used by a node to advertise its current keypad mode.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_KEYPAD_MODE_REPORT (0x0A)							
Keypad Mode							

Keypad Mode (8 bits)

This field is used to advertise the keypad mode currently active at the sending node.

CC:0063.02.0A.11.001 This field MUST be encoded according to Table 2.531.

2.2.117.12 Extended User Code Set Command

This command is used to set one or more User Codes in the receiving node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = EXTENDED_USER_CODE_SET (0x0B)							
Number of User Codes							
User Identifier 1 (MSB) 1							
User Identifier 2 (LSB) 2							
User ID Status 1							
Reserved				User Code Length 1			
User Code 1, 1							
...							
User Code N, 1							
...							
User Identifier 1 (MSB) M							
User Identifier 2 (LSB) M							
User ID Status M							
Reserved				User Code Length M			
User Code 1, M							
...							
User Code N, M							

Number of User Codes (8 bits)

This field is used to specify how many user codes blocks are contained in the actual command.

CC:0063.02.0B.11.001 This field MUST be in the range 1..255. A node MUST respect the Z-Wave MAC frame size or Transport service limits when sending this command, which means in most cases that this field SHOULD NOT be set to a value higher than 10.

CC:0063.02.0B.11.002 This field MUST be ignored and considered as set to 1 by a receiving node if it advertises no support for Multiple User Code Set (MUCS) in the User Code Capabilities Report Command

CC:0063.02.0B.11.003 The number of User Code blocks contained in the command MUST be according to this field. A User Code block MUST comprise the following fields:

- User Identifier
- User ID Status
- User Code Length
- User Code

User Identifier (16 bits)

This field is used to specify the User Identifier for the actual User Code block.

CC:0063.02.0B.11.004 The first byte MUST carry the most significant byte of the 16-bit value.

CC:0063.02.0B.11.005 The value 0 MUST indicate that the receiving node MUST set the User Code and User Status for all supported User Identifiers. The value 0 SHOULD be used only with User ID Status set to 0x00 (erasing all User Codes).

CC:0063.02.0B.11.006 If a non-existing User Identifier (higher than the advertised supported users number in the Users Number Report Command) is specified in this command, a receiving node MUST ignore the actual User Code block

User ID Status (8 bits)

CC:0063.02.0B.11.00E

The User ID Status field indicates the state of the User Identifier for the actual User Code block. This field MUST comply with Table 2.532, however the requirements in Table 2.531 MUST take precedence over the requirements in Table 2.532.

CC:0063.02.0B.11.008 If a non-supported User ID Status is specified, a receiving node MUST ignore the actual User Code block.

Table 2.532: Extended User Code Set::User ID Status encoding

Value	Description	Version
0x00	**Available**: The database entry represented by this User Identifier is not in use. This status MUST be used to delete codes or indicate that an identifier is not assigned. When a user enters an unassigned/unknown code, the node: • MUST indicate that the code is not accepted • MUST NOT grant access to this user • SHOULD take preventive actions for further attempts to enter User ID and/or User Code (refer to Section 2.2.117.4)	1
0x01	**Enabled/Grant Access**: The database entry represented by this User Identifier is in use and enabled. When a user enters a code with this status, the node: • MUST indicate that the code is accepted • MUST grant access to the user • MUST secure the Door Lock again after a timeout (either configured if set in Timed Operation or pre-defined if the Door Lock is set in Constant Operation)	1
0x02	**Disabled**: The database entry represented by this User Identifier is in use but disabled. E.g. it allows a controller to push User Codes to a supporting node and let these codes be subsequently activated by a local interface. When a user enters a code with this status, the node: • MUST indicate that the code is not accepted • MUST NOT grant access to the user	1
0x03	**Messaging**: The database entry represented by this User Identifier is in use. The value is used to relay notifications to a controlling application. E.g. A user may enter a messaging code for notifying that maintenance started or an application should activate the alarm system. When a user enters a code with this status, the node: • SHOULD indicate that the code is accepted • MUST NOT grant access to this user • MUST NOT take preventive actions for further attempts to enter User ID and/or User Code (refer to Section 2.2.117.4)	2
0x04	**Passage Mode**: The database entry represented by this User Identifier is in use. The value is used to let the Door Lock permanently grant access. Passage Mode can be activated and deactivated using the Door Lock Operation Set (Unsecured without timeout/Secured) or via Passage Mode User Code input. E.g. A receptionist comes in the morning and enters a Passage Mode User Code to allow anybody to subsequently enter the premises without entering a code until Passage Mode is deactivated again. When a user enters a code with this status, the node: • MUST indicate that the code is accepted • MUST toggle the Door Lock operation state between secured and unsecured without timeout (and also disable any auto-relock functionality) If the node supports the Door Lock Command Class, it: • MUST send a Door Lock Operation Report and Door Lock Configuration Report via the Lifeline Association Group to advertise the new Door Lock state	2

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Reserved

CC:0063.02.0B.11.009 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

User Code Length (4 bits)

This field is used to advertise the length in bytes of the corresponding User Code field in the actual User Code block.

CC:0063.02.0B.11.00A The User Code Length MUST be in the range 4..10 if the User ID Status is different than 0x00.

The User Code Length MUST be set to 0 if the User ID Status is set to 0x00.

User Code (N bytes)

This field is used to advertise the User Code to be set for the User ID for the actual User Code block.

CC:0063.02.0B.11.00B The length of this field in bytes MUST be according to the corresponding User Code Length field value. If the User Code Length is set to 0, this field MUST be omitted.

CC:0063.02.0B.11.00C Each byte in this field MUST be encoded with ASCII representation.

CC:0063.02.0B.11.00D A supporting node receiving a non-supported ASCII character MUST ignore the actual User Code block.

A supporting node receiving a User Code identical to one already set to another User ID or identical to the Admin Code MUST ignore the actual User Code block.

2.2.117.13 Extended User Code Get Command

This command is used to request the User Code of a specific User Identifier.

CC:0063.02.0C.11.001 The Extended User Code Report Command MUST be returned in response to this command

CC:0063.02.0C.11.002 This command MUST NOT be issued via multicast addressing.

CC:0063.02.0C.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = EXTENDED_USER_CODE_GET (0x0C)							
User Identifier 1 (MSB)							
User Identifier 2 (LSB)							
Reserved							Report more

User Identifier (16 bits)

This field is used to specify the requested User Identifier.

CC:0063.02.0C.11.004 The first byte MUST carry the most significant byte of the 16 bit value.

This field MUST NOT be set to 0.

CC:0063.02.0C.11.005 A supporting node MUST return this value in the first User Code block of the returned Extended User Code Report Command.

CC:0063.02.0C.11.006 If a non-existing User Identifier (higher than the advertised supported users number in the Users Number Report Command) is specified in this command, a responding node MUST return an Extended User Code Report Command with the User ID Status set to 0xFE: “Status not available”.

Report more (1 bit)

This field is used to instruct the receiving node to report as many User Codes as possible within a single Z-Wave command (that fits in a single Z-Wave frame) because the sending node intends to read the whole (or a large part of the) User Code database.

CC:0063.02.0C.11.007 This field MUST be ignored by a receiving node if it advertises no support for Multiple User Code Report (MUCR) in the User Code Capabilities Report Command.

CC:0063.02.0C.11.008 The value 0 MUST indicate to return a report for the requested User Identifier only.

The value 1 MUST indicate to return a report for the requested User Identifier and additionally report as many User Identifiers as possible in the response.

When this field is set to 1, a node advertising support for Multiple User Code Report (MUCR) in the User Code Capabilities Report Command:

CC:0063.02.0C.11.009
- MUST return at least 2 User Code blocks in the Extended User Code Report Command unless the last used User Identifier or a non-supported User Identifier is requested.

CC:0063.02.0C.12.001
- SHOULD return as many User Code blocks as possible.

CC:0063.02.0C.11.00A
- Subsequent User Code blocks MUST contain consecutive User Identifier having a User ID Status different than 0. For example, a node supporting 4 User IDs that are set as follow MUST report User Identifier 1, 3, and 4 when requested about User Identifier 1:
  - User ID 1, status 0
  - User ID 2, status 0
  - User ID 3, status 1, code 38473
  - User ID 4, status 4, code 9277

2.2.117.14 Extended User Code Report Command

This command is used to advertise the User Code of a specific User Identifier.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = EXTENDED_USER_CODE_REPORT (0x0D)							
Number of User Codes							
User Identifier 1 (MSB) 1							
User Identifier 2 (LSB) 1							
User ID Status 1							
Reserved				User Code Length 1			
User Code 1, 1							
...							
User Code N, 1							
...							
User Identifier 1 (MSB) M							
User Identifier 2 (LSB) M							
User ID Status M							
Reserved				User Code Length M			
User Code 1, M							
...							
User Code N, M							
Next User Identifier 1 (MSB)							
Next User Identifier 2 (LSB)							

Number of User Codes (8 bits)

This field is used to specify how many user codes blocks are contained in the actual command.

CC:0063.02.0D.11.001

This field MUST be in the range 1..255. A node MUST respect the Z-Wave MAC frame size or Transport service limits when sending this command, which means in most cases that this field SHOULD NOT be set to a value higher than 8.

CC:0063.02.0D.11.002 The number of User Code blocks contained in the command MUST be according to this field. A User Code block MUST comprise the following fields:

- User Identifier
- User ID Status
- User Code Length
- User Code

CC:0063.02.0D.11.003 This field MUST be set to 1 by a node advertising no support for Multiple User Code Report (MUCR) in the User Code Capabilities Report Command.

User Identifier (16 bits)

This field is used to specify the actual User Identifier for the actual User Code block.

CC:0063.02.0D.11.004 The first byte MUST carry the most significant byte of the 16-bit value.

User ID Status (8 bits)

CC:0063.02.0D.11.005 This field indicates the state of the User Identifier for the actual User Code block. This field MUST comply with Table 2.533.

Table 2.533: Extended User Code Report::User ID Status Encoding

Value	Description	Version
0x00	**Available**. Refer to Table 2.532.	1
0x01	**Enabled/Grant Access**. Refer to Table 2.532.	1
0x02	**Disabled**. Refer to Table 2.532.	1
0x03	**Messaging**. Refer to Table 2.532.	2
0x04	**Passage Mode**. Table 2.532.	2
0xFE	**Status not available**. The requested User Identifier is not valid (either 0 or higher than the supported users number in the Users Number Report Command)	1

CC:0063.02.0D.11.006 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Code Length (4 bits)

This field is used to advertise the length in bytes of the User Code field in the actual User Code block.

CC:0063.02.0D.11.007 The User Code Length MUST be in the range 4..10 if the User ID Status is different than 0x00 and 0xFE.

The User Code Length MUST be set to 0 if the User ID Status is 0x00 or 0xFE.

User Code (N bytes)

This field is used to advertise the User Code currently set for the User ID in the actual User Code block.

CC:0063.02.0D.11.008 The length of this field in bytes MUST be according to the corresponding User Code Length field value. If the User Code Length is set to 0, this field MUST be omitted.

CC:0063.02.0D.11.009 Each byte in this field MUST be encoded with ASCII representation.

Next User Identifier (16 bits)

This field is used to specify the next User Identifier in use in the User Code database after the User Identifier advertised in the last User Code block.

CC:0063.02.0D.11.00A

This field MUST be set to the next User Identifier having a User ID Status different than 0. The first byte MUST carry the most significant byte of the 16-bit value.

CC:0063.02.0D.11.00B The value 0 MUST indicate that the actual User Identifier is the last set at the sending node.

CC:0063.02.0D.11.00C If the status field is set to 0xFE for the last User Code block (User Identifier higher than the advertised supported users number in the Users Number Report Command), this field MUST be set to 0.

2.2.117.15 Admin Code Set Command

This command is used to set the Admin Code in the receiving node.

CC:0063.02.0E.11.001 This command MUST be ignored by a receiving node advertising no support for the Admin Code functionality (AC Support) in the User Code Capabilities Report Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = ADMIN_CODE_SET (0x0E)							
Reserved				Admin Code Length			
Admin Code 1							
...							
Admin Code N							

Reserved

CC:0063.02.0E.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Admin Code Length (4 bits)

This field is used to advertise the length in bytes of the Admin Code field in the command.

CC:0063.02.0E.11.003 This field MUST be set to 0 or in the range 4..10.

CC:0063.02.0E.11.004 Values in the range 4..10 MUST indicate that the receiving node MUST set the Admin Code as indicated in the Admin Code field.

CC:0063.02.0E.11.005 The Value 0 MUST indicate that the receiving node MUST deactivate the Admin Code.

If this field is set to 0, this command MUST be ignored by a receiving node if it advertises no support for Admin Code Deactivation (ACD) in the User Code Capabilities Report Command.

Admin Code (N bytes)

This field is used to advertise the Admin Code to be set for the node.

CC:0063.02.0E.11.006 The length of this field MUST be according to the Admin Code Length field value.

Each byte in this field MUST be encoded with ASCII representation.

CC:0063.02.0E.11.007 A supporting node receiving a non-supported ASCII character MUST ignore the command.

A supporting node receiving a Admin Code identical to one already set to another User ID MAY ignore the command.

When the Admin Code is input, the node:

CC:0063.02.0E.11.008
- MUST indicate that the code is accepted

CC:0063.02.0E.11.009
- MUST grant access to administrator functionalities, such as the network settings and/or User Code management interface.

Requirements in Table 2.531 (keypad modes) MUST take precedence over the above Admin Code requirements.

2.2.117.16 Admin Code Get Command

This command is used to request the Admin Code currently set at the receiving node.

CC:0063.02.0F.11.001 The Admin Code Report Command MUST be returned in response to this command

CC:0063.02.0F.11.002 This command MUST NOT be issued via multicast addressing.

CC:0063.02.0F.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = ADMIN_CODE_GET (0x0F)							

2.2.117.17 Admin Code Report Command

This command is used to advertise the Admin Code currently set at the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = ADMIN_CODE_REPORT (0x10)							
Reserved				Admin Code Length			
Admin Code 1							
...							
Admin Code N							

Reserved

CC:0063.02.10.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Admin Code Length (4 bits)

This field is used to advertise the length in bytes of the Admin Code field in the command.

CC:0063.02.10.11.002 This field MUST be in the range 4..10 if the Admin Code is set and advertised in this command.

CC:0063.02.10.11.003 This field MUST be set to 0 if the Admin Code is deactivated or if the sending node advertises no support for the Admin Code functionality (AC Support) in the User Code Capabilities Report Command.

Admin Code (N bytes)

This field is used to advertise the Admin Code currently set at the sending node.

CC:0063.02.10.11.004 The length of this field MUST be according to the Admin Code Length field value. If the Admin Code Length is set to 0, this field MUST be omitted.

CC:0063.02.10.11.005 Each byte in this field MUST be encoded with ASCII representation.

2.2.117.18 User Code Checksum Get Command

This command is used to request a User Code checksum representing all the User Codes currently set at the receiving node.

CC:0063.02.11.11.001 This command MUST be ignored by a node advertising no support for the User Code Checksum functionality (UCC Support) in the User Code Capabilities Report Command.

CC:0063.02.11.11.002

The User Code Checksum Report Command MUST be returned in response to this command if the User Code Checksum functionality (UCC Support) is advertised as supported in the User Code Capabilities Report Command.

CC:0063.02.11.11.003 This command MUST NOT be issued via multicast addressing.

CC:0063.02.11.11.004 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_CHECKSUM_GET (0x11)							

2.2.117.19 User Code Checksum Report Command

This command is used to advertise the current User Code checksum representing all the User codes set at the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CODE (0x63)							
Command = USER_CODE_CHECKSUM_REPORT (0x12)							
User Code Checksum 1 (MSB)							
User Code Checksum 2 (LSB)							

User Code Checksum (16 bits)

This field is used to advertise the checksum representing all User Codes set at the sending node.

CC:0063.02.12.11.001 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). Refer to *CRC-CCITT Source Code* for details.

CC:0063.02.12.11.002 This field MUST be set to 0x0000 if no User Code is set at node.

CC:0063.02.12.11.003 The checksum data MUST be built by concatenating defined User Codes data (with User ID Status different than 0) in the ascending User ID order. The Admin Code MUST NOT be part of the checksum calculation.

CC:0063.02.12.11.004 Each defined User Code data MUST be formatted as follows:

User ID (16 bits) | User ID Status (8 bits) | User Code (4..10 bytes)

For example, a node supporting 4 User IDs that are set as follow:

- User ID 1, status 4, code 9277
- User ID 2, status 0
- User ID 3, status 1, code 88473
- User ID 4, status 0

CC:0063.02.12.11.005 In this case, User Code data for User ID 1 and 3 MUST be concatenated to obtain the checksum data:

0x0001 | 0x04 | 0x39 32 37 37 | 0x0003 | 0x01 | 0x38 38 34 37 33

CC:0063.02.12.11.006 The checksum data MUST be: 0x00010439323737000301383834373 3.

The returned User Code Checksum field MUST be set to: 0xEAAD.

2.2.118 User Credential Command Class, version 1 [NEVER CERTIFIED]

> **Warning: THIS COMMAND CLASS HAS NEVER BEEN CERTIFIED**
>
> This command class has never been implemented and certified by a Z-Wave product. Therefore, this Command Class definition MAY be updated in a non-backwards compatible manner, or even removed.
>
> Consult with the Z-Wave Alliance Application Work Group if you consider implementing this Command Class.

The User Credential Command Class is used to manage user Credentials for gaining access to properties, typically through unlocking or opening doors. Those properties could be residential or commercial. Credential types include PIN Codes, Passwords, Radio-Frequency Identification (RFID), Bluetooth Low Energy (BLE), Near Field Communication (NFC), Ultra-Wideband (UWB), and Biometrics. This command class employs a user-centric model, allowing multiple Credentials to be associated with one User Unique Identifier.

CC:0083.01.00.11.000 Setting schedules for User Unique Identifiers is not required. However, if setting schedules for Unique User Identifiers is supported, then they MUST be supported using the *Schedule Entry Lock Command Class, version 4 [NEVER CERTIFIED]* or higher.

CC:0083.01.00.11.001 To prompt notifications based on activity for these User Unique Identifiers/Credentials, the Notification Command Class MUST be used.

CC:0083.01.00.12.002 When a node is first enrolled in a Z-Wave network, the controlling device SHOULD query the node for any existing User/Credential information before setting up new User Unique Identifiers. It is possible that nodes have already been populated with desired User Unique Identifiers before network inclusion using some other home automation technology or using a local interface.

2.2.118.1 Terminology

Credentials are configured on a supporting node for different **User Unique Identifiers**. Each User Unique Identifier is associated with none, one, or multiple Credentials. A Credential can only have one user.

A User Unique Identifier is used in the database to associate Credentials and schedules to a single User Unique Identifier slot within the node and Z-Wave network.

A Credential can be identified by a combination of its Credential Type and Credential Slot Number.

CC:0083.01.00.11.002 Once a Credential is assigned a Slot Number, this slot is considered occupied and MUST NOT be reused until the Credential is deleted or moved to a different slot.

The entry of a Credential can trigger different outcomes, such as opening a door, sending a command to an associated device, triggering a notification, or ignoring the Credential. This can be configured with the User Type. A **User Type** determines the category of access allowed.

CC:0083.01.00.13.001 A node MAY support the **Admin Code** functionality. The Admin Code is a separate unique PIN code that exists outside of the User and Credential databases, used to get access to the node's administrator functions, such as the network settings and/or User and Credential management. This functionality is also referred to as **AC Support**.

Note: The term 'Admin PIN Code' is also used in this document for compatibility reasons and is completely interchangeable with the term 'Admin Code'.

CC:0083.01.00.13.002 A node MAY require the Admin Code to be always set and in this case advertise that the Admin Code can be modified but cannot be deactivated using Z-Wave. This functionality is referred to as **ACD Support** (Admin Code Deactivation).

CC:0083.01.00.11.003 When ACD is supported, the Admin Code MUST be able to be deactivated using Z-Wave.

2.2.118.2 Compatibility Considerations

2.2.118.2.1 User Code Command Class Dependencies

CC:0083.01.00.23.007 For backwards compatibility considerations, a device MAY support both the User Code Command Class and User Credential Command Class.

In this case the data backing User Code Command Class is defined to be a user code credential table for numeric PIN codes.

If the User Code Command Class is supported (no matter the version):

CC:0083.01.00.21.004 - The *Credential Set Command* and *Credential Get Command* / *Credential Report Command* for Credential Type 0x01 (PIN Code) MUST act on the same data as the User Code Command Class.

CC:0083.01.00.21.014 - The User Credential Command Class's Credential Data Length for Credential Type 0x01 (PIN Code) MUST have Min value of 4 and Max value of 10.

CC:0083.01.00.21.018 - The User Credential Command Class MUST report the actual Credential Data for Credential Type 0x01 (PIN Code), i.e. Credential Read Back (CRB) MUST be set to 1.

If User Code Command Class, version 1 only is supported:

CC:0083.01.00.21.015 - The User Credential Command Class's number of supported Credential Slots for Credential Type 0x01 (PIN Code) MUST NOT exceed 255.

CC:0083.01.00.21.017 - The User Credential Command Class MUST NOT support User Types "Duress User" nor "Disposable User".

If User Code Command Class, version 2 or higher is supported:

CC:0083.01.00.21.016 - The User Code Command Class MUST support all numeric keys (decimal digits 0..9, i.e. ASCII 0x30..0x39) and MUST NOT support any other key.

If Admin Code functionality is supported along with User Code Command Class, ver. 2 or higher:

CC:0083.01.00.21.018 - Admin Code data manipulated through User Credential Command Class MUST be exactly reflected in User Code Command Class and vice versa.

CC:0083.01.00.21.020 Data MUST be mapped such that User Code Command Class's User Identifier (or Extended User Identifier) is equivalent to User Credential Command Class's Credential Slot, such that modifications using either API modify the same underlying credential.

CC:0083.01.00.21.021 The User Credential (Type PIN Code) / User Code mapping MUST be in accordance with Table 2.534 and Table 2.535.

CC:0083.01.00.21.006 When the User Code Command Class is used to add a new User Identifier (Slot), the device MUST also add a new User Unique Identifier to tie the new credential to.

For example, a Credential Set Command modify for User Unique Identifier 0x0001, Credential Type 0x01 (PIN Code), and Credential Slot 0x0007 will modify the same data as a User Code Set for User Identifier (Slot) 0x07 or Extended User Code Set for User ID (Slot) 0x0007.

CC:0083.01.00.22.008 It is RECOMMENDED for a controlling device to use one common method for controlling a node supporting both User Credential Command Class and User Code Command Class.

Table 2.534: Mapping User Credential (Type PIN Code) to User Code

User Credential CC		**User Code CC v1**	**User Code CC v2**
Credential Set		**User Code Report**	**(Ext.) User Code Report**
		User ID Status (v1)	*User ID Status (v2)*
PIN Code credential is deleted or does not exist.		0x00 Available	
User Set – for the user the credential belongs to		**User Code Report**	**(Ext.) User Code Report**
User Active State	*User Type*	*User ID Status (v1)*	*User ID Status (v2)*
0x00 Occupied Disabled	any	0x02 Reserved by administrator	0x02 Disabled
0x01 Occupied Enabled	0x00 General User	0x01 Occupied	0x01 Enabled / Grant Access
	0x03 Programming User		
	0x04 Non-Access User	0x02 Reserved by administrator	0x03 Messaging
	0x05 Duress User	0x01 Occupied	0x01 Enabled / Grant Access
	0x06 Disposable User		
	0x07 Expiring User		
	0x09 Remote Only User		

Table 2.535: Mapping User Code to User Credential (Type PIN Code)

User Code CC v1	**User Code CC v2**	**User Credential CC**		
User Code Set	**(Ext.) User Code Set**	**Credential Report / User Report**		
User ID Status (v1)	*User ID Status (v2)*			
0x00 Available		PIN Code credential is deleted. But belonging user is unchanged.		
User Code Set	**(Ext.) User Code Set**	**User Report** – of the user the credential belongs to		
User ID Status (v1)	*User ID Status (v2)*	*User Active State*	*User Type*	
			If the credential existed and is modified via the User Code CC	*Else, if the credential is newly added via the User Code CC*
0x01 Occupied	0x01 Enabled / Grant Access	0x01 Occupied Enabled	Belonging user's type is unchanged.	0x00 General User
0x02 Reserved by administrator	0x02 Disabled	0x00 Occupied Disabled		
n.a.	0x03 Messaging	0x01 Occupied Enabled	0x04 Non-Access User	
n.a.	0x04 Passage Mode		Belonging user's type is unchanged.	0x00 General User
0xFE Status not available[1]	n.a.	0x00 Occupied Disabled		

[1] User Code CC v1 User ID Status 0xFE SHOULD be ignored. Mapping MUST NOT be considered if ignored.

2.2.118.2.2 Schedule Entry Lock Command Class Dependencies

CC:0083.01.00.23.009 This Command Class MAY be used in conjunction with the *Schedule Entry Lock Command Class, version 4 [NEVER CERTIFIED]* to schedule access for users.

2.2.118.2.3 Notification Command Class Dependencies

CC:0083.01.00.21.010 This Command Class MUST be used with the *Notification Command Class, version 3-8*.

2.2.118.2.4 Door Lock Command Class Dependencies

CC:0083.01.00.21.011 A node supporting this Command Class MUST either:

- Support the *Door Lock Command Class, version 4*.
- Control the Door Lock Command Class via association groups.

CC:0083.01.00.21.012 A supporting node MUST reflect Credential inputs in the door lock status when relevant. For example, when the node becomes unsecured by a Credential input, the Door Lock Operation mode is updated to unsecure.

CC:0083.01.00.23.013 A node supporting this Command Class MAY have an Association group issuing the corresponding Door Lock Operation Set Commands when valid Credentials are input to control several door locks simultaneously.

2.2.118.3 Security Considerations

On top of those referenced in *User Code Command Class, version 2*, the following security measures should be taken into consideration:

CC:0083.01.00.42.014 - Numeric PIN codes SHOULD NOT be allowed if they contain solely consecutive digits (e.g. 123456 or 654321).

CC:0083.01.00.42.015 - Numeric PIN codes SHOULD NOT be allowed if there is solely one repeating digit (e.g. 111111).

CC:0083.01.00.41.016 - Numeric PIN codes MUST be at least 4 digits long.

CC:0083.01.00.41.017 - If a Credential is rejected, and the *Credential Set Command* was sent with Supervision Encapsulation, the *Supervision Report Command* MUST have a status of FAIL.

2.2.118.4 User Capabilities Get Command

This command is used to request User capabilities and limits from the node.

CC:0083.01.01.11.000 This command MUST NOT be issued via multicast addressing.

CC:0083.01.01.11.001 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast Node ID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

CC:0083.01.01.11.002 The *User Capabilities Report Command* MUST be returned in response to this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_CAPABILITIES_GET (0x01)							

2.2.118.5 User Capabilities Report Command

This command is used to report User capabilities and limits from the node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_CAPABILITIES_REPORT (0x02)							
Number of supported User Unique Identifiers (MSB)							
Number of supported User Unique Identifiers (LSB)							
Supported Credential Rules Bit Mask							
Max Length of User Name							
User Schedule Support	All Users Checksum Support	User Checksum Support	Reserved				
Supported User Types Bit Mask Length							
Supported User Types Bit Mask 1							
...							
Supported User Types Bit Mask N							

Number of supported User Unique Identifiers (16 bits)

This field advertises the maximum number of Users supported by the node. Refer to *User Set Command* for details.

CC:0083.01.02.11.000 This field MUST NOT be set to zero.

Supported Credential Rules Bit Mask (8 bits)

This field advertises the supported Credential Rules at the sending node. Refer to Table 2.541 for details.

CC:0083.01.02.11.001 A supporting node MUST support at least one Credential Rule.

CC:0083.01.02.11.002 This field MUST be encoded according to the following interpretation:

CC:0083.01.02.11.003
- Bit 0 in Bit Mask MUST be set to 0.
- Bit 1 in Bit Mask represents Credential Rule = 0x01 (Single).
- Bit 2 in Bit Mask represents Credential Rule = 0x02 (Dual).
- Bit 3 in Bit Mask represents Credential Rule = 0x03 (Triple).

CC:0083.01.02.11.004 If a Credential Rule is supported, the corresponding bit MUST be set to '1'.

CC:0083.01.02.11.005 If a Credential Rule is not supported, the corresponding bit MUST be set to '0'.

CC:0083.01.02.11.006 The bits set MUST comply with Table 2.541.

Max Length of User Name (8 bits)

This field advertises the maximum number of bytes of the User Name field supported by the node. Refer to *User Set Command* for details.

CC:0083.01.02.11.019 The value MUST be even in order to fit possible UTF-16 encoding.

CC:0083.01.02.13.007 If a User Name is received which is longer than the maximum allowed length, it MAY be truncated to fit.

CC:0083.01.02.11.008 A node MUST NOT send a User Name longer than the Max Length of User Name reported in the User Capabilities report.

User Schedule Support (1 bit)

This field indicates if the sending node supports scheduling access for a User Unique Identifier via the *Schedule Entry Lock Command Class, version 4 [NEVER CERTIFIED]* or higher.

CC:0083.01.02.11.009

The value 1 MUST indicate that the User Schedule functionality is supported.

CC:0083.01.02.11.010 The value 0 MUST indicate that the User Schedule functionality is not supported.

All Users Checksum Support (1 bit)

This field indicates if the sending node supports the All Users Checksum functionality. Refer to *All Users Checksum Get Command* for details.

CC:0083.01.02.11.011 The value 1 MUST indicate that the All Users Checksum functionality is supported.

CC:0083.01.02.11.012 The value 0 MUST indicate that the All Users Checksum functionality is not supported.

CC:0083.01.02.12.013 If only one checksum is going to be supported, it is RECOMMENDED that the All Users Checksum functionality be supported over *User Checksum Get Command* and *Credential Checksum Get Command*.

User Checksum Support (1 bit)

This field indicates if the sending node supports the User Checksum functionality. Refer to *User Checksum Get Command* for details.

CC:0083.01.02.11.014 The value 1 MUST indicate that the User Checksum functionality is supported.

CC:0083.01.02.11.015 The value 0 MUST indicate that the User Checksum functionality is not supported.

Reserved (5 bits)

CC:0083.01.02.11.016 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

Supported User Types Bit Mask Length (8 bits)

This field advertises the number of bytes in the following Supported User Types Bit Mask. Refer to Table 2.538 for details.

Supported User Types Bit Mask (N bytes)

This field advertises the supported User Types at the sending node. Refer to Table 2.538 for details.

CC:0083.01.02.11.017 A supporting node MUST support at least one User Type.

CC:0083.01.02.11.018 This field MUST be encoded according to the following interpretation:

- Bit 0 in Bit Mask 1 represents User Type = 0x00 (General User).
- Bit 3 in Bit Mask 1 represents User Type = 0x03 (Programming User).
- Bit 4 in Bit Mask 1 represents User Type = 0x04 (Non-Access User).
- ...

CC:0083.01.02.11.019 If a User Type is supported, the corresponding bit MUST be set to '1'.

CC:0083.01.02.11.020 If a User Type is not supported, the corresponding bit MUST be set to '0'.

CC:0083.01.02.11.021 The bits set MUST comply with Table 2.538.

2.2.118.6 Credential Capabilities Get Command

This command is used to request Credential capabilities and limits from the node.

CC:0083.01.03.11.000 The *Credential Capabilities Report Command* MUST be returned in response to this command.

CC:0083.01.03.11.001 This command MUST NOT be issued via multicast addressing.

CC:0083.01.03.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast Node ID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_CAPABILITIES_GET (0x03)							

2.2.118.7 Credential Capabilities Report Command

This command is used to report Credential capabilities and limits from the node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_CAPABILITIES_REPORT (0x04)							
Credential Checksum Support	Admin Code Support	Admin Code Deactivation Support	Reserved				
Number of Supported Credential Types							
Credential Type [Type N]							
...							
Credential Type [Type M]							
CL Support [Type N]	Reserved						
...							
CL Support [Type M]	Reserved						
Number of Supported Credential Slots [Type N] (MSB)							
Number of Supported Credential Slots [Type N] (LSB)							
...							
Number of Supported Credential Slots [Type M] (MSB)							
Number of Supported Credential Slots [Type M] (LSB)							
Min Length of Credential Data [Type N]							
...							
Min Length of Credential Data [Type M]							
Max Length of Credential Data [Type N]							
...							
Max Length of Credential Data [Type M]							
CL Recommended Timeout [Type N]							
...							
CL Recommended Timeout [Type M]							
CL Number of Steps [Type N]							
...							
CL Number of Steps [Type M]							
Maximum Credential Hash Length [Type N]							
...							
Maximum Credential Hash Length [Type M]							

Credential Checksum Support (1 bit)

This field indicates if the sending node supports the Credential Checksum functionality. Refer to *Credential Checksum Get Command* for details.

CC:0083.01.04.11.000 The value 1 MUST indicate that the Credential Checksum functionality is supported.

CC:0083.01.04.11.001 The value 0 MUST indicate that the Credential Checksum functionality is not supported.

Admin Code Support (1 bit)

This field indicates if the node supports an Admin Code (AC). Refer to *Terminology* for details.

CC:0083.01.04.11.013 The value 1 MUST indicate that the Admin Code functionality is supported.

CC:0083.01.04.11.014 The value 0 MUST indicate that the Admin Code functionality is not supported.

Admin Code Deactivation Support (1 bit)

This field indicates if the node supports Admin Code Deactivation (ACD). Refer to *Terminology* for details.

CC:0083.01.04.11.015 The value 1 MUST indicate that the Admin Code Deactivation functionality is supported.

CC:0083.01.04.11.016 The value 0 MUST indicate that the Admin Code Deactivation functionality is not supported.

Reserved (5 bits)

CC:0083.01.04.11.002 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

Number of Supported Credential Types (8 bits)

This field advertises the number of supported Credential Types.

CC:0083.01.04.11.012 This field MUST be followed by the following eight blocks of this number each:

- Credential Type (8 bits * Number of Supported Credential Types)
- Credential Learn Support flag and 7 Reserved bits (8 bits * Number of Supported Credential Types)
- Number of Supported Credential Slots for Type (16 bits * Number of Supported Credential Types)
- Min Length of Credential Data per Slot for Type fields (8 bits * Number of Supported Credential Types)
- Max Length of Credential Data per Slot for Type fields (8 bits * Number of Supported Credential Types)
- Credential Learn (CL) Recommended Timeout (8 bits * Number of Supported Credential Types)
- Credential Learn (CL) Number of Steps (8 bits * Number of Supported Credential Types)
- Maximum Credential Hash Length (8 bits * Number of Supported Credential Types)

Credential Type (8 bits)

This field advertises a supported Credential Type by this node.

CC:0083.01.04.11.004 This field MUST comply with values listed in Table 2.546.

Credential Learn (CL) Support (1 bit)

This field indicates if the sending node supports the Credential Learn functionality for that specific Credential Type.

CC:0083.01.04.11.005 The value 1 MUST indicate that the Credential Learn functionality is supported for that specific Credential Type.

CC:0083.01.04.11.006 The value 0 MUST indicate that the Credential Learn functionality is not supported for that specific Credential Type.

Reserved (7 bits)

CC:0083.01.04.11.007 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

Number of Supported Credential Slots (16 bits)

This field advertises the total number of supported Credential Slots for the given Credential Type. Refer to *Credential Set Command* for details.

CC:0083.01.04.13.008 Those Credential Slots MAY be distributed amongst Users unequally.

CC:0083.01.04.11.009 This field MUST NOT be set to zero.

Min Length of Credential Data (8 bits)

This field advertises the minimum number of bytes of the Credential Data per Slot for the given Credential Type. Refer to *Credential Set Command* for details.

CC:0083.01.04.11.013 This field MUST NOT be set to zero.

CC:0083.01.04.11.014 If UTF-16 encoding is used, the value MUST be even (e.g. applies for password credentials).

Max Length of Credential Data (8 bits)

This field advertises the maximum number of bytes of the Credential Data per Slot for the given Credential Type. Refer to *Credential Set Command* for details.

CC:0083.01.04.11.015 The value MUST NOT be less than the Min Length of Credential Data.

CC:0083.01.04.11.016 If UTF-16 encoding is used, the value MUST be even (e.g. applies for password credentials).

CC:0083.01.04.11.017 For all credential types where CRB is set to 1 the value MUST be chosen so that the node is able to actually report these credentials (e.g. fit the Z-Wave frame). Refer to the CRB flag of the *Credential Report Command*. For PIN codes also refer to the *User Code Command Class Dependencies*.

Credential Learn (CL) Recommended Timeout (8 bits)

CC:0083.01.04.11.010 This field indicates the recommended timeout for each step in the credential learn flow in seconds. If this credential type cannot be learned, this field MUST be 0. If this credential type can be learned, this field MUST NOT be 0.

Credential Learn (CL) Number of Steps (8 bits)

CC:0083.01.04.11.011 This field indicates the number of steps required in the credential learn flow. If this credential type cannot be learned, this field MUST be 0. If this credential type can be learned, this field MUST NOT be 0.

Maximum Credential Hash Length (8 bits)

This indicates the maximum length in bytes of a credential hash for a given credential type.

CC:0083.01.04.11.012 If credentials of a given credential type cannot be read back, then this value MUST be greater than 0, and the hashed credential data MUST be present in the *Credential Report Command*.

CC:0083.01.04.11.013 If credentials of a given credential type can be read back, then this value MUST be 0.

2.2.118.8 User Set Command

This command is used to add, delete, or modify Users in the node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_SET (0x05)							
Reserved						Operation Type	
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
User Type							
Reserved							User Active State
Credential Rule							
Expiring Timeout Minutes (MSB)							
Expiring Timeout Minutes (LSB)							
Reserved						User Name Encoding	
User Name Length							
User Name							

Reserved (6 bits)

CC:0083.01.05.11.000 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

Operation Type (2 bits)

The Operation Type field indicates the event to change User data.

CC:0083.01.05.11.001 This field MUST comply with Table 2.537.

CC:0083.01.05.11.002

Table 2.537: User Set::Operation Type

Value	Name	Definition
0x00	Add	A user with this User Unique Identifier is added.
0x01	Modify	A user with this User Unique Identifier is modified. The fields in this command MUST overwrite existing user data unless the data is rejected as invalid.
0x02	Delete	A user with this User Unique Identifier is deleted. Other fields aside from the User Unique Identifier and Delete Operation Type in this command MUST be ignored. Other fields aside from the User Unique Identifier and Delete Operation Type MAY be set to default values or not included at all. All Credentials and schedules associated with this user MUST also be deleted.

CC:0083.01.05.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Unique Identifier (16 bits)

The User Unique Identifier is used to recognize the user identity across the devices in a single network. This is a sixteen bit value that is supplied by the controller at the time of User Unique Identifier creation. Or, this value is specified by the node at creation time if the User Unique Identifier is locally generated via the node's local interface.

CC:0083.01.05.11.004 A User Unique Identifier MUST be any number in the range 1-65535.

CC:0083.01.05.13.005 A node MAY only support fewer than 65535 Users, as advertised in the *User Capabilities Report Command*.

CC:0083.01.05.11.006 Zero is an invalid User Unique Identifier and MUST NOT be used by the node unless in a *User Report Command* which is being returned in response to a *User Get Command* containing a User Unique Identifier of 0, and there are no existing users at the node.

CC:0083.01.05.11.007 Zero MUST be ignored by the receiving node unless received with the Delete Operation Type, in which case all users and their associated credentials and schedules MUST be deleted.

CC:0083.01.05.11.008 If the User Unique Identifier is set to 0, all other fields aside from the Operation Type MUST be ignored, and those other fields MAY be set to default values or not included at all.

CC:0083.01.05.11.009 To clear all Users and their associated Credentials and schedules, the User Unique Identifier MUST be set to 0 with the Delete Operation Type.

CC:0083.01.05.11.010 If all users are deleted, a *User Report Command* with User Report Type "User(s) Deleted" (0x02) and User Unique Identifier of 0x0000 MUST be returned to the sender and sent to the lifeline.

CC:0083.01.05.11.011 If a user is added, a *User Report Command* with User Report Type "User Added" (0x00) MUST be returned to the sender and sent to the lifeline.

CC:0083.01.05.11.012 If a user is modified, a *User Report Command* with User Report Type "User Modified" (0x01) MUST be returned to the sender and sent to the lifeline.

CC:0083.01.05.11.013 If a single user is deleted, a *User Report Command* with User Report Type "User(s) Deleted" (0x02) MUST be returned to the sender and sent to the lifeline, even if that user is the last one on the node.

CC:0083.01.05.11.014 If a User Unique Identifier is specified in this command as greater than the max advertised in the *User Capabilities Report Command*, a receiving node MUST ignore the command.

CC:0083.01.05.11.015 If a User Unique Identifier is specified in this command as zero for an Add or Modify Operation Type, a receiving node MUST ignore the command

CC:0083.01.05.11.016

Any Credential added to the node MUST be associated with a User Unique Identifier.

CC:0083.01.05.11.017 There MUST NOT be a limit to the number of credentials that can be assigned to a single User Unique Identifier.

CC:0083.01.05.12.018 However, all credentials SHOULD NOT be put on a single user.

User Type (8 bits)

CC:0083.01.05.11.019 The User Type determines the category of access given to a user. It is used to assign the user when created or modified. This field MUST comply with Table 2.538.

CC:0083.01.05.11.020

Table 2.538: User Set::User Type

Numeric Value (*User Set Command*)	Bit Value (*User Capabilities Report Command*)	Name	Definition
0x00	0x01	General User	User has access, provided proper Credential is supplied.
0x01	0x02	Reserved	N/A.
0x02	0x04	Reserved	N/A.
0x03	0x08	Programming User	User has the ability to both program and MAY operate the node. This user can manage the users and user schedules. In all other respects the user matches the general (default) user. Programming User is the only user that can disable the user interface (keypad, remote, etc.). This can also be referred to as admin user.
0x04	0x10	Non-Access User	User is recognized by the node but does not have the ability to open the node. This user will only cause the node to generate the appropriate event notification to any associated device. When a non-access event is generated, a *Notification Report Command* with Notification Type "Access Control" (0x06), Notification Event "Non-Access credential entered via local interface" (0x33), and Event/State parameters containing only *Credential Usage Data* MUST be sent.
0x05	0x20	Duress User	User has the ability to open the node but a *Notification Report Command* with Notification Type "Emergency Alarm" (0x0A) and Notification Event "Panic Alert" (0x04) MUST also be sent to the Lifeline association group when a Credential is used to alert emergency services or contacts.
0x06	0x40	Disposable User	User has the ability to open the node once after which the node MUST change the corresponding user record User Active State value to Occupied Disabled automatically by the node. The User Active State MUST NOT be set to Occupied Disabled if there is an enabled, inactive schedule attached to the user.
0x07	0x80	Expiring User	User has the ability to open the node for a specified time in Expiring Timeout Minutes after the first use of the PIN code, RFID code, Fingerprint, or other Credential. After the time has elapsed, the User Active State value MUST be set to Occupied Disabled automatically by the node. The node MUST persist the timeout across reboots such that the specified time is honored. If this user has an enabled schedule attached to it, the Expiring Timeout Minutes countdown MUST NOT start unless the credential is entered during that schedule's active period.
0x08	0x01 (second byte)	Reserved	N/A.
0x09	0x02 (second byte)	Remote Only User	User access and PIN code is restricted to remote lock/unlock commands only. This type of user might be useful for regular delivery services to prevent a PIN code Credential created for them from being used at the keypad. The PIN code Credential would only be provided over-the-air for the lock/unlock commands.

CC:0083.01.05.11.021 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Table 2.539: Credential Usage Data

Byte
User Unique Identifier (MSB)
User Unique Identifier (LSB)
Number of Credentials
Credential Type [Credential N]
Credential Slot Number (MSB) [Credential N]
Credential Slot Number (LSB) [Credential N]
...
Credential Type [Credential M]
Credential Slot Number (MSB) [Credential M]
Credential Slot Number (LSB) [Credential M]

CC:0083.01.05.11.022 All User Types except the Programming User (0x03) MUST be allowed to be assigned a schedule.

CC:0083.01.05.11.023 Programming User (0x03) MUST NOT be scheduled.

CC:0083.01.05.11.024 If a User Type is updated from a different User Type to Programming User (0x03) and the user has an associated schedule, the schedule MUST be deleted automatically by the node.

CC:0083.01.05.11.025 If a user has a schedule and it is enabled but inactive, the node MUST NOT allow the user to access the node with its assigned Credential(s) and MAY instead send a *Notification Report Command* with Notification Type "Access Control" (0x06), Notification Event "Valid credential access denied due to the User's schedule being inactive" (0x30), and Event/State parameters containing only *Credential Usage Data* MUST be sent.

CC:0083.01.05.11.026 If a user has a schedule and it is disabled, the node MUST allow the user 24/7 permanent access to the node with its assigned Credential(s).

CC:0083.01.05.11.027 If a user's credential is used to lock/close the node where the User Type is not Non-Access User, a *Notification Report Command* with Notification Type "Access Control" (0x06), Notification Event "Credential lock/close operation" (0x23), and Event/State parameters containing only *Credential Usage Data* MUST be sent.

CC:0083.01.05.11.028 If a user's credential is used to unlock/open the node where the User Type is not Non-Access User, a *Notification Report Command* with Notification Type "Access Control" (0x06), Notification Event "Credential unlock/open operation" (0x24), and Event/State parameters containing only *Credential Usage Data* MUST be sent.

CC:0083.01.05.11.029 If a Non-Access User Type's credential is entered at the node and that user's access is valid (User Active State is Occupied Enabled and access is not restricted via schedule at the time the credential is entered), a *Notification Report Command* with Notification Type "Access Control" (0x06), Notification Event "Non-Access credential entered via local interface" (0x33), and Event/State parameters containing only *Credential Usage Data* MUST be sent.

CC:0083.01.05.11.030 If a User Type is specified in this command as one not advertised as supported in the *User Capabilities Report Command*, a receiving node MUST ignore the command.

Reserved (7 bits)

CC:0083.01.05.11.031 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

User Active State (1 bit)

The User Active State determines whether a user is allowed to access the node or not with its assigned Credentials. It is used to assign the user when created or modified.

CC:0083.01.05.11.032 This field MUST comply with Table 2.540.

Table 2.540: User Set::User Active State

Value	Name	Definition
0x00	Occupied Disabled	Access not Granted
0x01	Occupied Enabled	Access Granted

CC:0083.01.05.11.033 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0083.01.05.11.034 If a User Active State is set to Occupied Disabled, the node MUST NOT allow the user to access the node with its assigned Credential(s) and MAY instead send a *Notification Report Command* with Notification Type "Access Control" (0x06), Notification Event "Valid credential access denied due to User Active State being set to Occupied Disabled" (0x2F), and Event/State parameters containing only *Credential Usage Data* MUST be sent.

Credential Rule (8 bits)

The Credential Rule is used to identify multi-factor authentication for the user.

CC:0083.01.05.11.035 The Credential Rule enumeration used in various commands MUST indicate the Credential rule that can be applied to a particular user.

CC:0083.01.05.11.036 This field MUST comply with Table 2.541.

CC:0083.01.05.11.037

Table 2.541: User Set::Credential Rule

Numeric Value (*User Set Command*)	Bit Value (*User Capabilities Report Command*)	Name	Definition
0x01	0x02	Single	Only one Credential is REQUIRED for node operation.
0x02	0x04	Dual	Any two Credentials are REQUIRED for node operation.
0x03	0x08	Triple	Any three Credentials are REQUIRED for node operation.

CC:0083.01.05.11.038 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0083.01.05.11.039 If a Credential Rule is specified in this command as one not advertised as supported in the *User Capabilities Report Command*, a receiving node MUST ignore the command.

CC:0083.01.05.11.040 If a user enters at least one but not enough credentials for the User's Credential Rule, the node MUST NOT allow the user to access the node and MAY instead send a *Notification Report Command* with Notification Type "Access Control" (0x06), Notification Event "User access denied due to not enough credentials entered for the User's Credential Rule" (0x31), and Event/State parameters containing only *Credential Usage Data* for each entered credential MUST be sent.

Expiring Timeout Minutes (16 bits)

CC:0083.01.05.11.041 If the User Type is Expiring User, then this is the time, in minutes, the user is able to use their Credentials before User Active State value MUST be automatically set to Occupied Disabled.

CC:0083.01.05.11.042 If the User Type is Expiring User, this field MUST be set to a non-zero value.

CC:0083.01.05.11.053 For all other User Types, this field MUST be set to zero and MUST be ignored by a receiving node.

Reserved (5 bits)

CC:0083.01.05.11.043 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

User Name Encoding (3 bits)

CC:0083.01.05.11.044 This field is used to indicate the character encoding for the User Name char field. This field MUST comply with Table 2.542.

Table 2.542: User Name Encoding

Value	Description
0x00	Using standard ASCII codes, see *ASCII Codes* (values 128-255 are ignored)
0x01	Using standard and OEM Extended ASCII codes, see *ASCII Codes*.
0x02	Unicode UTF-16, in big endian order

CC:0083.01.05.11.045 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

User Name Length (8 bits)

This field specifies the length of the following User Name field, in bytes.

CC:0083.01.05.12.046 The User Name SHOULD be no longer than "Max Length of User Name" specified in the *User Capabilities Report Command*.

CC:0083.01.05.13.047 If the name is longer, it MAY be truncated.

CC:0083.01.05.11.054 If the name causes the full z-wave command not to respect the Z-Wave MAC frame size or Transport service limits, it MUST be truncated.

User Name (Variable Length)

This field specifies the User's name.

CC:0083.01.05.13.048 Names are human readable identifiers that MAY be shared between multiple sources controlling the same node.

CC:0083.01.05.11.049 The User Name MUST be consistent with the given encoding scheme and length.

CC:0083.01.05.11.050 The User Name is not NULL-terminated. User Name Length MUST be used to determine a name's length.

CC:0083.01.05.12.051 If the User Name is not provided, it SHOULD be set to the default value.

CC:0083.01.05.12.052 The default value SHOULD be "User-[UniqueUserIdentifier]" where [UniqueUserIdentifier] is the decimal User Unique Identifier without padding (e.g. User-7, User-275).

2.2.118.9 User Get Command

This command is used to retrieve data about a specified User.

CC:0083.01.06.11.000 The *User Report Command* MUST be returned in response to this command.

CC:0083.01.06.11.001 This command MUST NOT be issued via multicast addressing.

CC:0083.01.06.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast Node ID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_GET (0x06)							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							

User Unique Identifier (16 bits)

For this field's description, refer to the *User Set Command*.

CC:0083.01.06.11.003 If the User Unique Identifier is zero, then the first existing User Unique Identifier MUST be returned in the *User Report Command*.

CC:0083.01.06.11.004

If the specified User Unique Identifier does not exist, then a responding node MUST return a *User Report Command* with the User Modifier Type set to 0x00 “DNE”.

CC:0083.01.06.11.005 If the requested User Unique Identifier was 0, indicating that the first existing User Unique Identifier be returned, but there are no existing Users at the node, then a responding node MUST return a *User Report Command* with the User Modifier Type set to 0x00 “DNE” and the User Unique Identifier set to 0.

2.2.118.10 User Report Command

This command returns the User data.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_REPORT (0x07)							
User Report Type							
Next User Unique Identifier (MSB)							
Next User Unique Identifier (LSB)							
User Modifier Type							
User Modifier Node ID (MSB)							
User Modifier Node ID (LSB)							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
User Type							
Reserved							User Active State
Credential Rule							
Expiring Timeout Minutes (MSB)							
Expiring Timeout Minutes (LSB)							
Reserved						User Name Encoding	
User Name Length							
User Name							

For “User Unique Identifier”, “User Type”, “User Active State”, “Credential Rule”, “Expiring Timeout Minutes”, “User Name Encoding”, “User Name Length”, and “User Name” descriptions, refer to the *User Set Command*.

User Report Type (8 bits)

This field indicates the reason for sending the User Report message.

This field can be an error when setting User data has failed due to the current state of the node. This helps provide more information to the controller but is not meant to provide errors for bad data being sent, such as an unsupported User Unique Identifier. Commands that are never valid will be ignored.

CC:0083.01.07.11.004 This field MUST comply with Table 2.543 and Table 2.544.

Table 2.543: User Report::User Report Types Table 1

Value	Name	Definition	Report Data	Report Destination
0x00	User Added	A new user was added.	The new data.	Return to sender and send to lifeline.
0x01	User Modified	An existing user was modified.	The new data.	Return to sender and send to lifeline.
0x02	User(s) Deleted	A single or bulk user deletion was performed.	The User Unique Identifier from the request MUST be in the report back. If the User Unique Identifier was zero, zero MUST be in the report. If a non-zero User Unique Identifier deletion was requested, this report MUST contain the deleted data.	Return to sender and send to lifeline.
0x03	User Unchanged	A user add/modify action was performed where no data changed.	The data already occupying the User Unique Identifier.	Return to sender but do not send to lifeline.
0x04	Response to Get	This report is in response to a User Get request.	The data at the node.	Return to sender but do not send to lifeline.
0x05	User Add Rejected Location Occupied	A user add operation is rejected due to the User Unique Identifier already being occupied. If attempting to add a user where a user at that User Unique Identifier already exists, and the new user data differs, the Add operation MUST be rejected and this command MUST be sent instead. If an Add is rejected, a Modify operation MAY be used instead. If all fields in the User Add command match the data at the node, a "User Unchanged" (0x03) User Report Type MUST be returned instead.	The data already occupying the User Unique Identifier.	Return to sender but do not send to lifeline.

Table 2.544: User Report::User Report Types Table 2

Value	Name	Definition	Report Data	Report Destination
0x06	User Modify Rejected Location Empty	A user modify operation is rejected due to the User Unique Identifier location being empty. If attempting to modify a user where a user at that User Unique Identifier does not exist, the Modify operation MUST be rejected and this command MUST be sent instead. If a Modify is rejected, an Add MAY be used instead.	The rejected data.	Return to sender but do not send to lifeline.
0x07	Zero Expiring Minutes Invalid	A user add/modify command contained zero Expiring Timeout Minutes where the node's current state does not allow this. This MUST be returned when a User Set command is sent with Expiring User Type and zero Expiring Timeout Minutes where the current User Type is Non-Expiring. Other combinations of User Type and Expiring Minutes should either result in different User Report Types or no response at all.	The data already occupying the User Unique Identifier.	Return to sender but do not send to lifeline.

CC:0083.01.07.11.005 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0083.01.07.11.007 If a User is specified in a User Set command for an Add or Modify operation for a User Unique Identifier where that User data is already located at the specified User Unique Identifier, the receiving node MUST send a User Report Command with User Report Type "User Unchanged" (0x03).

Next User Unique Identifier (16 bits)

The Next User Unique Identifier is used to iterate through all Users in the User database.

CC:0083.01.07.11.000 This MUST be a non-zero value if there is at least one occupied entry after the requested User Unique Identifier in the User database and MUST be zero if there are no more occupied entries.

An example of a using the Next User Unique Identifier to interview Users and Credentials is given in Figure 2.27.

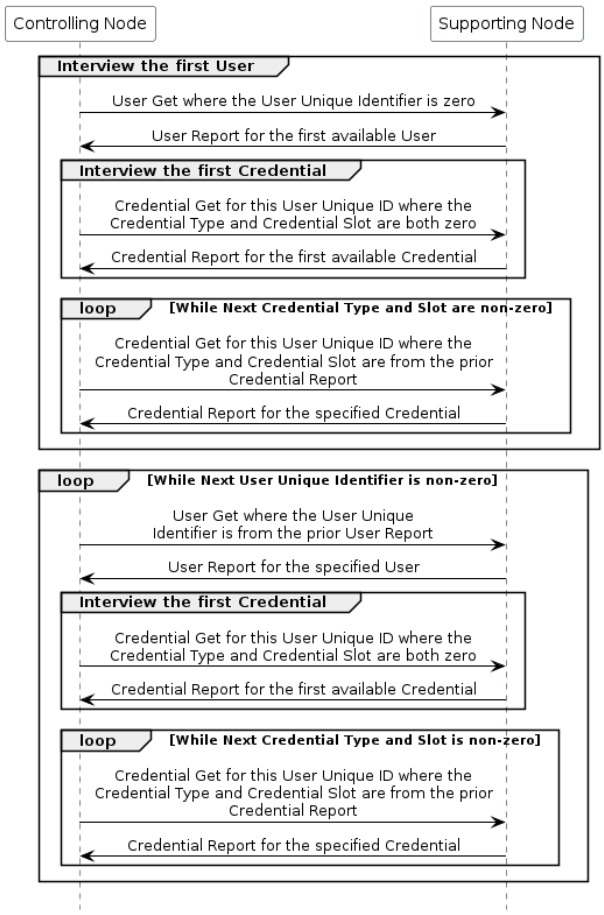


Figure 2.27: Using Next User Unique Identifier to interview Users

User Modifier Type (8 bits)

User Modifier Type indicates how this user was last modified on the node.

CC:0083.01.07.13.001 This MAY be used to audit all users on a node.

CC:0083.01.07.11.002 This field MUST comply with Table 2.545.

Table 2.545: User Set::User Modifier Types

Value	Name	Definition
0x00	DNE	The user does not exist and therefore has no modifier
0x01	Unknown	The user was added via an unknown source
0x02	Z-Wave	The user was added with a Z-Wave command
0x03	Locally	The user was added locally, such as by using a keypad or admin card
0x04	Mobile App or other IoT technology	The user was added with a mobile app without a Z-Wave connection (BLE, Wi-Fi, etc.)

CC:0083.01.07.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0083.01.07.11.006 If no user data is modified, the “User Modifier Type” and “User Modifier Node ID” MUST remain unchanged.

User Modifier Node ID (16 bits)

The User Modifier Node ID indicates which specific source last changed this user on the node. When the User Modifier Type is Z-Wave, this field is the node ID of the specific node which changed the user. For all other User Modifier Types this field’s definition is manufacturer specific.

2.2.118.11 Credential Set Command

This command adds, removes, or modifies Credentials in a node for an existing user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_SET (0x0A)							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
Credential Type							
Credential Slot (MSB)							
Credential Slot (LSB)							
Reserved						Operation Type	
Credential Length							
Credential Data							

User Unique Identifier (16 bits)

See *User Set Command* for this field's description.

Credential Type (8 bits)

CC:0083.01.0A.11.000 The type of this specific Credential. This field MUST comply with Table 2.546.

Table 2.546: Credential Type Encoding

Value	Name	Definition
0x00	None	Reserved for Next Credential Type in Credential Report
0x01	PIN Code	A numeric code typed in at the node
0x02	Password	Alpha-numeric code typed in at the node
0x03	RFID Code	A numeric code entered over a Radio Frequency ID connection
0x04	BLE	A numeric code entered over a Bluetooth Low energy connection
0x05	NFC	A numeric code entered over a Near Field Communication connection
0x06	UWB	A numeric code entered over an Ultra-Wideband connection
0x07	Eye Biometric	Biometric data relating to the eye(s) such as an iris, retina, or scleral vein which is scanned at the node
0x08	Face Biometric	Biometric data relating to the face which is scanned at the node
0x09	Finger Biometric	Biometric data relating to the finger(s) such as finger geometry or a fingerprint which is scanned at the node
0x0A	Hand Biometric	Biometric data relating to the hand(s) such as a palm print or hand geometry which is scanned at the node
0x0B	Unspecified Biometric	Biometric data that does not fall into other biometric types specified in this Credential Type which is scanned at the node

Biometric examples referenced from [18].

CC:0083.01.0A.11.001 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0083.01.0A.11.002 If a Credential Type is specified in this command as one not advertised as supported in the *Credential Capabilities Report Command*, a receiving node MUST ignore the command.

CC:0083.01.0A.11.020 PIN Codes MUST be transmitted in ASCII format.

CC:0083.01.0A.11.021 Passwords MUST be transmitted in Unicode UTF-16 format, in big endian order.

Credential Slot (16 bits)

The Credential Slot is a sixteen bit address used to store Credentials.

CC:0083.01.0A.11.003 The range of values MUST be one to Number of Supported Credential Slots for the given Credential Type, inclusive (see *Credential Capabilities Report Command*). Zero is an invalid Credential Slot unless being using in a Delete operation.

CC:0083.01.0A.11.004 If a Credential Slot is specified in this command as one not advertised as supported in the *Credential Capabilities Report Command* for that Credential Type, a receiving node MUST ignore the command.

CC:0083.01.0A.11.005 If a Credential Slot of zero is specified in this command for a non-Delete Operation Type, a receiving node MUST ignore the command.

Credential Slots are unique per Credential Type. For example, Credential Type 2, Slot 1 is distinct from Credential Type 3, Slot 1.

Reserved (6 bits)

CC:0083.01.0A.11.006 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

Operation Type (2 bits)

The Operation Type field indicates the event to change Credential data.

CC:0083.01.0A.11.007 This field MUST comply with Table 2.547.

CC:0083.01.0A.11.008

Table 2.547: Credential Set::Operation Type

Value	Name	Definition
0x00	Add	A Credential for the given User Unique Identifier of the given Credential Type is added to the given Credential Slot.
0x01	Modify	A Credential for the given User Unique Identifier is modified. The fields in this command MUST overwrite existing user data unless the new data is determined to be invalid.
0x02	Delete	One or more Credentials are deleted. Other fields aside from the User Unique Identifier, Credential Type, Credential Slot and Delete Operation Type in this command MUST be ignored. Those MAY be set to default values or not included at all. If the Credential Type and Credential Slot are non-zero, then that Credential MUST be deleted if the User Unique Identifier matches the belonging user or is set to zero. The receiving node MUST then send a *Credential Report Command* with Credential Report Type "Credential Deleted" (0x02) with the deleted data to the sender. If the node's state changed, the report MUST also be sent to the lifeline. If the User Unique Identifier and Credential Type are non-zero and Credential Slot is zero, then all Credentials of that Credential Type for that User Unique Identifier MUST be deleted. The receiving node MUST then send a *Credential Report Command* with Credential Report Type "Credential Deleted" (0x02) with the User Unique Identifier, Credential Type, and Credential Slot echoed back to the sender. If the node's state changed, the report MUST also be sent to the lifeline. If the User Unique Identifier is non-zero and Credential Type is zero, then all Credentials of all Credential Types for that User Unique Identifier MUST be deleted. The receiving node MUST then send a *Credential Report Command* with Credential Report Type "Credential Deleted" (0x02) with the User Unique Identifier, Credential Type, and Credential Slot echoed back to the sender. If the node's state changed, the report MUST also be sent to the lifeline. If the User Unique Identifier, Credential Type, and Credential Slot are zero, then all Credentials for all Users MUST be deleted. The receiving node MUST then send a *Credential Report Command* with Credential Report Type "Credential Deleted" (0x02) with the User Unique Identifier, Credential Type, and Credential Slot echoed back to the sender. If the node's state changed, the report MUST also be sent to the lifeline. If the User Unique Identifier and Credential Slot are zero, and Credential Type is non-zero, then everything for that Credential Type MUST be deleted. The receiving node MUST then send a *Credential Report Command* with Credential Report Type "Credential Deleted" (0x02) with the User Unique Identifier, Credential Type, and Credential Slot echoed back to the sender. If the node's state changed, the report MUST also be sent to the lifeline.

CC:0083.01.0A.11.009 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be

ignored by a receiving node.

CC:0083.01.0A.11.010 If all credentials are deleted, a *Credential Report Command* with Credential Report Type "Credential Deleted (0x02) MUST be returned to the sender and sent to the lifeline.

CC:0083.01.0A.11.011 If a credential is added, a *Credential Report Command* with Credential Report Type "Credential Added" (0x00) MUST be returned to the sender and sent to the lifeline.

CC:0083.01.0A.11.012 If a credential is modified, a *Credential Report Command* with Credential Report Type "Credential Modified" (0x01) MUST be returned to the sender and sent to the lifeline.

CC:0083.01.0A.11.013 If a credential is deleted, a *Credential Report Command* with Credential Report Type "Credential Deleted" (0x02) MUST be returned to the sender and sent to the lifeline.

Credential Length (8 bits)

Credential Length is the length of the following Credential data, in bytes.

CC:0083.01.0A.11.014 The length of the Credential data MUST be no longer than the Max Length of Credential Data for the specific Credential Type specified in the *Credential Capabilities Report Command*.

CC:0083.01.0A.11.015 The length of the Credential data MUST be no shorter than the Min Length of Credential Data for the specific Credential Type specified in the *Credential Capabilities Report Command*.

CC:0083.01.0A.13.016 This MAY be zero when deleting Credentials.

Credential Data (Variable Length)

The Credential data to set for the Credential being added or modified.

CC:0083.01.0A.11.018 Duplicate credentials within a Credential Type MUST NOT be allowed.

If a node also supports Admin Code: .. raw:: latex

requirement{CC:0083.01.0A.11.020}{0}

The DuplicateCredential Credential Report Type requirement detailed in Table 2.549 MUST also apply to a received Credential Set command that meets the following conditions: - The Credential Set command has Credential Type PIN Code. - The PIN Code credential data is a duplicate of the Admin Code.

CC:0083.01.0A.13.019 If a credential is used at the node that does not match any existing credential, a *Notification Report Command* with Notification Type "Access Control" (0x06) and Notification Event "Invalid credential used to access the node" (0x32) MAY be sent.

2.2.118.12 Credential Get Command

This command is used to retrieve Credential data for a given User Unique Identifier, type, and slot.

CC:0083.01.0B.11.000 The *Credential Report Command* MUST be returned in response to this command.

CC:0083.01.0B.11.001 This command MUST NOT be issued via multicast addressing.

CC:0083.01.0B.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast Node ID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_GET (0x0B)							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
Credential Type							
Credential Slot (MSB)							
Credential Slot (LSB)							

For fields' description, refer to the *Credential Set Command*.

This command can be used to request a specific credential associated with a specific slot of a specific credential type:

CC:0083.01.0B.11.003 If the User Unique Identifier, Credential Type, and Credential Slot are all non-zero, then the Credential associated with that user, type, and slot MUST be returned.

CC:0083.01.0B.11.004 If no Credential exists for that user or type or slot (of that type), a Credential with length zero MUST be returned.

CC:0083.01.0B.11.008 If the User Unique Identifier is zero and the Credential Type and Credential Slot are both non-zero, then the Credential associated with that type and slot MUST be returned.

CC:0083.01.0B.11.009 If no Credential exists for type or slot (of that type), a Credential with length zero MUST be returned.

This command can also be used to request the first credential associated with a user or a credential type:

CC:0083.01.0B.11.010 If the User Unique Identifier is non-zero, the Credential Type is non-zero and the Credential Slot is zero, then the first available Credential for that User Unique Identifier and that type MUST be returned.

CC:0083.01.0B.11.005 If the User Unique Identifier is non-zero and the Credential Type and Credential Slot are both zero, then the first available Credential for that User Unique Identifier MUST be returned.

CC:0083.01.0B.11.007 If the User Unique Identifier is zero, the Credential Type is non-zero and the Credential Slot is zero, then the first available Credential of that type MUST be returned.

CC:0083.01.0B.11.011 If the User Unique Identifier, the Credential Type and the Credential Slot are all zero, then the first available Credential MUST be returned.

CC:0083.01.0B.11.012 If no Credential exists for the user and type combination requested, a Credential with length zero MUST be returned.

CC:0083.01.0B.11.006 Other combinations of zero and non-zero User Unique Identifier, Credential Type, and Credential Slot are considered invalid and MUST be ignored.

2.2.118.13 Credential Report Command

This command reports data about a Credential on a node for an existing user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_REPORT (0x0C)							
Credential Report Type							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
Credential Type							
Credential Slot (MSB)							
Credential Slot (LSB)							
CRB	Reserved						
Credential Length							
Credential Data							
Credential Modifier Type							
Credential Modifier Node ID (MSB)							
Credential Modifier Node ID (LSB)							
Next Credential Type							
Next Credential Slot (MSB)							
Next Credential Slot (LSB)							

For “User Unique Identifier”, “Credential Type”, and “Credential Slot” descriptions, refer to the *Credential Set Command*.

CC:0083.01.0C.11.010

When used in a *Credential Get Command*, the non-zero value is used to iterate through credentials associated with a specific user and type. The zero value is used to iterate globally through all credentials associated with a type. However, when responding to a *Credential Get Command*, the User Unique Identifier field MUST be set to the User Unique Identifier of the user associated with this credential.

Credential Report Type (8 bits)

This field indicates the reason for sending the Credential Report message.

This field can be an error when setting Credential data has failed due to the current state of the node. This helps provide more information to the controller but is not meant to provide errors for bad data being sent, such as an unsupported Credential Slot. Commands that are never valid will be ignored.

CC:0083.01.0C.11.011 This field MUST comply with Table 2.548, Table 2.549, and Table 2.550.

Table 2.548: Credential Report::Credential Report Types Table 1

Value	Name	Definition	Report Data	Report Destination
0x00	Credential Added	A new credential was added.	The new data.	Return to sender and send to lifeline.
0x01	Credential Modified	An existing credential was modified.	The new data.	Return to sender and send to lifeline.
0x02	Credential Deleted	A single or bulk credential deletion request was made.	For bulk deletions, the User Unique Identifier, Credential Type, and Credential Slot from the request MUST be echoed back. If the User Unique Identifier, Credential Type, and/or Credential Slot were zero, zero MUST be echoed back for the respective fields. For single deletions, when the Credential Type and Credential Slot are both non-zero in the deletion request, this report MUST contain the actually deleted data. This means, if the User Unique Identifier was set to zero in the Credential Set Command (where Credential Type and Slot were non-zero), the resulting Credential Report Command MUST contain the actual User Unique Identifier the deleted credential belonged to.	Return to sender and send to lifeline.
0x03	Credential Unchanged	A credential add/modify action was performed where no data changed.	The data already occupying the User Unique Identifier, Credential Type, and Credential Slot.	Return to sender but do not send to lifeline.
0x04	Response to Get	This report is in response to a Credential Get request.	The data at the node.	Return to sender but do not send to lifeline.
0x05	Credential Add Rejected Location Occupied	A credential add operation is rejected due to the Credential Type and Credential Slot already being occupied. If attempting to add a credential where a credential of that Credential Type at that Credential Slot already exists, and the new credential data differs, the Add operation MUST be rejected and this command MUST be sent instead. If an Add is rejected, a Modify operation MAY be used instead.	The data already occupying the Credential Type and Credential Slot.	Return to sender but do not send to lifeline.

Table 2.549: Credential Report::Credential Report Types Table 2

Value	Name	Definition	Report Data	Report Destination
0x06	Credential Modify Rejected Location Empty	A credential modify operation for an existing user is rejected due to the Credential Type and Credential Slot location being empty. If attempting to modify a credential where a credential of that Credential Type at that Credential Slot does not exist, the Modify operation MUST be rejected and this command MUST be sent instead. If a Modify is rejected, an Add operation MAY be used instead.	The rejected data.	Return to sender but do not send to lifeline.
0x07	Duplicate Credential	If a duplicate credential is specified in a *Credential Set Command* for a Credential Type where that duplicate is located at a different Credential Slot or attached to a different User Unique Identifier, the receiving node MUST reject the command and instead send this command. If the duplicate is with the Admin Code, the Credential Report Type MUST be Duplicate Admin PIN Code (0x0A) instead.	This report MUST contain the existing credential data. For example, if PIN Code 1234 is at Credential Slot 0x0002 but a *Credential Set Command* contains Credential Slot 0x0003, this command containing the Credential data with Credential Slot 0x0002 MUST be sent. Or, for example, if PIN Code 1234 is at Credential Slot 0x0002 for User Unique Identifier 0x0003 but a *Credential Set Command* contains User Unique Identifier 0x0004 and Credential Slot 0x0002, this command containing the Credential data with User Unique Identifier 0x0003 and Credential Slot 0x0002 MUST be sent. Controllers MAY re-assign credentials using *User Credential Association Set Command*.	Return to sender but do not send to lifeline.
0x08	Manufacturer Security Rules	If a credential is rejected by the node due to not following manufacturer security rules, such as a node not allowing a PIN Code of only repeating digits, this command MUST be sent.	This report MUST contain the rejected data.	Return to sender but do not send to lifeline.

Table 2.550: Credential Report::Credential Report Types Table 3

Value	Name	Definition	Report Data	Report Destination
0x09	Wrong User Unique Identifier	If the Credential Type and Credential Slot are already assigned to a different User Unique Identifier where the credential is not a duplicate, this command MUST be returned. Or, if the Credential Type and Credential Slot are not already assigned to a different User Unique Identifier where the credential is not a duplicate, but the User Unique Identifier is empty, this command MUST be returned with Modifier Type "Does Not Exist" (0x00).	This report MUST contain the existing credential data if the location is assigned to a different user or the rejected data with Modifier Type "Does Not Exist" (0x00) if the User Unique Identifier, Credential Type, and Credential Slot are all empty.	Return to sender but do not send to lifeline.
0x0A	Duplicate Admin PIN Code	If a duplicate PIN Code is specified in a *Credential Set Command* for a PIN Code Credential Type where that duplicate is the Admin Code, the receiving node MUST reject the command and instead send this command.	This report MUST contain the rejected data.	Return to sender but do not send to lifeline.

CC:0083.01.0C.11.012 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0083.01.0C.11.015 If a *Credential Get Command* requests a Credential Type and Credential Slot where they belong to a different User Unique Identifier, the receiving node MUST send a *Credential Report Command* with Credential Report Type "Response to Get" (0x04) with the correct User Unique Identifier specified.

CC:0083.01.0C.11.013 If a credential is specified in this command for an Add or Modify operation for a User Unique Identifier, Credential Type, and Credential Slot where that credential is already located at the specified Credential Slot for the specified User Unique Identifier and Credential Type, the receiving node MUST send a *Credential Report Command* with Credential Report Type "Credential Unchanged" (0x03).

CC:0083.01.0C.11.014 If no credential data is modified, the "Credential Modifier Type" and "Credential Modifier Node ID" MUST remain unchanged.

Credential Read Back (CRB) (1 bit)

This field is set to one if Credential Length and Credential Data contain the actual credential, and to zero if the credential cannot be read back from the device.

CC:0083.01.0C.11.001 The Credential Read Back field MUST always be set to the same value within a credential type.

CC:0083.01.0C.11.021 If the credential can be read back (Maximum Credential Hash Length 0 in the *Credential Capabilities Report Command*), the CRB flag MUST be set to 1. Otherwise (Maximum Credential Hash Length is greater than 0 in the *Credential Capabilities Report Command*), the CRB flag MUST be set to 0.

CC:0083.01.0C.11.020 Credentials of type PIN code MUST be reported by the node (CRB is set to 1) if mapped to the User Code Command Class.

CC:0083.01.0C.12.002 In any case, non-biometric credential types SHOULD be reported by the node (CRB is set to 1).

CC:0083.01.0C.11.000 If the credential cannot be read back (CRB is zero), the node MUST instead report a hash sum of the credential.

This hash sum can be used by the controlling node to track credential changes without revealing the actual Credential Data. It is up to the supporting node to choose the hashing algorithm.

CC:0083.01.0C.12.003 The hashing algorithm SHOULD allow easy detection of Credential updates by either complying with modern hash-function requirements or being a sequence number.

CC:0083.01.0C.12.004 The hashing algorithm SHOULD be one way.

Reserved (7 bits)

CC:0083.01.0C.11.005 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

Credential Length (8 bits)

See *Credential Set Command* for this field's description.

CC:0083.01.0C.11.006 In addition, this length MUST refer to the length of the hash sum of the Credential if Credential Read Back (CRB) is set to zero.

CC:0083.01.0C.11.019 If CRB is zero, the Minimum and Maximum Length of Credential Data as advertised in the *Credential Capabilities Report Command* do not matter. However, the Credential Length MUST NOT be zero.

CC:0083.01.0C.11.023 If the credential cannot be read back, the Credential Length MUST be smaller than or equal to the Maximum Credential Hash Length in the *Credential Capabilities Report Command* for this credential type.

Credential Data (Variable Length)

See *Credential Set Command* for this field's description.

CC:0083.01.0C.11.007 In addition, this data MUST be the hash sum of the Credential if Credential Read Back is set to zero.

Credential Modifier Type (8 bits)

Refer to User Modifier Type in *User Report Command* for this field's description.

Credential Modifier Node ID (16 bits)

Refer to User Modifier Node ID in *User Report Command* for this field's description.

Next Credential Type (8 bits)

The Next Credential Type is used in tandem with the Next Credential Slot to iterate through all credentials for a given user (if User Unique Identifier is non-zero) or a given Credential Type (if User Unique Identifier is zero). It indicates the Credential Type for the next Credential to request when iterating.

CC:0083.01.0C.11.002 This means: If in a valid *Credential Get Command*, the returned data MUST comply with Table 2.551.

Table 2.551: Credential Report::Returned data

Request			Response (current)	Response (next)			Iterate by	Filter by (AND)	
UUID	Type	Slot		Exists	User DNE	Cred.DNE		UUID	Type
0	0	0	First	2nd	N/A	N/A	Type (0,NZ,NZ)		
0	NZ	0	First for type	2nd	N/A	N/A	Type (0,NZ,NZ)		Type
0	NZ	NZ	Requested	Next	N/A	Next after requested	Type (0,NZ,NZ)		Type
NZ	0	0	First for user	2nd	None	N/A	UUID (NZ,NZ,NZ)	UUID	
NZ	NZ	0	First for user and type	2nd	None	N/A	UUID (NZ,NZ,NZ)	UUID	Type
NZ	NZ	NZ	Requested	Next	None	Next after requested	UUID (NZ,NZ,NZ)	UUID	

CC:0083.01.0C.11.022 If the requested user has been non-zero but does not exist, then these Next Credential fields MUST set to zero

CC:0083.01.0C.11.024 If the requested user has been non-zero (no matter the requested type or slot) and exists, then these Next Credential fields MUST advertise the next existing credential for that user after the reported credential (among all credential types, but no credentials of other users).

CC:0083.01.0C.11.025 If the requested user has been zero (no matter the requested type or slot), then these Next Credential fields MUST advertise the next existing credential for that user after the reported credential (among all credential types).

CC:0083.01.0C.11.016 If the requested user has been non-zero and the requested slot has been non-zero but does not exist (for the requested user), then these Next Credential fields MUST advertise the next existing credential for that user after the requested credential (among all credential types, but no credentials of other users).

CC:0083.01.0C.11.017 If the requested user has been zero and the requested slot has been non-zero but does not exist (for all users), then these Next Credential fields MUST advertise the next existing credential after the requested credential (among all credential types despite the user).

CC:0083.01.0C.11.018 This field MUST comply with the values listed in Table 2.546.

CC:0083.01.0C.11.008 This MUST be a non-zero value if there is at least one more credential for this user (or credential type) and MUST be set to 0x00 if there are no more credentials for this user (or credential type).

CC:0083.01.0C.12.018 During the iteration throught credentials using *Credential Get Command* it is RECOMMENDED for the controlling node to use the same value of the requested user (zero or non-zero) as in the first *Credential Get Command* request.

Next Credential Slot (16 bits)

The Next Credential Slot is used in tandem with the Next Credential Type to iterate through all credentials for a given user (if User Unique Identifier is non-zero) or a given Credential Type (if User Unique Identifier is zero). It indicates the Credential Slot for the next Credential to request when iterating. The value can range from one to the maximum number of credentials supported by this type.

CC:0083.01.0C.11.009 This MUST be a non-zero value if there is at least one more credential for this user (or credential type) and MUST be set to zero if there are no more credentials for this user (or credential type).

2.2.118.14 Credential Learn Start Command

This command initiates the local credential learn process on the supporting node.

CC:0083.01.0F.11.000 The *Credential Learn Status Report Command* MUST be returned at the end of the learn process (successful, unsuccessful, or canceled).

CC:0083.01.0F.11.001 If the learn process is successful, the *Credential Report Command* MUST be returned at the end of the learn process.

CC:0083.01.0F.11.002 If a credential is rejected as invalid during the learn process, the same *Credential Report Command* that would be sent if the node received a *Credential Set Command* with that credential MUST be sent. For example, if the credential is a duplicate and rejected for this reason, a *Credential Report Command* with a Credential Report Type of Duplicate Credential (0x07) will be sent.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_LEARN_START (0x0F)							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
Credential Type							
Credential Slot (MSB)							
Credential Slot (LSB)							
Reserved						Operation Type	
Credential Learn Timeout							

For "User Unique Identifier", "Credential Type", and "Credential Slot" descriptions, refer to the *Credential Set Command*.

CC:0083.01.0F.11.003 If a Credential Type is specified in this command as one not advertised as supported in the *Credential Capabilities Report Command*, a receiving node MUST ignore the command.

CC:0083.01.0F.11.004 If a Credential Type is specified in this command as one not advertised as having Credential Learn Support in the *Credential Capabilities Report Command*, a receiving node MUST ignore the command.

CC:0083.01.0F.11.005 If a User Unique Identifier is specified in this command as one not advertised as supported in the *User Capabilities Report Command*, a receiving node MUST ignore the command.

CC:0083.01.0F.11.006 If a Credential Slot for the Credential Type is specified in this command as one not advertised as supported in the *Credential Capabilities Report Command*, a receiving node MUST ignore the command.

Reserved (6 bits)

CC:0083.01.0F.11.007 This field MUST be set to zero by a sending node and MUST be ignored by a receiving node.

Operation Type (2 bits)

The Operation Type field indicates the event to change Credential data.

CC:0083.01.0F.11.008 This field MUST comply with Table 2.552.

CC:0083.01.0F.11.009

Table 2.552: Credential Learn Start::Operation Type

Value	Name	Definition
0x00	Credential Learn Add	A Credential for the given User Unique Identifier of the given Credential Type is added to the given Credential Slot. If a Credential at this index already exists, this operation MUST be rejected and a Credential Learn Modify operation MAY be used instead.
0x01	Credential Learn Modify	A Credential for the given User Unique Identifier of the given Credential Type is modified for the given Credential Slot. If a Credential at this index does not already exist, this operation MUST be rejected and a Credential Learn Add operation MAY be used instead.

CC:0083.01.0F.11.010 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Credential Learn Timeout (8 bits)

Timeout for each credential learn step on the node, in seconds.

CC:0083.01.0F.13.011 The node MAY choose to return a timeout failure earlier than the specified timeout value.

CC:0083.01.0F.11.012 The node MUST NOT continue the learn process after the provided timeout has elapsed.

If no credential was learned and timeout (either provided or less than provided) was reached:

CC:0083.01.0F.11.013
- The node MUST stop the learn process.

CC:0083.01.0F.12.014

- The node SHOULD advertise the user with audio-visual effect that the learn process has ended.

CC:0083.01.0F.11.015

- The node MUST report back to the initiating node the *Credential Learn Status Report Command* command with a timeout status.

This command is from the GET commands family and a successful Supervision Report does not necessarily indicate learn status.

2.2.118.15 Credential Learn Cancel Command

This command cancels the local credential learn process on the supporting node.

CC:0083.01.10.11.000 The *Credential Learn Status Report Command* MUST be returned in response to this command with a Credential Learn Status of "Ended Not Due to Timeout".

CC:0083.01.10.11.001 The learn process MUST end in response to this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_LEARN_CANCEL (0x10)							

2.2.118.16 Credential Learn Status Report Command

This command reports data about a Credential Learn on a node for an existing user.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_LEARN_REPORT (0x11)							
Credential Learn Status							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
Credential Type							
Credential Slot (MSB)							
Credential Slot (LSB)							
Credential Learn Steps Remaining							

For "User Unique Identifier", "Credential Type", and "Credential Slot" descriptions, refer to the *Credential Set Command*

CC:0083.01.11.11.004 This command MUST be sent for each step of the Credential Learn process and on completion of the credential learn.

Credential Learn Status (8 bits)

This field indicates the result of the credential learn process.

CC:0083.01.11.11.000 This field MUST comply with Table 2.553.

CC:0083.01.11.11.001

Table 2.553: Credential Learn Report::Credential Learn Status

Value	Name	Definition
0x00	Started	Credential learn step has started.
0x01	Success	A Credential was learned and stored in the corresponding slot.
0x02	Already in Progress	The node is already in the learning process. The current step of the learn process MUST continue until the original timeout (provided or less than provided) expires.
0x03	Ended Not Due to Timeout	The learn process has ended before the timeout (provided or less than provided) expired. If this status is returned, the learn process MUST end. No credential was learned, and the slot was left untouched. For example, a node could decide to stop the learn process after determining the first credential attempt is invalid, or it could decide to allow additional attempts as part of the same *Credential Learn Start Command*. This status MUST be returned in response to the *Credential Learn Cancel Command*.
0x04	Timeout	A Credential Learn process timed out. No credential was learned, and the slot was left untouched.
0x05	Credential Learn Step Retry	One step of the credential learn process was not completed correctly and MUST be retried.
0xFE	Invalid Credential Learn Add Operation Type	The requested add operation type is invalid. If a Credential Learn Add is requested when a credential already exists at the specified slot, this status MUST be returned. If this status is returned, the learn process MUST NOT be entered. No credential was learned, and the slot was left untouched.
0xFF	Invalid Credential Learn Modify Operation Type	The requested modify operation type is invalid. If a Credential Learn Modify is requested when no credential exists at the specified slot, this status MUST be returned. If this status is returned, the learn process MUST NOT be entered. No credential was learned, and the slot was left untouched.

CC:0083.01.11.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Credential Learn Steps Remaining (8 bits)

CC:0083.01.11.11.003 This field indicates the number of remaining steps in the credential learn process for the learn process to be completed. This field MUST NOT be 0 if Credential Learn Status is Started, Already in Progress, or Credential Learn Step Retry. This field MUST be 0 for all other defined Credential Learn Status values. For the first step in a flow where the Credential Learn Status is Started, this value MUST be the total number of steps required to learn the credential.

2.2.118.17 User Credential Association Set Command

This command is used to associate an existing credential with an updated user, in place of the user it is currently linked to.

For example, a biometric credential might be added directly to a lock using a lock's local interface. Those limited local interfaces can make it challenging to associate the newly added biometric credential with an already existing user, and as a result the lock may create a new user entry for the credential. That user may later want to use an interface on their controller to move that biometric credential over to a previously existing user (that may have been programmed through the controller, and have existing credentials like an RFID card and a PIN Code). They can use this command to accomplish

that goal. At the end of this process, the credential will no longer be associated with its starting user.

CC:0083.01.12.11.000 A node receiving this command MUST respond by sending a *User Credential Association Report Command*.

CC:0083.01.12.11.001 This command MUST NOT be issued via multicast addressing.

CC:0083.01.12.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast Node ID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_CREDENTIAL_ASSOCIATION_SET (0x12)							
Credential Type							
Source Credential Slot (MSB)							
Source Credential Slot (LSB)							
Destination User Unique Identifier (MSB)							
Destination User Unique Identifier (LSB)							
Destination Credential Slot (MSB)							
Destination Credential Slot (LSB)							

CC:0083.01.12.11.003 The Destination User Unique Identifier MUST be non-zero, and MUST reference an existing User Unique Identifier.

CC:0083.01.12.11.004 The Credential Type MUST be one supported by the receiving device.

CC:0083.01.12.11.005 The Source Credential Slot MUST have an existing credential in it, and the Destination Credential Slot MUST NOT have an existing credential in it.

Credential Type (8 bits)

This field advertises the Credential Type of the Credential that will be associated with a different User Unique Identifier.

CC:0083.01.12.11.006 This field MUST comply with values listed in Table 2.546.

Source Credential Slot (16 bits)

This field advertises the Credential Slot of the Credential that will be associated with a different User Unique Identifier.

Destination User Unique Identifier (16 bits)

The Destination User Unique Identifier is used to identify the User Unique Identifier that will be linked to the credential after the operation of this command.

CC:0083.01.12.11.007 If the Destination User Unique Identifier does not reference an already existing User Unique Identifier, then the node receiving this command MUST report back to the initiating node with a failure status and MUST NOT change the User or Credential data in response to this command.

Destination Credential Slot (16 bits)

The Destination Credential Slot is used to identify the Credential Slot where the Credential will be stored after completing the operation of this command.

CC:0083.01.12.11.008 If the Destination Credential Slot is not supported by the receiving node, or the Destination Credential Slot is already occupied by another preexisting credential, then the node receiving this command MUST report back to the initiating node with a failure status and MUST NOT change the User or Credential data in response to this command.

2.2.118.18 User Credential Association Report Command

This command is used to make other nodes aware of the action taken in response to the *User Credential Association Set Command*.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_CREDENTIAL_ASSOCIATION_REPORT (0x13)							
Credential Type							
Source Credential Slot (MSB)							
Source Credential Slot (LSB)							
Destination User Unique Identifier (MSB)							
Destination User Unique Identifier (LSB)							
Destination Credential Slot (MSB)							
Destination Credential Slot (LSB)							
User Credential Association Status							

For "Credential Type", "Source Credential Slot", "Destination User Unique Identifier", and "Destination Credential Slot" descriptions, refer to the *User Credential Association Set Command*.

User Credential Association Status (8 bits)

This field indicates the result of the user credential association set process.

CC:0083.01.13.11.000 This field MUST comply with Table 2.554.

Table 2.554: User Credential Association Report::User Credential Association Status

Value	Name	Definition
0x00	Success	The operation was performed successfully, and now the Source Credential is associated with the Destination User Unique Identifier and stored in the Destination Credential Slot.
0x01	Credential Type Invalid	The Credential Type does not reference a valid credential type for the target device, for example because the identifier provided does not refer to a credential type supported by that device.
0x02	Source Credential Slot Invalid	The Source Credential Slot does not reference a valid credential slot within the given Credential Type, for example because the identifier provided is outside the allowed range.
0x03	Source Credential Slot Empty	The Source Credential Slot does not reference an existing credential within the given Credential Type on the target device.
0x04	Destination User Unique Identifier Invalid	The Destination User Unique Identifier does not reference a valid user identifier, for example because the identifier provided is outside the allowed range.
0x05	Destination User Unique Identifier Nonexistent	The Destination User Unique Identifier does not reference an existing user on the target device.
0x06	Destination Credential Slot Invalid	The Destination Credential Slot does not reference a valid credential slot within the given Credential Type, for example because the identifier provided is outside the allowed range.
0x07	Destination Credential Slot Occupied	The Destination Credential Slot is already occupied by an existing credential within the given Credential Type prior to the attempted operation.

CC:0083.01.13.11.001 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

2.2.118.19 All Users Checksum Get Command

This command is used to request an All Users checksum representing the User Unique Identifiers, User Types, User Active States, Credential Rules, User Name Encodings, User Name Lengths, User Names, and all the Credentials currently set at the receiving node for all User Unique Identifiers with any data.

CC:0083.01.14.11.000 This command MUST be ignored by a node advertising no support for the All Users Checksum functionality in the *User Capabilities Report Command*.

CC:0083.01.14.11.001 The *All Users Checksum Report Command* MUST be returned in response to this command if the All Users Checksum functionality is advertised as supported in the User Capabilities Report Command.

CC:0083.01.14.11.002 This command MUST NOT be issued via multicast addressing.

CC:0083.01.14.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = ALL_USERS_CHECKSUM_GET (0x14)							

2.2.118.20 All Users Checksum Report Command

This command is used to advertise the current All Users checksum representing the User Unique Identifiers, User Types, User Active States, Credential Rules, User Name Encodings, User Name Lengths, User Names, and all the Credentials set at the sending node for all User Unique Identifiers with any data. For these field's definitions, refer to the *User Set Command* and the *Credential Set Command*.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = ALL_USERS_CHECKSUM_REPORT (0x15)							
All Users Checksum (MSB)							
All Users Checksum (LSB)							

All Users Checksum (16 bits)

This field is used to advertise the checksum representing the User Unique Identifiers, User Types, User Active States, Credential Rules, User Name Encodings, User Name Lengths, User Names, and all Credentials set at the sending node for all User Unique Identifiers with any data.

CC:0083.01.15.11.000 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). Refer to *CRC-CCITT Source Code* for details.

CC:0083.01.15.12.011 If the on-demand calculation of the checksum of the full database on that particular device might take longer than 2 seconds, it is RECOMMENDED to either calculate the checksum in background after each database modification or not to support the All Users Checksum feature.

CC:0083.01.15.11.001 The checksum data MUST be built by concatenating the User Unique Identifier, User Type, User Active State, Credential Rule, User Name Encoding, User Name Length, User Name, and then defined Credentials data in the ascending Credential Type and then Slot Number order for the User Unique Identifier where User Unique Identifier is also in ascending order.

CC:0083.01.15.11.002 Each User Unique Identifier checksum MUST be formatted as follows:

> User Unique Identifier (16 bits) | User Type (8 bits) | User Active State (8 bits) | Credential Rule (8 bits) | User Name Encoding (8 bits) | User Name Length (8 bits) | User Name (User Name Length bytes)

Followed with each defined User Unique Identifier Credential data formatted as follows:

Credential Type (8 bits) | Credential Slot (16 bits) | Credential Length (8 bits) | Credential Data (Credential Length bytes)

CC:0083.01.15.11.003 Credential Length and Credential Data MUST be set to what is reported in the *Credential Report Command*.

CC:0083.01.15.11.004 If a Credential Length of 0 is reported or a hashed value is reported rather than the full Credential Data, that Credential Length of 0 or hashed Credential Data value MUST be used in the checksum calculation.

CC:0083.01.15.11.005 If there is no Credential data set for a User Unique Identifier, the checksum MUST NOT be modified other than for the User Unique Identifier data.

CC:0083.01.15.11.006 If there is no Users data (and thus no Credentials data) set at the node at all, the checksum MUST be set to 0x0000.

For example, a node with Users Data:

User Unique Identifier 0x0003, User Type 0x04 (Non-Access User), User Active State 0x01 (Occupied Enabled), Credential Rule 0x01 (Single), User Name Encoding 0x00 (Standard ASCII), User Name Length 0x06, User Name Jackie

and 4 Credentials that are set as follows:

- Credential Type 0x01 (PIN Code), Credential Slot 0x0002, Credential Length 4, Credential Data 9277
- Credential Type 0x01 (PIN Code), Credential Slot 0x0004, Credential Length 6, Credential Data 954988
- Credential Type 0x02 (Password), Credential Slot 0x0001, Credential Length 34 (Unicode UTF-16 format, in big endian order), Credential Data zwavenodepassword
- Credential Type 0x09 (Finger Biometric), Credential Slot 0x0008, Credential Length 2, Credential Data 0x4583 (Hashed Biometric Data)

User Unique Identifier 0x0005, User Type 0x00 (General User), User Active State 0x01 (Occupied Enabled), Credential Rule 0x01 (Single), User Name Encoding 0x00 (Standard ASCII), User Name Length 0x04, User Name Mike

and 2 Credentials that are set as follows:

- Credential Type 0x01 (PIN Code), Credential Slot 0x0003, Credential Length 4, Credential Data 4812
- Credential Type 0x08 (Face Biometric), Credential Slot 0x0003, Credential Length 2, Credential Data 0x2755 (Hashed Biometric Data)

User Unique Identifier 0x0007, User Type 0x06 (Disposable User), User Active State 0x01 (Occupied Enabled), Credential Rule 0x01 (Single), User Name Encoding 0x00 (Standard ASCII), User Name Length 0x00

and no Credentials set

CC:0083.01.15.11.007 In this case, data for the User Unique Identifier MUST be concatenated to obtain the checksum data in this way:

0x0003 | 0x04 | 0x01 | 0x01 | 0x00 | 0x06 | 0x4A 61 63 6B 69 65

- 0x01 | 0x0002 | 0x04 | 0x39 32 37 37
- 0x01 | 0x0004 | 0x06 | 0x39 35 34 39 38 38
- 0x02 | 0x0001 | 0x22 | 0x00 7A 00 77 00 61 00 76 00 65 00 6E 00 6F 00 64 00 65 00 70 00 61 00 73 00 73 00 77 00 6F 00 72 00 64
- 0x09 | 0x0008 | 0x02 | 0x45 83

0x0005 | 0x00 | 0x01 | 0x01 | 0x00 | 0x04 | 0x4D 69 6B 65

- 0x01 | 0x0003 | 0x04 | 0x34 38 31 32

- 0x08 | 0x0003 | 0x02 | 0x27 55

0x0007 | 0x06 | 0x01 | 0x01 | 0x00 | 0x00

CC:0083.01.15.11.008 The checksum data MUST be: 0x000304010100064A61636B6965010002043932373701000406393534393838
02000122007A0077006100760065006E006F006400650070006100730073007700 6F00720064
0900080245830005000101000 44D696B6501000304343831320800030227550007060101 0000

CC:0083.01.15.11.009 The returned All Users Checksum field MUST be set to: 0xE602.

CC:0083.01.15.11.010 For another example, a node with no Users Data set, MUST return 0x0000 since there are no Users or Credentials at the node.

For nodes that support Admin Code functionality:

CC:0083.01.15.11.011 Admin Code information is NOT included in All Users Checksum, so a controlling node MUST separately request an Admin PIN Code Report with an *Admin PIN Code Get Command* to get the Admin Code information.

2.2.118.21 User Checksum Get Command

This command is used to request a User checksum representing the User Type, User Active State, Credential Rule, and all the Credentials currently set at the receiving node for a User Unique Identifier. For these field's definitions, refer to the *User Set Command* and the *Credential Set Command*.

CC:0083.01.16.11.000 This command MUST be ignored by a node advertising no support for the User Checksum functionality in the *User Capabilities Report Command*.

CC:0083.01.16.11.001 The *User Checksum Report Command* MUST be returned in response to this command if the User Checksum functionality is advertised as supported in the *User Capabilities Report Command*.

CC:0083.01.16.11.002 This command MUST NOT be issued via multicast addressing.

CC:0083.01.16.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_CHECKSUM_GET (0x16)							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							

User Unique Identifier (16 bits)

See *User Set Command* for this field's description.

2.2.118.22 User Checksum Report Command

This command is used to advertise the current User checksum representing the User Type, User Active State, Credential Rule, User Name Encoding, User Name Length, User Name, and all the Credentials set at the sending node for a User Unique Identifier.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = USER_CHECKSUM_REPORT (0x17)							
User Unique Identifier (MSB)							
User Unique Identifier (LSB)							
User Checksum (MSB)							
User Checksum (LSB)							

User Unique Identifier (16 bits)

See *User Set Command* for this field's description.

User Checksum (16 bits)

This field is used to advertise the checksum representing the User Type, User Active State, Credential Rule, User Name Encoding, User Name Length, User Name, and all Credentials set at the sending node for a User Unique Identifier.

CC:0083.01.17.11.000 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). Refer to *CRC-CCITT Source Code* for details.

CC:0083.01.17.11.001 The checksum data MUST be built by concatenating the User Type, User Active State, Credential Rule, User Name Encoding, User Name Length, User Name, and then defined Credentials data in the ascending Credential Type and then Slot Number order for the User Unique Identifier.

CC:0083.01.17.11.002 Each User Unique Identifier checksum MUST be formatted as follows:

> User Type (8 bits) | User Active State (8 bits) | Credential Rule (8 bits) | User Name Encoding (8 bits) | User Name Length (8 bits) | User Name (User Name Length bytes)

Followed with each defined User Unique Identifier Credential data formatted as follows:

> Credential Type (8 bits) | Credential Slot (16 bits) | Credential Length (8 bits) | Credential Data (Credential Length bytes)

CC:0083.01.17.11.003 Credential Length and Credential Data MUST be set to what is reported in the *Credential Report Command*.

CC:0083.01.17.11.004 If a Credential Length of 0 is reported or a hashed value is reported rather than the full Credential Data, that Credential Length of 0 or hashed Credential Data value MUST be used in the checksum calculation.

CC:0083.01.17.11.005 If there is no Credential data set for a User Unique Identifier, the checksum MUST NOT be modified other than for the User Unique Identifier data.

CC:0083.01.17.11.006 If there is no User data (and thus no Credentials data) set at the node for a User Unique Identifier, the checksum MUST be set to 0x0000.

For example, a node with User Data:

> User Type 0x04 (Non-Access User), User Active State 0x01 (Occupied Enabled), Credential Rule 0x01 (Single), User Name Encoding 0x00 (Standard ASCII), User Name Length 0x04, User Name Matt
>
> and 4 Credentials that are set as follows:
>
> - Credential Type 0x01 (PIN Code), Credential Slot 0x0002, Credential Length 4, Credential Data 9277
> - Credential Type 0x01 (PIN Code), Credential Slot 0x0004, Credential Length 6, Credential Data 954988
> - Credential Type 0x02 (Password), Credential Slot 0x0001, Credential Length 34 (Unicode UTF-16 format, in big endian order), Credential Data zwavenodepassword
> - Credential Type 0x0A (Hand Biometric), Credential Slot 0x0008, Credential Length 2, Credential Data 0x7199 (Hashed Biometric Data)

CC:0083.01.17.11.007 In this case, data for the User Unique Identifier MUST be concatenated to obtain the checksum data in this way:

0x04 | 0x01 | 0x01 | 0x00 | 0x04 | 0x4D 61 74 74

- 0x01 | 0x0002 | 0x04 | 0x39 32 37 37
- 0x01 | 0x0004 | 0x06 | 0x39 35 34 39 38 38
- 0x02 | 0x0001 | 0x22 | 0x00 7A 00 77 00 61 00 76 00 65 00 6E 00 6F 00 64 00 65 00 70 00 61 00 73 00 73 00 77 00 6F 00 72 00 64
- 0x0A | 0x0008 | 0x02 | 0x71 99

CC:0083.01.17.11.008 The checksum data MUST be: 0x04010100044D617474010002043932373701000406393534393838 02000122007A007700610076006500 6E006F006400650070006100730073007700 6F007200640A0008027199.

CC:0083.01.17.11.009 The returned User Credential Checksum field MUST be set to: 0x9024.

For another example, a node with User Data:

> User Type 0x04 (Non-Access User), User Active State 0x01 (Occupied Enabled), Credential Rule 0x01 (Single), User Name Encoding 0x00 (Standard ASCII), User Name Length 0x06, User Name Lillie
>
> and no Credentials set.

CC:0083.01.17.11.010 In this case, data for the User Unique Identifier MUST be concatenated to obtain the checksum data in this way:

0x04 | 0x01 | 0x01 | 0x00 | 0x06 | 0x4C 69 6C 6C 69 65

CC:0083.01.17.11.011 The checksum data MUST be: 0x04010100064C696C6C6965.

CC:0083.01.17.11.012 The returned User Credential Checksum field MUST be set to: 0xF900.

CC:0083.01.17.11.013 For another example, a node with no User Data set at User Unique Identifier 0x0010, MUST return 0x0000 since there is no User at that User Unique Identifier.

2.2.118.23 Credential Checksum Get Command

This command is used to request a Credential checksum representing the Credential Slots, Credential Lengths, and Credential Data at the receiving node for a Credential Type. For these field's definitions, refer to the *Credential Set Command*.

CC:0083.01.18.11.000 This command MUST be ignored by a node advertising no support for the Credential Checksum functionality in the *Credential Capabilities Report Command*.

CC:0083.01.18.11.001 The *Credential Checksum Report Command* MUST be returned in response to this command if the Credential Checksum functionality is advertised as supported in the *Credential Capabilities Report Command*.

CC:0083.01.18.11.002 This command MUST NOT be issued via multicast addressing.

CC:0083.01.18.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel Multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_CHECKSUM_GET (0x18)							
Credential Type							

Credential Type (8 bits)

CC:0083.01.18.11.004 This field MUST comply with values listed in Table 2.546.

2.2.118.24 Credential Checksum Report Command

This command is used to advertise the current Credential checksum representing the Credential Slots, Credential Lengths, and Credential Data at the receiving node for a Credential Type.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = CREDENTIAL_CHECKSUM_REPORT (0x19)							
Credential Type							
Credential Checksum (MSB)							
Credential Checksum (LSB)							

Credential Type (8 bits)

CC:0083.01.19.11.000 This field MUST comply with values listed in Table 2.546.

Credential Checksum (16 bits)

This field is used to advertise the current Credential checksum representing the Credential Slots, Credential Lengths, and Credential Data at the receiving node for a Credential Type.

CC:0083.01.19.11.001 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). Refer to *CRC-CCITT Source Code* for details.

CC:0083.01.19.11.002 The checksum data MUST be built by concatenating the Credential Slot, Credential Length, and Credential Data in the ascending Credential Slot order for the Credential Type.

CC:0083.01.19.11.003 Each Credential checksum MUST be formatted as follows:

> Credential Slot (16 bits) | Credential Length (8 bits) | Credential Data (Credential Length bytes)

CC:0083.01.19.11.004 Credential Length and Credential Data MUST be set to what is reported in the *Credential Report Command*.

CC:0083.01.19.11.005 If a Credential Length of 0 is reported or a hashed value is reported rather than the full Credential Data, that Credential Length of 0 or hashed Credential Data value MUST be used in the checksum calculation.

CC:0083.01.19.11.006 If there is no Credentials data set at the node for a Credential Type, the checksum MUST be set to 0x0000.

For example, a node with Credential data:

- Credential Type 0x01 (PIN Code), Credential Slot 0x0002, Credential Length 4, Credential Data 9277
- Credential Type 0x01 (PIN Code), Credential Slot 0x0004, Credential Length 6, Credential Data 954988
- Credential Type 0x02 (Password), Credential Slot 0x0001, Credential Length 34 (Unicode UTF-16 format, in big endian order), Credential Data zwavenodepassword
- Credential Type 0x07 (Eye Biometric), Credential Slot 0x0008, Credential Length 2, Credential Data 0x2401 (Hashed Biometric Data)

CC:0083.01.19.11.007 In this case, data for the Credential Type PIN Code MUST be concatenated to obtain the checksum data in this way:

> 0x0002 | 0x04 | 0x39 32 37 37 | 0x0004 | 0x06 | 0x39 35 34 39 38 38

CC:0083.01.19.11.008 The checksum data MUST be: 0x0002043932373700040639353439383 8.

CC:0083.01.19.11.009 The returned Credential Checksum field MUST be set to: 0xD867.

CC:0083.01.19.11.010 In this case, data for the Credential Type Password MUST be concatenated to obtain the checksum data in this way:

0x0001 | 0x11 | 0x00 7A 00 77 00 61 00 76 00 65 00 6E 00 6F 00 64 00 65 00 70 00 61 00 73 00 73 00 77 00 6F 00 72 00 64

CC:0083.01.19.11.011 The checksum data MUST be: 0x000122007A00770061007600650 06E006F006400650070006100730073007700 6F007200

CC:0083.01.19.11.012 The returned Credential Checksum field MUST be set to: 0x6F76.

CC:0083.01.19.11.013 In this case, data for the Credential Type Eye Biometric MUST be concatenated to obtain the checksum data in this way:

0x0008 | 0x02 | 0x24 01

CC:0083.01.19.11.014 The checksum data MUST be: 0x0008022401

CC:0083.01.19.11.015 The returned Credential Checksum field MUST be set to: 0xC06E.

CC:0083.01.19.11.016 In this case, data for the Credential Type RFID Code MUST return 0x0000 since there are no RFID Codes set at the node.

2.2.118.25 Admin Code Information

The Admin Code (also 'Admin PIN Code') is the mechanism by which compatibility is maintained with the *User Code Command Class, version 2* if the node supports Admin Code within the User Code Command Class. Some nodes have a need for a method of ingress or manipulation that is tied to the node, and not a specific User. The Admin Code is intended to exist statically, with no attachment to a specific User or Credential.

In practical terms, it can be visualized similarly to a permanent Programming User with a persistent PIN Code Credential, albeit with a few differences:

- There are not multiple Admin Codes
- The 'user' does not have a UUID and cannot be modified, deleted or moved
- The 'user' only has 1 PIN Code and no other Credentials, and no other credentials may be added
- The PIN Code cannot be moved or deleted, but can be modified or deactivated (if deactivation is supported by the node).
- Delete All Users or All Credentials do not affect Admin Code.

In-depth requirements and functions are explained in detail below.

This functionality is primarily intended to ensure minimization of undefined or undesired behavior when nodes supporting User Code and User Credential exist on a network.

CC:0083.01.00.13.003 If the User Code Command Class is NOT supported by the node, only User Credential, the Admin Code functionality MAY still be used if needed to fulfill other operational requirements of the node.

CC:0083.01.00.12.003 However, it is RECOMMENDED that in the case where User Code is NOT supported, a Programming User(s) SHOULD be used instead.

If the Admin Code is supported:

CC:0083.01.00.11.005 - PIN Code Credentials MUST be supported.

CC:0083.01.00.11.006 - The Admin Code MUST NOT be tied to any other Credential Type (RFID, Fingerprint, etc).

CC:0083.01.00.11.007 - The Admin Code MUST be subject to all of the restrictions and requirements of the PIN Code Credential Type.

CC:0083.01.00.11.008 - The Admin Code MUST NOT conflict with the operation of any existing PIN Code Credentials and vice versa.

CC:0083.01.00.11.009 - The Admin Code MUST NOT be a duplicate of any existing PIN Code Credentials and vice versa.

CC:0083.01.00.21.013 In the case that the node supports both User Code and User Credential Command Classes, requirements in *Compatibility Considerations* MUST supersede the requirements for the following commands:

- *Admin PIN Code Set Command*

- *Admin PIN Code Get Command*
- *Admin PIN Code Report Command*

2.2.118.26 Admin PIN Code Set Command

This command is used to set the Admin Code in the receiving node.

CC:0083.01.1A.13.001 When a node advertising AC Support receives an Admin PIN Code Set, an Admin PIN Code Report with an appropriate Result Code (see Table 2.555) MAY be required to be returned to the sender and/or Lifeline association group, subject to the later requirements in this section.

CC:0083.01.1A.12.001 In the case of no appropriate code, Unspecified Node Error (0x0F) SHOULD be used.

CC:0083.01.1A.11.001 When a node receives an Admin PIN Code Set that is successful in changing the Admin Code, an Admin PIN Code Report with Result Code 0x01 (Modified) MUST be returned to sender and sent to the Lifeline association group.

CC:0083.01.1A.13.002 When a node not advertising AC support receives an Admin PIN Code Set command, the node MAY either return an Admin PIN Code Report with Result Code 0x0D (AC Not Supported) to the sender or ignore the command.

CC:0083.01.1A.11.002

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = ADMIN_PIN_CODE_SET (0x1A)							
Reserved				Admin PIN Code Length			
Admin PIN Code 1							
...							
Admin PIN Code N							

Reserved

CC:0083.01.1A.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Admin PIN Code Length (4 bits)

This field is used to advertise the length in bytes of the Admin PIN Code field in the command.

CC:0083.01.1A.11.015 This field MUST be set to 0 or within the range Min..Max Credential Length of Credential Type PIN Code.

CC:0083.01.1A.11.016 Values greater than 0 and in accordance with the supported range MUST indicate that the receiving node MUST set the Admin Code as indicated in the Admin PIN Code field.

CC:0083.01.1A.11.006 The value 0 MUST indicate that the receiving node MUST deactivate the Admin Code if ACD is supported by the node.

CC:0083.01.1A.13.003 When a node not advertising ACD support receives an Admin PIN Code Set with code length 0, the node MAY either return an Admin PIN Code Report with Result Code 0x0E (ACD Not Supported) to the sender, or ignore the command.

Admin PIN Code (N bytes)

This field is used to advertise the Admin Code to be set for the node.

CC:0083.01.1A.11.007 The length of this field MUST be set according to the Admin PIN Code Length field value.

CC:0083.01.1A.11.008 Each byte in this field MUST be encoded with ASCII representation.

CC:0083.01.1A.11.009 Unless superseded by other requirements, the encoding and characters that the Admin Code supports MUST match the characters and encoding supported by credentials of Credential Type PIN Code.

CC:0083.01.1A.11.010 A supporting node receiving a non-supported ASCII character MUST ignore the command.

CC:0083.01.1A.13.004

A supporting node receiving an Admin Code identical to the current Admin Code MAY either return an Admin PIN Code Report with a result code of 0x03 (Unmodified) to the sending node, or ignore the command.

CC:0083.01.1A.13.005 A supporting node receiving an Admin Code identical to a PIN Code that already exists in a current Credential MAY either return an Admin PIN Code Report with a result code of 0x07 (Duplicate Credential) to the sending node, or ignore the command.

CC:0083.01.1A.11.011 A supporting node receiving an Admin Code that fails a manufacturer security rule check MUST return an Admin PIN Code Report with a result code of 0x08 (Manufacturer Security Rule) to the sending node.

CC:0083.01.1A.11.012 A supporting node receiving an Admin Code with an invalid length, format, or encoding MUST ignore the command.

When the Admin Code is input, the node:

CC:0083.01.1A.11.013
- MUST indicate that the code is accepted

CC:0083.01.1A.11.014
- MUST grant access to administrator functionalities, such as the network settings and/or User Credential management interface.

CC:0083.01.1A.13.007
- MAY operate the node

2.2.118.27 Admin PIN Code Get Command

This command is used to request the Admin Code currently set at the receiving node.

CC:0083.01.1B.13.001 If AC functionality is NOT supported, an Admin PIN Code Report Command with Result Code AC Not Supported (0x0D) MAY be returned to the sending node in response to this command, or simply ignored.

CC:0083.01.1B.11.001 If AC functionality is supported, the Admin PIN Code Report Command with an appropriate Result Code MUST be returned in response to this command.

The supported Get result codes are:

- 0x04: Response to Get
- 0x0F: Unspecified Node Error

See Table 2.555 for further information on allowed result codes.

CC:0083.01.1B.11.002 This command MUST NOT be issued via multicast addressing.

CC:0083.01.1B.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = ADMIN_PIN_CODE_GET (0x1B)							

2.2.118.28 Admin PIN Code Report Command

This command is used to advertise the Admin Code currently set at the sending node.

CC:0083.01.1C.11.001 When a node's Admin Code is updated using a local method (keypad, mobile application, etc.), an Admin PIN Code Report with Result Code 0x01 (Modified) MUST be sent to the Lifeline association group.

CC:0083.01.1C.11.002 Admin PIN Code Reports with operation result codes not specified in Table 2.555 MUST be ignored.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_USER_CREDENTIAL (0x83)							
Command = ADMIN_PIN_CODE_REPORT (0x1C)							
Admin PIN Code Operation Result				Admin PIN Code Length			
Admin PIN Code 1							
...							
Admin PIN Code N							

Admin PIN Code Operation Result

See Admin PIN Code Set and Admin PIN Code Get for specific requirements for the Operation Result field.

This field allows a node to pass information about the result of an Admin Code operation to another node.

CC:0083.01.1C.11.003

Table 2.555: Admin PIN Code Report::Admin PIN Code Operation Result

Numeric Value	Operation (Get/Set)	Name	Description
0x01	Modified	Set	Admin Code was modified. Reports with this code MUST be returned to the sender and sent on the Lifeline association group.
0x03	Unmodified	Set	Admin Code was not modified as the current code is identical.
0x04	Response to Get	Get	Successful Get operation.
0x07	Fail - Duplicate Credential	Set	Admin Code was not modified - a duplicate PIN code exists in the Credential database.
0x08	Fail - Manufacturer Security Rule	Set	Admin Code was not modified as the provided code fails a manufacturer security rule.
0x0D	Error - AC Not Supported	Get/Set	Operation cannot be completed as Admin Code functionality is unsupported.
0x0E	Error - ACD Not Supported	Set	Operation cannot be completed as Admin Code cannot be disabled.
0x0F	Unspecified Node Error	Get/Set	Issue with Get or Set not explicitly stated in the specification.

Admin PIN Code Length (4 bits)

This field is used to advertise the length in bytes of the Admin PIN Code field in the command.

CC:0083.01.1C.11.008 If the Admin Code is set and advertised in this command, this field MUST be set to a value greater than 0 and within the range Min..Max Credential Length of Credential Type PIN Code.

CC:0083.01.1C.11.005 This field MUST be set to 0 if the Admin Code is deactivated or if the sending node advertises no support for the Admin Code functionality (AC Support) in the Credential Capabilities Report Command.

Admin PIN Code (N bytes)

This field is used to advertise the Admin Code currently set at the sending node.

CC:0083.01.1C.11.006 The length of this field MUST be set according to the Admin PIN Code Length field value. If the Admin PIN Code Length is set to 0, this field MUST be omitted.

CC:0083.01.1C.11.007 Each byte in this field MUST be encoded with ASCII representation.

2.2.119 Window Covering Command Class, version 1

The Window Covering Command Class is used to control window covering devices.

The Window Covering Command Class is an actuator control command class. Refer to Section 2.1.6.

2.2.119.1 Terminology

A window covering device may be **closed** or **open**. The term closed represents the lowest light throughput, while open represents the highest light throughput.

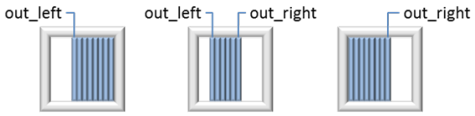
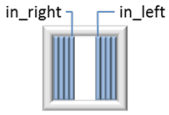
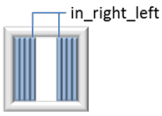
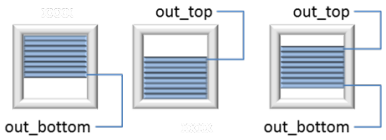
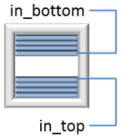
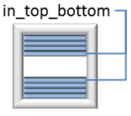
A window covering device may provide one or more properties such as up/down movement combined with for example slats angle control.

Each of the available properties is advertised. A window covering device may offer precise control of the position of a property or it may offer a more limited movement control, where it is possible to start and stop movement in one of two directions but where the target position cannot be specified.

If a window covering parameter is marked as **position unknown**, this indicates that the actual property can only be controlled via start and stop of movement.

The parameters controlling edges are identified as outlined in Table 2.556.

Table 2.556: Identification of edges

Edge Title	Description	Usage Examples
out_left	Outbound edge towards the left	
out_right	Outbound edge towards the right	
in_left	Inbound edge towards the left	
in_right	Inbound edge towards the right	
in_right_left	Inbounded edges controlled horizontally as one	
out_bottom	Outbound edge towards the bottom	
out_top	Outbound edge towards the top	
in_bottom	Inbound edge towards the bottom	
in_top	Inbound edge towards the top	
in_bottom_top	Inbound edges controlled vertically as one	

2.2.119.2 Compatibility Considerations

This Command Class replaces the following command classes:

- Basic Window Covering Command Class [OBSOLETED]
- Move To Position Window Covering Command Class [OBSOLETED]

2.2.119.2.1 Motor Control Device Class Support

A node supporting this Command Class SHOULD comply implement the Motor Control Specific Device Class B or C (refer to [34] or Section 7) for backwards compatibility with existing window covering control applications.

2.2.119.2.2 Command Class Dependencies

A node supporting the Window Covering Command Class, Version 1 MUST support the Multilevel Switch Command Class, version 3 or newer.

With the exception for slats angle control, the following mapping MUST apply:

- The Multilevel Switch value 0x00 MUST represent the least light, i.e. covering fully closed.
- The Multilevel Switch value 0x63 MUST represent the most light, i.e. covering fully opened.

For slats angle control, the following mapping MUST apply:

- The Multilevel Switch value 0x00 MUST represent slats closed to the one side.
- The Multilevel Switch value 0x32 MUST represent slats open.
- The Multilevel Switch value 0x63 MUST represent slats closed to the other side.

By mapping the Multilevel Switch value 0 to the closed position, a multicasted or broadcasted Multilevel Switch Command that turns off light sources will also cause window coverings to close, thus reducing light from the outside.

A supporting node MAY use the Multilevel Switch Command Class to control multiple features like up/down control as well as slats angle control. For instance, the Start Level Change command may start changing the slats angle towards the extreme angle and when reaching that extreme angle, it may start moving up or down.

This way, multiple functions can be controlled from a general purpose remote control, although with less convenience than if a dedicated window covering remote control is used.

2.2.119.3 Window Covering Parameters

Each distinct property of a window covering node is accessed via a dedicated parameter.

Two variants are defined for many parameters: One allows full position control, including movement control. The other variant is limited to movement control.

If a node supports full position control of a property (even parameter IDs), it MUST NOT support the corresonding parameter limited to movement control (odd parameter IDs).

The term “Closed” represents the lowest light throughput, while “Open” represents the highest light throughput of the window covering.

Table 2.557: Window Covering Parameter IDs

Parm ID	Description	Value Range	Encoding
Outbound horizontal control			
0	out_left: Outbound edge towards the left Right/Left movement (Position unkonwn)	N/A	Level Change Up = Opening Level Change Down = Closing
1	out_left: Outbound edge towards the left Right/Left position Parm 0 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open

continues on next page

Table 2.557 – continued from previous page

Parm ID	Description	Value Range	Encoding
2	out_right: Outbound edge towards the right Right/Left movement (Position unknown)	N/A	Level Change Up = Opening Level Change Down = Closing
3	out_right: Outbound edge towards the right Right/Left position Parm 2 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open
Inbound horizontal control			
4	in_left: Inbound edge towards the left Right/Left movement (Position unknown)	N/A	Level Change Down = Closing Level Change Up = Opening
5	in_left: Inbound edge towards the left Right/Left position Parm 4 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open
6	in_right: Inbound edge towards the right Right/Left movement (Position unknown)	N/A	Level Change Down = Closing Level Change Up = Opening
7	in_right: Inbound edge towards the right Right/Left position Parm 6 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open
8	in_right_left: Inbound edges controlled horizontally as one Right/Left movement (Position unknown)	N/A	Level Change Down = Closing Level Change Down = Opening
9	in_right_left: Inbound edges controlled horizontally as one Right/left position Parm 8 MUST NOT be supported	0x00-0x63	Level Change Down = Closing Level Change Up = Opening
Angle control of vertical slats			
10	Vertical slats angle Right/Left movement (Position unkonwn)	N/A	Level Change Down = Closing; to the right inside Level Change Up = Closing; to the left inside (Open is passed midway between the two closing positions)
11	Vertical slats angle Right/Left position Parm 10 MUST NOT be supported	0x00-0x63	0x00 = Closed; to the right inside 0x32 = Open 0x63 = Closed; to the left inside
Outbound vertical control			
12	out_bottom: Outbound edge towards the bottom Up/Down movement (Position unknown)	N/A	Level Change Down = Closing Level Change Up = Opening

continues on next page

Table 2.557 – continued from previous page

Parm ID	Description	Value Range	Encoding
13	out_bottom: Outbound edge towards the bottom Up/Down position Parm 12 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open
14	out_top: Outbound edge towards the top Up/Down movement (Position unknown)	N/A	Level Change Down = Closing Level Change Up = Opening
15	out_top: Outbound edge towards the top Up/Down position Parm 14 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open
Inbound vertical control			
16	in_bottom: Inbound edge towards the bottom Up/Down movement (Position unknown)	N/A	Level Change Down = Closing Level Change Up = Opening
17	in_bottom: Inbound edge towards the bottom	0x00-0x63	0x00 = Closed 0x63 = Open
18	in_top: Inbound edge towards the top Up/Down movement (Position unknown)	N/A	Level Change Down = Closing Level Change Up = Opening
19	in_top: Inbound edge towards the top Up/Down position Parm 18 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open
20	in_top_bottom: Inbound edges controlled vertically as one Up/Down movement (Position unknown)	N/A	Level Change Down = Closing Level Change Up = Opening
21	in_top_bottom: Inbound edges controlled vertically as one Up/Down position Parm 20 MUST NOT be supported	0x00-0x63	0x00 = Closed 0x63 = Open
Angle control of horizontal slats			
22	Horizontal slats angle Up/Down movement (Position unknown)	N/A	Level Change Down = Closing; up inside Level Change Up = Closing; down inside (Open is passed midway between the two closing positions)
23	Horizontal slats angle Up/Down position Parm 22 MUST NOT be supported	0x00-0x63	0x00 = Closed; up inside 0x32 = Open 0x63 = Closed; down inside

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

A device MAY implement up to 100 hardware levels (including 0) for a given parameter. If a device implements less than 100 hardware levels, the hardware levels MUST be distributed uniformly over the entire range.

The mapping of parameter values to hardware levels MUST be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level. An example is found in Table 150.

Table 2.558: Parameter value mapping to a limited number of hardware levels (example)

Horizontal Slats Angle	Hardware Level
0x00..0x13	100% Up inside
0x14..0x27	50% Up inside
0x28..0x3B	Open
0x3C..0x4F	50% Down inside
0x50..0x63	100% Down inside

2.2.119.4 Window Covering Supported Get Command

This command is used to request the supported properties of a device.

The Window Covering Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods

7	6	5	4	3	2	1	0
Comand Class = COMMAND_CLASS_WINDOW_COVERING (0x6A)							
Command = WINDOW_COVERING_SUPPORTED_GET (0x01)							

2.2.119.5 Window Covering Supported Report Command

This command is used to advertise the supported properties of a device.

7	6	5	4	3	2	1	0
Comand Class = COMMAND_CLASS_WINDOW_COVERING (0x6A)							
Command = WINDOW_COVERING_SUPPORTED_REPORT (0x02)							
Reserved				Number of Parameter Mask Bytes			
Parameter Mask 1							
...							
Parameter Mask N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Parameter Mask bytes (4 bits)

The Number of Parameter Masks field MUST advertise the number of bytes carrying the Covering Parameter Mask field.

The value MUST be in the range 1..15.

Parameter Mask (N Bytes)

The Parameter Mask field MUST advertise the Parameters supported by the device.

The length of this field MUST be advertised by the Number of Parameter Masks field.

- Bit 0 in Bit Mask 1 indicates if Parameter ID 0 is supported
- Bit 1 in Bit Mask 1 indicates if Parameter ID 1 is supported
- …

For the definition of Parameter IDs, refer to Table 2.557.

2.2.119.6 Window Covering Get Command

This command is used to request the status of a specified Covering Parameter.

The Window Covering Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Comand Class = COMMAND_CLASS_WINDOW_COVERING (0x6A)							
Command = WINDOW_COVERING_GET (0x03)							
Parameter ID							

Parameter ID (8 bits)

This field MUST specify the Parameter for which the status is requested.

For the definition of Parameter IDs, refer to Table 2.557.

A node receiving an even Parameter ID (marked as “position unknown”) MUST set the Current Value and Target Value fields to 0x00 and the Duration field to 0xFE in the returned Report if the window covering is not moving (i.e., stationary).

2.2.119.7 Window Covering Report Command

This command is used to advertise the status of a Parameter.

7	6	5	4	3	2	1	0
Comand Class = COMMAND_CLASS_WINDOW_COVERING (0x6A)							
Command = WINDOW_COVERING_REPORT (0x04)							
Parameter ID							
Current Value							
Target Value							
Duration							

Parameter ID (8 bits)

This field MUST advertise the Parameter covered by this report.

For the definition of Parameter IDs, refer to Table 2.557.

Current Value (8 bits)

The Current Value field MUST advertise the current value of the Parameter identified by the Parameter ID field.

The Current Value SHOULD be identical to the Target Value when a transition has ended.

For the definition of valid parameter values, refer to Table 2.557.

Target Value (8 bits)

The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition for the advertised Parameter ID.

If a transition is initiated (i.e., for any Parameter IDs) in an interactive fashion via a local user interface or via a Start Level Change command, the advertised Target Value MUST be 0x00 or 0x63, depending on the direction. If there is not ongoing transation, the Current Value and the Target value field MUST set to 0X00 and Duration field MUST set to 0xFE for an even Parameter IDs.

For the definition of valid parameter values, refer to Table 2.557.

Duration (8 bits)

The Duration field SHOULD advertise the time needed to reach the Target Value at the actual transition rate. The encoding of the Duration field MUST be according to Table 2.10.

The Duration is defined as the interval from start of the transition until the Target Value is reached.

2.2.119.8 Window Covering Set Command

This command is used to control one or more parameters in a window covering device.

7	6	5	4	3	2	1	0
Comand Class = COMMAND_CLASS_WINDOW_COVERING (0x6A)							
Command = WINDOW_COVERING_SET (0x05)							
Reserved			Parameter Count				
Parameter ID 1							
Value 1							
...							
Parameter ID N							
Value N							
Duration							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Parameter Count (5 bits)

This field MUST specify the number of (Parameter ID, Value) datasets contained in the Window Covering Set command.

Parameter ID (N bytes)

This field MUST specify the Parameter to receive a new value.

A receiving node MUST ignore even Parameters ID (marked as “position unknown”).

For the definition of Parameter IDs, refer to Table 2.557.

Value (N bytes)

This field MUST specify the value of the Parameter identified by the Parameter ID field.

For the definition of valid parameter ID values, refer to Table 2.557.

Duration (8 bits)

The Duration field MUST specify the time that the transition should take from the current value to the new target value.

A supporting device SHOULD respect the specified Duration value.

The encoding of the Duration field MUST be according to Table 2.9.

2.2.119.9 Window Covering Start Level Change Command

This command is used to initiate a transition of one parameter to a new level.

A receiving node MUST initiate the transition to a new value for the specified Parameter ID.

7	6	5	4	3	2	1	0
Comand Class = COMMAND_CLASS_WINDOW_COVERING (0x6A)							
Command = WINDOW_COVERING_START_LEVEL_CHANGE (0x06)							
Res	Up/Down	Res					
Parameter ID							
Duration							

Up/Down (1 bit)

This field MUST specify the direction of the level change.

If the Up/Down bit is set to 0 the level change MUST be increasing.

If the Up/Down bit is set to 1 the level change MUST be decreasing.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Parameter ID (8 bits)

This field MUST specify the Parameter to start a transition.

A node receiving this command MUST accept all its supported parameter IDs (even and odd).

For the definition of Parameter IDs, refer to Table 2.557.

Duration (8 bits)

The level change rate MUST be calculated to match a level change from the minimum value to the maximum value during the time specified by the Duration field.

A supporting device SHOULD respect the specified Duration value.

The encoding of the Duration field MUST be according to Table 2.9.

2.2.119.10 Window Covering Stop Level Change Command

This command is used to stop an ongoing transition.

7	6	5	4	3	2	1	0
Comand Class = COMMAND_CLASS_WINDOW_COVERING (0x6A)							
Command = WINDOW_COVERING_STOP_LEVEL_CHANGE (0x07)							
Parameter ID							

Parameter ID (8 bits)

This field MUST specify the parameter to stop a transition.

A receiving node MUST stop the transition if the specified Parameter ID is currently in transition to a new value.

A receiving node MUST NOT stop ongoing transitions for other Parameter IDs than the one specified.

A node receiving this command MUST accept all parameter IDs supported by the device.

For the definition of Parameter IDs, refer to Table 2.557.

3 Management Command Classes

3.1 Management Command Class Overview

General command class overview and rules are described in the *Application Command Classes* and are valid for the Command Classes presented in this section.

No additional considerations apply for the Management Command Classes.

3.2 Management Command Class Definitions

The following subchapters contain definitions of Management Command Classes.

3.2.1 Application Capability Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST NOT support this Command Class.

New implementations MUST use the Supervision Command Class to report support or no support of a given Command Class. Refer to Section 4.

The Application Capability Command Class comprises commands for handling issues related to dynamic support for command classes.

Examples include nodes, which only support certain command classes when included securely, and controllers that only support command classes for primary controllers when they are actually operating as primary controllers.

The intention of the Application Capability Command Class is to provide tools for handling exceptions. Controlling nodes SHOULD maintain a local representation of supported command classes by destination nodes rather than relying on destination nodes returning Application Capability messages. Destination nodes capable of returning Application Capability messages MUST list Application Capability Command Class as supported only.

3.2.1.1 Not Supported Command Class Command

The Not Supported Command Class Command is used by a node to indicate to a requesting node that the requested command class is not supported. The offending command and command class are encapsulated.

A node SHOULD use this command to indicate to other nodes that a command is not supported.

A node receiving this command SHOULD use the information to clean up internal states by requesting up-to-date information on command class support from the sending node using Node Information frame. This includes adjusting user interface details accordingly.

Table 3.1: Not Supported Command Class Command (one byte)

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_APPLICATION_CAPABILITY (0x57)							
Command = COMMAND_COMMAND_CLASS_NOT_SUPPORTED (0x01)							
Dynamic	*Reserved*						
Offending Command Class (0x20 – 0xEE)							
Offending Command							

Table 3.2: Not Supported Command Class Command (two bytes)

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_APPLICATION_CAPABILITY (0x57)							
Command = COMMAND_COMMAND_CLASS_NOT_SUPPORTED (0x01)							
Dynamic	*Reserved*						
Offending Command Class MSB (0xF1 – 0xFF)							
Offending Command Class LSB (0x00 – 0xFF)							
Offending Command							

Payload data following the offending command is omitted.

Dynamic (1 bit)

If the "Dynamic" flag is set ('1'), the sending node has been designed to support this command class but the current operational conditions prevent the node from supporting this command class.

One example is the AddNode command used for network management. This command is only supported when a controller is operating as primary controller.

Table 3.3: Dynamic Bit

Field	Meaning
'0'	Command not supported (Permanently)
'1'	Dynamic support (Currently no support)

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Offending Command Class (1 or 2 bytes)

This field indicates the offending command class received by the destination node. The size of the command class field depends on the value of the first byte.

Offending Command (1 byte)

This field indicates the offending command received by the destination node.

3.2.2 Application Status Command Class, version 1

This command class contains commands that are not directly related to a specific functionality in the application, but are useful for maintaining an optimal Z-Wave system.

3.2.2.1 Compatibility Considerations

3.2.2.1.1 Node Information Frame (NIF)

A supporting node MUST always advertise the Application Status Command Class in its NIF, regardless of the inclusion status and security bootstrapping outcome.

3.2.2.2 Application Busy Command

The Application Busy Command used to instruct a node that the node that it is trying to communicate with is busy and is unable to service the request right now.

Table 3.4: Application Busy

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_APPLICATION_STATUS (0x22)							
Command = APPLICATION_BUSY (0x01)							
Status							
Wait Time							

Status (8 bits)

The status field can have the following values:

Table 3.5: Status Bits

Status	Description
0	Try again later
1	Try again in Wait Time seconds
2	Request queued, executed later

Wait Time (8 bits)

The wait time field indicates the number of seconds a node should wait before retrying the request.

3.2.2.3 Application Rejected Request Command

All supported commands are typically executed unconditionally and the only handshake is acknowledgement on the protocol level. Some applications can however be in a state where the application rejects to execute a supported command. The Application Rejected Request Command used to instruct a node that the command was rejected by the application in the receiving node.

Table 3.6: Application Rejected Request Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_APPLICATION_STATUS (0x22)							
Command = APPLICATION_REJECTED_REQUEST (0x02)							
Status							

Status (8 bits)

This field MUST be set to 0. The value 0 MUST indicate that the received (supported) command has been rejected by the application at the receiving node

3.2.3 Association Command Class, version 1 [OBSOLETED]

> **Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED**
>
> New implementations MUST NOT support this Command Class.
>
> New implemenetations MUST use the *Association Command Class, version 2*, or newer instead.

The Association Command Class is used to manage associations to NodeID destinations. A NodeID destination may be a simple device or the Root Device of a Multi Channel device.

An association group sends an unsolicited command to the configured destinations when triggered by an event. The parameters of the command may be dynamic, e.g. the temperature of a sensor reading or the light level for a dimmer.

3.2.3.1 Compatibility Considerations

CC:0085.01.00.22.001 The Association Group Information (AGI) Command Class SHOULD be supported to enable automated discovery of association group properties.

CC:0085.01.00.22.002 It is RECOMMENDED that a node which implements the Association Command Class also implements the Multi Channel Association Command Class for compatibility with End Point destinations. For instance, a wall switch may be configured to control one specific outlet of a power strip if the wall switch supports Multi Channel Association.

3.2.3.2 Z-Wave Plus Considerations

CC:0085.01.00.11.001 The Z-Wave Plus certification program mandates that association group 1 is reserved for the Lifeline association group. Association group 1 MUST NOT be assigned to any other use than the Lifeline group. The actual Device Type of a given product specifies mandatory commands which the device
CC:0085.01.00.13.001 must be able to send to a lifeline group destination. A manufacturer MAY add additional commands to the Lifeline group.

The Z-Wave Plus certification program mandates support for the Association Group Information (AGI) Command Class if a device supports the Association Command Class.

The Z-Wave Plus certification program recommends that a composite device is designed as a Multi Channel device.

3.2.3.3 Security Considerations

CC:0085.01.00.41.001 A node that has been S0/S2 bootstrapped MUST NOT accept Association commands unless the commands are received via the highest security key granted to the node during bootstrapping.

CC:0085.01.00.41.002 A supporting node issuing commands via association groups MUST send those commands with its highest granted Security Class.

CC:0085.01.00.42.001 A supporting node issuing Set or Report type commands via association groups SHOULD use Supervision encapsulation only if sending commands with S2 (or higher security) encapsulation.

3.2.3.4 Association Set Command

This command is used to add destinations to a given association group.

CC:0085.01.01.12.001 The receiving node SHOULD add the specified NodeID destinations to the specified association group.

CC:0085.01.01.13.001 This command MAY be ignored if the association group is already full.

CC:0085.01.01.11.001 Routing end nodes MUST have return routes assigned to all association destinations.

CC:0085.01.01.51.001 Unless the association destination is a gateway, a controlling node MUST NOT create an association if the association destination node does not support the controlling commands (Set/Get types) that

CC:0085.01.01.51.002 the actual association group will be sending. The AGI Command Class MUST be used to probe the commands that a given association group will be sending.

CC:0085.01.01.52.001 A controlling node SHOULD NOT create an association if the source and destination nodes are bootstrapped with different security levels.

Table 3.7: Association Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_SET (0x01)							
Grouping Identifier							
NodeID 1							
...							
NodeID N							

Grouping Identifier (8 bits)

CC:0085.01.01.11.002 This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

CC:0085.01.01.11.003 A receiving node MUST ignore an unsupported Grouping Identifier.

NodeID (N bytes)

This field specifies a list of NodeIDs that are to be added to the specified association group.

3.2.3.5 Association Get Command

This command is used to request the current destinations of a given association group.

CC:0085.01.02.11.001 The Association Report Command MUST be returned in response to this command.

CC:0085.01.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:0085.01.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.8: Association Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_GET (0x02)							
Grouping Identifier							

Grouping Identifier (8 bits)

CC:0085.01.02.11.004 This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

CC:0085.01.02.12.001 A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

3.2.3.6 Association Report Command

This command is used to advertise the current destinations of a given association group.

If the node supports the Multi Channel Association Command Class,

CC:0085.01.03.11.001 • the node MUST advertise the same Node ID destinations in this report as in the Multi Channel
• Association Report Command

CC:0085.01.03.11.002 • the node MUST NOT advertise the NodeID of End Point destinations in this report.

Table 3.9: Association Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_REPORT (0x03)							
Grouping Identifier							
Max Nodes Supported							
Reports to Follow							
NodeID 1							
...							
NodeID N							

Grouping Identifier (8 bits)

CC:0085.01.03.11.003 This field is used to advertise the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

Max Nodes Supported (8 bits)

CC:0085.01.03.13.001 The maximum number of destinations supported by the advertised association group. Each destination MAY be a NodeID destination or an End Point destination (if the node supports the Multi Channel Association Command Class).

Reports to Follow (8 bits)

The entire list destinations of the advertised association group may be too long for one command.
CC:0085.01.03.11.004 This field MUST advertise how many report frames will follow this report.

NodeID (N bytes)

This field advertises a list of NodeID destinations of the advertised association group. The list of
CC:0085.01.03.11.005 NodeIDs MUST be empty if there are no NodeID destinations configured for the advertised association group.

3.2.3.7 Association Remove Command

This command is used to remove destinations from a given association group.

CC:0085.01.04.11.001 If the node supports the Multi Channel Association Command Class the node MUST NOT remove End Point destinations in response to this command.

Table 3.10: Association Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_REMOVE (0x04)							
Grouping Identifier							
NodeID 1							
...							
NodeID N							

Grouping Identifier (8 bits)

This field is used to specify from which association group the specified NodeID destinations should be removed.

CC:0085.01.04.11.002 Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

CC:0085.01.04.11.003 This field MUST be interpreted in combination with the NodeID field.

CC:0085.01.04.11.004 A node that receives an unsupported Grouping Identifier MUST ignore this command.

NodeID (N bytes)

This field specifies a list of NodeID destinations that are to be removed from the specified association group.

CC:0085.01.04.11.005 The Grouping Identifier and NodeID fields MUST be interpreted as indicated in Table 3.11.

Table 3.11: Association Remove, V1::Parameter Interpretation

Grouping Identifier	Number of NodeIDs	Interpretation
> 0	> 0	Remove specified NodeIDs from the specified association group (MANDATORY V1)
> 0	$= 0$	Remove all NodeIDs from the specified association group (RECOMMENDED V1)
$= 0$	$>= 0$	(*Reserved* V1)

CC:0085.01.04.11.006 A sending node MUST NOT send reserved parameter combinations and a receiving node MUST ignore reserved parameter combinations.

3.2.3.8 Association Supported Groupings Get Command

This command is used to request the number of association groups that this node supports.

CC:0085.01.05.11.001 The Association Supported Groupings Report Command MUST be returned in response to this command.

CC:0085.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:0085.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.12: Association Supported Groupings Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_GROUPINGS_GET (0x05)							

3.2.3.9 Association Supported Groupings Report Command

This command is used to advertise the maximum number of association groups implemented by this node.

Table 3.13: Association Supported Groupings Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_GROUPINGS_REPORT (0x06)							
Supported Groupings							

Supported Groupings (8 bits)

This field is used to advertise the number of association groups that this node supports.

CC:0085.01.06.11.001 Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

3.2.4 Association Command Class, version 2

The Association Command Class is used to manage associations to NodeID destinations. A NodeID destination may be a simple device or the Root Device of a Multi Channel device.

The following sections specify commands which were extended or added in version 2.

3.2.4.1 Compatibility Considerations

CC:0085.02.00.21.001 A device supporting this command class version MUST also support the Association Command Class, version 1.

This version introduces - New methods for the removal of associations via the Association Remove Command - New commands

- Association Specific Group Get Command
- Association Specific Group Report Command

The considerations of Section 3.2.3.1 also apply to this version.

3.2.4.2 Z-Wave Plus Considerations

The considerations of Section 3.2.3.2 also apply to this version.

3.2.4.3 Security Considerations

The considerations of Section 3.2.3.3 also apply to this version.

3.2.4.4 Association Remove Command

The Association Remove Command is used to remove NodeID destinations from a given association group.

CC:0085.02.04.11.001 If the node supports the Multi Channel Association Command Class the node MUST NOT remove End Point destinations in response to this command.

Table 3.14: Association Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_REMOVE (0x04)							
Grouping Identifier							
NodeID 1							
...							
NodeID N							

Grouping identifier (8 bits)

This field is used to specify from which association group the specified NodeID destinations are to be removed.

CC:0085.02.04.11.002 This field MUST be interpreted in combination with the NodeID field.

CC:0085.02.04.11.003 A node that receives an unsupported Grouping Identifier MUST ignore this command; with the exception of Grouping Identifier 0, which MUST be accepted. Refer to Table 3.15.

NodeID (N bytes)

This field specifies a list of NodeID destinations that are to be removed. The Grouping Identifier and
CC:0085.02.04.11.004 NodeID fields MUST be interpreted as indicated in Table 3.15:

Table 3.15: Association Remove, V2::Parameter Interpretation

Grouping Identifier	Number of NodeIDs	Interpretation
> 0	> 0	Remove destination NodeIDs from association group (MANDATORY V1, MANDATORY V2)
> 0	$= 0$	Remove all destination NodeIDs from association group (RECOMMENDED V1, MANDATORY V2)
$= 0$	> 0	Remove destination NodeIDs from all association groups (*Reserved* V1, MANDATORY V2)
$= 0$	$= 0$	Remove all destination NodeIDs from all association groups (*Reserved* V1, MANDATORY V2)

3.2.4.5 Association Specific Group Get Command

This command allows a portable controller to interactively create associations from a multi-button device to a destination that is out of direct range.

CC:0085.02.0B.12.001 It is OPTIONAL for a device to support this functionality. However, a receiving device MUST always
CC:0085.02.0B.11.001 return the Association Specific Group Report Command in response to this command. If the device
CC:0085.02.0B.12.002 does not support this functionality, a Group parameter value of 0 SHOULD be advertised in the Association Specific Group Report Command that is returned in response to this command.

This functionality allows a supporting multi-button device to detect a key press and subsequently advertise the identity of the key. The following sequence of events takes place:

- The user activates a special identification sequence and pushes the button to be identified
- The device issues a Node Information frame (NIF)
- The NIF allows the portable controller to determine the NodeID of the multi-button device
- The portable controller issues an Association Specific Group Get Command to the multi-button device
- The multi-button device returns an Association Specific Group Report Command that advertises the association group that represents the most recently detected button

The Association Group Information (AGI) Command Class provides a centralized alternative for the discovery of available association groups and the capabilities of these groups. A device supporting
CC:0085.02.0B.12.003 this functionality SHOULD also support the Association Group Information (AGI) Command Class.

CC:0085.02.0B.11.002 This command MUST NOT be issued via multicast addressing. A receiving node MUST NOT return
CC:0085.02.0B.11.003 a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.16: Association Specific Group Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_SPECIFIC_GROUP_GET (0x0B)							

Clarification: Previous text revisions of this specification presented conflicting guidelines for the Association Specific Group Get Command when received by non-supporting devices.

The values 0 and 1 were both suggested for the Group field of the Association Specific Group Report
CC:0085.02.0B.12.004 Command. It is RECOMMENDED that the value 0 is returned by non-supporting devices.

3.2.4.6 Association Specific Group Report Command

This command is used to advertise the association group that represents the most recently detected button.

Table 3.17: Association Specific Group Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION (0x85)							
Command = ASSOCIATION_SPECIFIC_GROUP_REPORT (0x0C)							
Group							

Group (8 bits)

This field is used to advertise the association group that represents the most recently detected button.

CC:0085.02.0C.11.001 The value of this field MUST be in the range 0..255.

CC:0085.02.0C.11.002 If a supporting device implements a multi-button device, the Group field MUST advertise an association group which represents the most recently activated button. The actual association group SHOULD be able to control an actuator device, e.g. via the Basic Command Class.
CC:0085.02.0C.12.001

CC:0085.02.0C.12.002 This field SHOULD be set to 0 if the functionality is not supported or if the most recent button event does not map to an association group.

Clarification: Previous text revisions of this specification presented conflicting guidelines for the Association Specific Group Get Command when received by non-supporting devices.

CC:0085.02.0C.12.003 The values 0 and 1 were both suggested for the Group field of the Association Specific Group Report Command. It is RECOMMENDED that the value 0 is returned by non-supporting devices.

3.2.5 Association Command Class, version 3

The Association Command Class version 3 introduces the capability to discover the highest security class of a target when issuing controlling commands via association groups.

3.2.5.1 Compatibility Considerations

CC:0085.03.00.21.001 If a node supports the version 3 of this Command Class and the Multi Channel Association Command Class, it MUST support the Multi Channel Association Command Class, version 4 or newer.

3.2.5.2 Interoperability Considerations

CC:0085.03.00.31.001 Any supporting node MUST learn the granted security classes of a destination before sending controlling commands to a destination via association groups.

Commands are marked as controlling or supporting in [25].

CC:0085.03.00.32.001 The following discovery is RECOMMENDED:

1. Request the NIF and read its contents, look for S2/Supervision and S0
2. If S2 is supported, for every granted S2 key starting from the highest:
   a. Issue the S2 encrypted controlling command using Supervision Get encapsulation (and Multi Channel encapsulation if End Point source or destination)
   b. If receiving a Supervision Report, stop the discovery and use the current S2 Security Class for controlling the target NodeID.
   c. Else if no answer is returned (or S2 Nonce Reports), proceed with a lower granted S2 key.
3. If S0 is supported, try using S0:
   a. If S2 was in the NIF :
      i. Issue the S0 encrypted controlling command using Supervision Get encapsulation
      ii. If receiving a Supervision Report, stop the discovery and use S0 as the Security Class to control the target NodeID
   b. If S2 was not in the NIF:
      i. Issue a S0 encrypted S0 Security Command Supported Get command
      ii. If receiving S0 encrypted S0 Security Command Supported Report command, stop the discovery and use S0 for controlling the target NodeID
4. Settle for non-secure level if no Security Class was found. Supervision encapsulation MAY be used non-securely only if it was in the NIF.

CC:0085.03.00.33.001 A supporting node MAY use another algorithm using the following assumptions:

CC:0085.03.00.31.002 - A controlling node MUST NOT associate Node A to a Node B destination that does not support the Command Class that the Node A will be controlling.

CC:0085.03.00.31.003 - A controlling node MUST NOT associate Node A to a Node B destination if Node A was not granted Node B's highest Security Class.

Role Type Specification provides recommended timeouts when waiting for responses to Get type commands.

Further, when issuing (supporting or controlling) commands via an association group:

CC:0085.03.00.31.004 - A supporting node MUST NOT use Supervision encapsulation if the destination does not support the *Supervision Command Class, version 1*.

CC:0085.03.00.31.006 - A supporting node MUST use Supervision encapsulation if the destination supports the Supervision Command Class.

CC:0085.03.00.31.005

- A supporting node MUST NOT use Multi Command encapsulation if the destination does not support the Multi Command Command Class.

3.2.6 Association Command Class, version 4

The Association Command Class version 4 introduces the capability to discover the highest security class of a target when issuing unsolicited supporting commands (unsolicited Reports and Notifications) via association groups.

3.2.6.1 Compatibility Considerations

CC:0085.04.00.21.001 If a node supports the version 4 of this Command Class and the Multi Channel Association Command Class, it MUST support the Multi Channel Association Command Class, version 5 or newer.

3.2.6.2 Interoperability Considerations

CC:0085.04.00.31.001 Any supporting node MUST learn the granted security classes of a destination before sending controlling or unsolicited supporting commands to a destination via association groups.

CC:0085.04.00.32.002 The learned security class SHOULD be saved in the device after a successful learn process to prevent further learn process.

The discovery is considered to be failed if the destination node did not respond on any security class nor non-securely.

CC:0085.04.00.31.007 CC:0085.04.00.32.003 If the discovery fails, the discovery MUST be marked as incomplete and the discovery SHOULD be attempted later.

The supporting node is allowed to use the learned security class to send unsolicited supporting commands on a level lower that its own highest security class.

CC:0085.04.00.31.008 The learned security level can be used only to send unsolicited supporting commands or control commands (Set, see Section 3.2.5.2). The supporting node MUST NOT reply on control commands (with a Report on a Get command) on lower security class than its highest security class even if a lower security class is used for sending unsolicited supporting commands.

Commands are marked as controlling or supporting in [25].

The discovery process is the same for the discovery of control commands (Set commands, see Section 3.2.5.2) and for unsolicited supporting commands.

CC:0085.04.00.32.001 The following discovery is RECOMMENDED:

1. If the destination node B previously used to send a command on the sender node A's highest level, node A MUST use this level for sending unsolicited supporting commands to node B
2. Request the NIF and read its contents, look for S2/Supervision and S0
3. If S2 is supported, for every granted S2 key starting from the highest:
    a. Issue the S2 encrypted controlling or unsolicited supporting command using Supervision Get encapsulation (and Multi Channel encapsulation if End Point source or destination).
    b. If receiving a Supervision Report, stop the discovery and use the current S2 Security Class for controlling or sending unsolicited supporting commands to the target NodeID.
    c. Else if no answer is returned (or S2 Nonce Reports), proceed with a lower granted S2 key.
4. If S0 is supported, try using S0:
    a. If S2 was in the NIF :
        i. Issue the S0 encrypted controlling or unsolicited supporting command using Supervision Get encapsulation
        ii. If receiving a Supervision Report, stop the discovery and use S0 as the Security Class for controlling or sending unsolicited supporting commands to the target NodeID
    b. If S2 was not in the NIF:

i. Issue a S0 encrypted S0 Security Command Supported Get command

ii. If receiving S0 encrypted S0 Security Command Supported Report command, stop the discovery and use S0 for controlling or sending unsolicited supporting commands to the target NodeID

5. Settle for non-secure level if no Security Class was found. Supervision encapsulation MAY be used non-securely only if it was in the NIF.

CC:0085.04.00.33.001 A supporting node MAY use another algorithm using the following assumptions:

CC:0085.04.00.31.003 • A controlling node MUST NOT associate Node A to a Node B destination if Node A was not granted Node B's highest Security Class.

Role Type Specification provides recommended timeouts when waiting for responses to Get type commands.

Further, when issuing (supporting or controlling) commands via an association group:

CC:0085.04.00.31.004 • A supporting node MUST NOT use Supervision encapsulation if the destination does not support the *Supervision Command Class, version 1*

CC:0085.04.00.31.006 • A supporting node MUST use Supervision encapsulation if the destination supports the Supervision Command Class

CC:0085.04.00.31.005 • A supporting node MUST NOT use Multi Command encapsulation if the destination does not support the Multi Command Command Class.

3.2.7 Association Command Configuration Command Class, version 1

The Association Command Configuration Command Class defines the commands necessary for a 2nd node to add and delete commands to NodeIDs in a group as defined in the Association Command Class in a 1st node.

The device MUST implement the Association Command Class as 'supported'

The Association Command Class and the Command Configuration Command Class MUST be linked through the following dependencies:

- Nodes added to an association through an Association Set or Association Composite Set MUST be reported in Command Configuration Reports with the Command Class and command identifiers transmitted to the nodes.
- Nodes added to an association through a Command Configuration Set MUST also be reported in an Association Report and Association Composite Report
- All commands associated to a grouping identifier/NodeID pair will be removed as a result of an Association Remove. The related command records will be released
- Command(s) associated to a grouping identifier/NodeID/endpoint pair will be removed as a result of an Association Composite remove. The related command(s) record will be released.

The memory consumption of supporting full command sizes in all combinations of groupings identifiers and Node IDs is very extensive; hence the command class supports a memory flexible implementation.

The Command Class allows a device to support a number of command records. A command record consists of the grouping identifier, the Node ID and the command. The size of the command MAY be restricted by the device through the Max command length field. The command MUST be the complete command needed (I.e. All relevant encapsulations MUST be included in the command).

When no command records are free (all has been used), new commands MUST NOT be allocated to Node IDs before one or more command records have been freed up (through the Association Remove Command)

If the 2nd node runs out of free command records before it has finalized its command configurations, it MUST accept that the application on the device has full control of the remaining Node IDs. Alternatively the 2nd node can abort the command configuration process. In this case it is RECOMMENDED that the 2nd node free up the used command records in the aborted command configuration attempt.

A device can report the maximum number of command records, the number of free command records, and the max command length supported through the Command Records Supported Report.

In order to support sharing knowledge of how a device controls nodes without using extensive memory resources a Configurable Cmd field is supported. When configurable Cmd= 0x0 then a 2nd node can only monitor the commands. It cannot control them.

The V/C field allows a device to decrease the memory utilization to a minimum. When a device reports V/C=0x01, then the Command Configuration Set and Command Configuration Report MUST always use command class identifier and command identifier equal to a Basic Set Command Records Supported Get. This allows the device to only store the value field and thereby save memory resources.

3.2.7.1 Command Records Supported Get Command

The Command Records Supported Get Command is used to request the number of free command records available (grouping Identifier, Node ID, command), the maximum command records supported in the device and information regarding the maximum command length supported in the device.

The maximum number of groupings the given node supports is available through the Association Command Class.

The Command Records Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.18: Command Records Supported Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION (0x9B)							
Command = COMMAND_RECORDS_SUPPORTED_GET (0x01)							

3.2.7.2 Command Records Supported Report Command

The Command Records Supported Report Command used to report information regarding the Command records.

Table 3.19: Command Records Supported Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION (0x9B)							
Command = COMMAND_RECORDS_SUPPORTED_REPORT (0x02)							
Max command length						V/C	Conf. Cmd
Free Command records 1							
Free Command records 2							
Max Command records 1							
Max Command records 2							

Configurable Cmd (1 bit)

Table 3.20: Configurable Command bit

Configurable Cmd	Functionality
0	The local application has full control of the commands associated with the grouping. The commands can be monitored from a 2nd network node.
1	The specific commands associated with the grouping can be controlled and monitored from a 2nd network node. This option includes also the level field used by the command *Transfer Scene* in the *Controller Replication Command Class*. In this case the level value is transferred via the command *Basic Set* in the *Basic Command Class*.

V/C (1 bit)

Table 3.21: V/C bit

V/C	Functionality
0	Command type. A Z-Wave command can be added to the node
1	Value type. A Value field is specified for a Node ID using the command *Basic Set* in the *Basic Command Class*. Level field originates from the command *Transfer Scene* in the *Controller Replication Command Class*.

Max command length (6 bits)

If Configurable Cmd is equal to 0x0, the field SHOULD return 0x0.

If Configurable Cmd is equal to 0x1 the field SHOULD return the maximum length of a command which can be associated to a node in a grouping. The minimum max command length allowed is 0x03.

Example:

A product reports a Max command length = 0x03. In this case the product can be programmed with a Multilevel Switch Set, but not a Switch Multilevel Start Level Change.

Multilevel Switch Set has a command length of 3

Multilevel Switch Start Level Change has a command length of 4

Free Command Records (16 bits)

The field specifies the current number of free Command Records which can be configured in the device through the Command Configuration Set Command.

Max Command records (16 bits)

The field specifies the maximum number of Command Records which can be configured in the device through the Command Configuration Set Command.

3.2.7.3 Command Configuration Set Command

The Command Configuration Set Command used to specify which commands SHOULD be sent to nodes within a given Grouping identifier.

Every Command Configuration Set will utilize one Command record from the pool of free Command records.

If the device has no free command records when receiving the Command Configuration Set, the command will be ignored.

The Application on the node may alter the commands when needed.

Table 3.22: Command Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION (0x9B)							
Command = COMMAND_CONFIGURATION_SET (0x03)							
Grouping identifier							
Node ID							
Command length							
Command Class identifier							
Command identifier							
Command byte 1							
...							
Command byte n							

Grouping identifier (8 bits)

This grouping identifier used to specify how nodes are grouped together. The grouping identifier values MUST be a sequence starting from 1. This field MAY be ignored in case the node only supports one group.

Node ID (8 bits)

This field contains the node ID within the grouping specified, that should receive the command.

Command length (8 bits)

This field specifies the complete command length (including command class and command identifiers).

Example: Switch Multilevel Set 0x20. The command length field MUST be equal to 0x03.

Command Class Identifier (8 bits)

This field contains the identifier of the command class which should be sent to the Node ID. In case the Configurable Cmd field is equal to 0x0 is this field and the following ignored and can therefore be omitted.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set command.

Command identifier (8 bits)

This field contains the identifier of the Command, which should be sent to the specified Node ID. In case the Configurable Cmd field is equal to 0x0 is this field ignored and can therefore be omitted.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set Command.

Command byte (N bytes)

These fields contain the command parameters, which should be sent to the Node ID. In case the Configurable Cmd field is equal to 0x0 is these fields ignored and can therefore be omitted.

3.2.7.4 Command Configuration Get Command

The Command Configuration Get Command is used to request the commands specified for to a Node ID within a given Grouping identifier.

The Command Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.23: Command Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION (0x9B)							
Command = COMMAND_CONFIGURATION_GET (0x04)							
Grouping identifier							
Node ID							

Grouping identifier (8 bits)

This group identifier used to specify how nodes are grouped together. The group identifier values MUST be a sequence starting from 1. This field MUST be ignored in case the node only supports one group.

Node ID (8 bits)

This field specifies the node ID within the grouping.

3.2.7.5 Command Configuration Report Command

The Command Configuration Report Command used to report the commands specified for a Node ID within a given Grouping identifier.

Table 3.24: Command Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION (0x9B)							
Command = COMMAND_CONFIGURATION_REPORT (0x05)							
Grouping identifier							
Node ID							
First	Reserved			Reports to follow			
Command length							
Command Class identifier							
Command identifier							
Command byte 1							
...							
Command byte N							

Grouping identifier (8 bits)

This group identifier identifies the group. The group identifier values MUST be a sequence starting from 1.

Node ID (8 bits)

This field contains the node ID as requested in the Association Command Get.

Reports to follow (4 bits)

This value indicates how many report frames left before transferring the entire list of commands.

First (1 bit)

This field indicates that this report is the first report relating to a Grouping identifier/Node ID pair

Command length (8 bits)

This field specifies the complete command length (including command class and command identifiers).

Example: Multilevel Switch Set 0x20. The command length field MUST be equal to 0x03.

Command Class Identifier (8 bits)

This field contains the identifier of the command class which is sent to the Node ID.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set command

Command identifier (8 bits)

This field contains the identifier of the Command which is sent to the specified Node ID.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set Command

Command byte (N bytes)

These fields contain the command parameter which is sent to the Node ID.

3.2.8 Association Group Information (AGI) Command Class, version 1

The Association Group Information (AGI) Command Class allows a node to advertise the capabilities of each association group supported by a given application resource.

CC:0059.01.00.12.001 Controllers and installer tools SHOULD use AGI information to support controller-assisted button-to-button association and GUI-based drag-and-drop association in a plug-and-play fashion.

Centralized gateway-based deployments may create a single association from the lifeline association group to a central management application.

3.2.8.1 Compatibility considerations

CC:0059.01.00.21.001 A node supporting the AGI Command Class MUST support the Association Command Class.

3.2.8.1.1 Multi Channel considerations

The Association Group Information (AGI) Command Class also applies to Multi Channel devices.

If a node implements Multi Channel End Points and supports the AGI Command Class, each indi-
CC:0059.01.00.21.002 vidual End Point MUST support the AGI Command Class.

3.2.8.2 Association Principles

A light control transmitter may have one or more keys. A key is mapped to one or more association groups.

Figure 3.1: Light Control Transmitter (example)

Figure 3.2: Lamp Module (example)

A light control transmitter key can be configured to control a lamp by adding the NodeID of the lamp to an association group that represents the key.

3.2.8.3 The Association Group Information

A node may implement one or more association groups. If the device implements association groups,
CC:0059.01.00.12.002 the device SHOULD provide an Association Group Information (AGI) table as described in Table 3.25.

Table 3.25: Layout of One Line of the Association Group Information Table

Group identifier	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8) M bytes

An AGI table entry carries a number of fields. The information is typically static and can be defined at compile time. Thus, the AGI table does not need to occupy any RAM or non-volatile storage.

3.2.8.3.1 Group identifier

The Group identifier indicates which association group the actual table entry relates to. As association groups are always numbered in a sequence starting from 1, an actual implementation does not have to store the group identifier in its memory. The number of association groups may be requested via the Association Groupings Get or Multi Channel Association Groupings Get commands.

CC:0059.01.00.11.001 Association group 1 is reserved for the Z-Wave Plus Lifeline association group. Group 1 MUST NOT be assigned to any other use than the Lifeline group. The actual Device Type specifies a mandatory
CC:0059.01.00.11.002 list of commands which the device MUST send to all targets associated to the Lifeline group. A
CC:0059.01.00.13.001 manufacturer MAY add additional commands to the Lifeline group. Refer to [34] and *Device Type v2 Specification*.

CC:0059.01.00.11.003 The Lifeline group MUST be advertised for the Root Device of a Multi Channel device. A Multi
CC:0059.01.00.12.003 Channel End Point SHOULD advertise commands for association group 1 which End Points can send via the Root Device Lifeline group if a Multi Channel Association is created from the Root Device Lifeline group.

3.2.8.3.2 Profile

CC:0059.01.00.13.002 A device MAY implement one or several AGI profiles. The profile defines the scope of events which triggers the transmission of commands to the actual association group. As an example a temperature sensor may issue different Basic Set Command parameters when the temperature exceeds a threshold and when the temperature drops below a threshold. The actual behavior is application dependent and out of scope of the AGI Command Class.

The profile identifiers are referenced in Table 3.36 - *AGI Profiles*. Profiles are divided into categories in the following sections.

3.2.8.3.3 General profiles

The “General” category comprises the “Not Applicable” and “Lifeline” profiles. The “Not Applicable”
CC:0059.01.00.11.004 profile identifier MUST be advertised if the actual association group does not match any of the defined profiles.

CC:0059.01.00.11.005 The “Lifeline” profile identifier MUST be advertised for association group 1 of the Root Device.

3.2.8.3.4 Control profiles

"Control" profiles are intended for association groups of which commands are triggered by a user control mechanism, such as keys or buttons. As an example, a light control transmitter may comprise two control keys that can each control a group of lamps.

A control key may control up and down dimming as well as on/off state. Thus, a control key may send more than one command with one specific parameter. A control key may be implemented as a physical key, a group of icons, touch screen gestures or other means. The actual implementation is out of scope of this command class.

CC:0059.01.00.13.003 In case of multiple logical functions in a device, the device MAY implement one Multi Channel End Point for each logical function; e.g. each push button is implemented as a separate End Point. Table 3.26 gives an example of a Multi Channel a battery-powered two-button light control device.

Table 3.26: Example: AGI Tables for Two-Button Light Control Transmitter

Root Device:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	General: Lifeline	Central Scene Notification Notification Report Battery Report Device Reset Locally Notification	Lifeline
2	Control: Key1	Basic Set	On/Off control (Button 1)
3	Control: Key1	Multilevel Switch Set	Dimmer control (Button 1)
4	Control: Key2	Basic Set	On/Off control (Button 2)
5	Control: Key2	Multilevel Switch Set	Dimmer control (Button 2)
End Point 1:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	General: Lifeline	-	Lifeline
2	Control: Key1	Basic Set	On/Off control (Button 1)
3	Control: Key1	Multilevel Switch Set	Dimmer control (Button 1)
End Point 2:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	General: Lifeline	-	Lifeline
2	Control: Key2	Basic Set	On/Off control (Button 1)
3	Control: Key2	Multilevel Switch Set	Dimmer control (Button 1)

The Profile identifiers Control:Key 1 and Control:Key 2 allows an installer tool to determine which Multi Channel End Point and which association group to use in order to configure buttons to control a light dimmer. The Group Name allows a human user to make a qualified decision even if the installer tool is not, e.g. because it is running an older version not updated with newer profile identifiers.

End Points 1 and 2 in the above example do not support any Command Class that must be sent via the Z-Wave Plus Lifeline. Therefore End Points 1 and 2 advertise an empty Command Class list for the Lifeline association group.

The Root Device in the above example advertises the Basic Set Command in association group 2.
CC:0059.01.00.12.004 This is a feature of End Point 1. A controlling node SHOULD NOT create associations from other Root Device association groups than the Lifeline group if it supports the Multi Channel Command Class. The purpose of the Root Device Association Groups (with the exception of the Lifeline group) is only to provide backwards compatibility for legacy devices without support for the Multi Channel Command Class.

3.2.8.3.5 Sensor profiles

"Sensor" profiles are intended for association groups of which commands are triggered by sensor readings. As an example, a sensor product could comprise two temperature sensors.

Sensor profile identifiers are constructed by prepending the Multilevel Sensor Command Class identifier (referred to with AGI_PROFILE_SENSOR) to a multilevel sensor type defined in the Multilevel Sensor Command Class. The values in Table 8 only serve as examples to illustrate the construction of AGI Sensor Profile Identifiers. For the full list of available sensor types, refer to the "Multilevel Sensor Report Command".

CC:0059.01.00.12.005 In case of multiple logical functions in a device, the device SHOULD implement one Multi Channel End Point for each logical function; e.g. each sensor instance is implemented as a separate End Point. Table 3.27 gives an example of a Multi Channel battery-powered sensor device with two multilevel sensor functions:

Table 3.27: Example: AGI Tables for Two-Function Sensor

Root Device:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	General: Lifeline	Multilevel Sensor Report Notification Report Battery Report Device Reset Locally Notification	Lifeline
2	Sensor: Temperature	Basic Set	On/Off control (Indoor temperature)
3	Sensor: Temperature	Basic Set	On/Off control (Outdoor temperature)
End Point 1			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	Sensor: Temperature	Multilevel Sensor Report	Indoor temperature via Lifeline
2	Sensor: Temperature	Basic Set	On/Off control (Indoor temperature)
End Point 2			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	Sensor: Temperature	Multilevel Sensor Report	Outdoor temperature via Lifeline
2	Sensor: Temperature	Basic Set	On/Off control (Outdoor temperature)

The Profile identifier is Sensor:Temperature for both End Points but the Group Name allows a user to determine which End Point and which association group to use in order to configure the indoor temperature sensor to control a heating element.

The Root Device in the above example advertises the Multilevel Sensor Report command in the Lifeline group. Residing in a Multi Channel device, the Root Device does not actually implement any application functionality. The Multilevel Sensor Report Command is a feature of the End Points which is advertised for backwards compatibility with legacy devices not supporting the Multi Channel Command Class.

Likewise, the Root device advertises the Basic Set Command in association groups 2 and 3. This is also a feature of the End Points. Refer to the Multi Channel Command Class for details on backwards compatibility.

End Point 1 in the above example advertises the Multilevel Sensor Report Command in association group 1. By advertising that zero NodeIDs are supported for association group 1, End Point 1 indicates that this command is sent via the Root Device Lifeline group if a Multi Channel association is created for the Root Device Lifeline group. The same principle applies to End Point 2.

3.2.8.3.6 Notification profiles

"Notification" profiles are intended for association groups of which commands are triggered by detected events or state changes. As an example, a detector product could comprise one smoke sensor and one CO2 detector.

Notification profile identifiers are constructed by prepending the Notification Command Class identifier (referred to with AGI_PROFILE_NOTIFICATION) to a notification type defined in the Notification Command Class. The values below only serve as an example to illustrate the construction of AGI Notification Profile identifiers. For the full list of available notification types, refer to the Notification Command Class (Section 2).

CC:0059.01.00.12.006 In case of multiple logical functions in a device, the device SHOULD implement one Multi Channel End Point for each logical function.

The following example outlines how a standalone smoke alarm is represented by an AGI table:

Table 3.28: Example: AGI Table for One-Function Alarm Device

Root Device:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	General: Lifeline	Notification Report Battery Report Device Reset Locally Notification	Lifeline
2	Notification	Basic Set	On/Off control (Smoke)

The Profile identifier Notification SmokeAlarm indicates to an installer that the Basic Set Command of association group 2 is issued in response to smoke alarm events.

3.2.8.3.7 Command Class and Command

CC:0059.01.00.13.004 This field contains the command class and the command that is sent to targets associated with this association group if an event occurs. This field MAY advertise a list of commands and MAY contain a combination of extended and normal command classes. Refer to Section 3.2.8.9.

The command list is used by a controlling application to discover what functionality the node provides via a given association group. It also allows the controlling node to ensure that the associated node understands the received commands.

CC:0059.01.00.13.005 A node MAY list only one command if the actual association group sends different commands which actually relate to the same overall functionality and belong to the same Command Class and Commands Class version (e.g. Multilevel Switch Set / Multilevel Switch Start Level Change / Multilevel Stop Level Change)

CC:0059.01.00.12.007 In case Multi Channel End Points are implemented, the Lifeline association group (group 1) of End Points SHOULD be used to advertise commands that will be sent via the Root Device Lifeline group if a Multi Channel association is created from the Root Device Lifeline group.

3.2.8.3.8 Association group name

This field contains a name reflecting the purpose of the group, e.g. "On/Off control (Smoke)".

CC:0059.01.00.11.006 The Root Device Lifeline group MUST be named "Lifeline".

CC:0059.01.00.11.007 Group names MUST be encoded using UTF-8. The available capacity for characters depends on the actual characters encoded with UTF-8. Plain ASCII characters only occupy one byte while special characters may need two or more bytes for representation.

3.2.8.4 Association Group Name Get

This command is used to query the name of an association group.

CC:0059.01.01.11.001 The Association Group Name Report Command MUST be returned in response to this command.

CC:0059.01.01.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.29: Association Group Name Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO (0x59)							
Command = ASSOCIATION_GROUP_NAME_GET (0x01)							
Grouping Identifier							

Grouping Identifier (1 byte)

This field is used to specify the requested association group identifier.

CC:0059.01.01.12.001 A node receiving this command for an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

3.2.8.5 Association Group Name Report

This command is used to advertise the assigned name of an association group.

Table 3.30: Association Group Name Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO (0x59)							
Command = ASSOCIATION_GROUP_NAME_REPORT (0x02)							
Grouping Identifier							
Name Length							
Name 1							
...							
Name N							

Grouping Identifier (1 byte)

This field is used to advertise the actual association group identifier.

Name Length (1 byte)

CC:0059.01.02.11.001 This field indicates the length in bytes of the Name field. The value MUST be in the range 0..42.

Name (N bytes)

This field is used to indicate the assigned name for the actual Group Identifier.

CC:0059.01.02.11.002 The length of this field in bytes MUST comply with the advertised value in the Name Length field.

CC:0059.01.02.11.003 The string MUST NOT contain any appended termination characters.

CC:0059.01.02.11.004 The characters MUST be encoded in UTF-8 format.

3.2.8.6 Association Group Info Get

This command is used to request the properties of one or more association group.

CC:0059.01.03.11.001 The Association Group Info Report MUST be returned in response to this command.

CC:0059.01.03.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.31: Association Group Info Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO (0x59)							
Command = ASSOCIATION_GROUP_INFO_GET (0x03)							
Refresh cache	List Mode	Reserved					
Grouping Identifier							

List Mode (1 bit)

This field is used to request the properties of the supported association groups of a node.

CC:0059.01.03.11.003 If List Mode is set to 1, a receiving node MUST ignore the Grouping Identifier field and return the
CC:0059.01.03.13.001 properties of all its supported associations groups. The receiving node MAY return the response in several Reports, (e.g. due to memory constraints or a high number of association groups).

CC:0059.01.03.11.004 If List Mode is set to 0, a receiving node MUST advertise the properties of the association group identified by the Grouping Identifier.

CC:0059.01.03.11.005 The Association Group Info Report returned in response to this command MUST advertise the same List Mode value as specified by this field.

Refresh Cache (1 bit)

CC:0059.01.03.11.006 If AGI information is transferred via a gateway, the gateway MUST cache information for all nodes; also listening nodes.

The AGI Get command flag "Refresh Cache" is used by a management application to instruct a gateway to update its cache on the first chance given. In case of a sleeping device, this may require a user operation to wake up the device. In case of a Wake Up device, the gateway may wait for the next Wake Up Notification from the actual device.

CC:0059.01.03.11.007 The "Refresh Cache" value 0 MUST specify that the gateway is to return its cached information.

CC:0059.01.03.11.008 The "Refresh Cache" value 1 MUST specify that the gateway is to return its cached information and update its cache on the first chance given.

CC:0059.01.03.12.001 The "Refresh Cache" SHOULD be set to 0 by a sending node.

A receiving node may have re-configurable AGI properties, e.g. if it is a configurable multi-purpose
CC:0059.01.03.13.002 remote control. If the "Dynamic Info" flag of the AGI Report command is set to 1, a sending node MAY set the "Refresh Cache" flag in the AGI Get command to force the gateway to update its cache, e.g. after the sending node has re-configured the AGI properties of the receiving node.

Reserved

CC:0059.01.03.11.009 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Grouping Identifier (1 byte)

CC:0059.01.03.11.00A This field is used to specify the requested association group identifier. This value MUST be ignored if the List Mode field is set to 1.

CC:0059.01.03.12.002 A node receiving this command for an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

3.2.8.7 Association Group Info Report

This command is used to advertise the properties of one or more association groups.

Table 3.32: Association Group Info Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO (0x59)							
Command = ASSOCIATION_GROUP_INFO_REPORT (0x04)							
List mode	Dynamic Info	Group Count					
Grouping Identifier							
Mode = 0							
Profile MSB							
Profile LSB							
Reserved = 0							
Event Code MSB = 0							
Event Code LSB = 0							
...							
Grouping Identifier							
Mode = 0							
Profile MSB							
Profile LSB							
Reserved = 0							
Event Code MSB = 0							
Event Code LSB = 0							

List Mode (1 bit)

CC:0059.01.04.11.001 The List Mode field MUST advertise the same value as specified in the Association Group Info Get command which caused this command to be returned.

CC:0059.01.04.13.001 If List Mode is 1, a sending node is advertising the properties of all association groups. The sending node MAY return several Reports.

CC:0059.01.04.11.002 If List Mode is 0 a sending node MUST advertise the properties of one association group.

Dynamic Info (1 bit)

CC:0059.01.04.13.002 If the Dynamic Info field is set to 1, the information MAY change and a controlling node SHOULD
CC:0059.01.04.11.003 perform periodic cache refresh for this node. Nodes MUST set this bit if they are able to change the Association Group Information on the fly.

CC:0059.01.04.11.004 If this field is set to 0, a controlling node MUST NOT request this information more than one time.

Group Count (6 bits)

CC:0059.01.04.11.005 This field indicates the number of association groups advertised in the command. The Grouping Identifier, Mode, Profile, Reserved and Event Code fields MUST be repeated for each association group.

CC:0059.01.04.11.006 If List Mode is set to 0, the Group Count field MUST be set to 1.

If List Mode is set to 1, the Group Count field MUST advertise the number of association group property blocks following this field.

A node receiving this report may determine the total number of association groups via the Association Supported Groupings Get or Multi Channel Association Supported Groupings Get commands.

Grouping Identifier (1 byte)

CC:0059.01.04.11.007 This field MUST advertise the association group identifier of the actual association group property block.

Mode (1 byte)

CC:0059.01.04.11.008 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Profile (2 bytes)

CC:0059.01.04.11.009 This field is used to advertise the profile of the actual association group. Refer to Section 3.2.8.3.2 and Table 3.36. This field MUST be encoded according to Table 3.36.

Reserved

CC:0059.01.04.11.00A This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Event Code (2 bytes)

CC:0059.01.04.11.00B This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

3.2.8.8 Association Group Command List Get

This command is used to request the commands that are sent via a given association group.

CC:0059.01.05.11.001 The Association Group Command List Report MUST be returned in response to this command.

CC:0059.01.05.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.33: Association Group Command List Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO (0x59)							
Command = ASSOCIATION_GROUP_COMMAND_LIST_GET (0x05)							
Allow cache	Reserved						
Grouping Identifier							

Allow Cache (1 bit)

This field indicates that a Z-Wave Gateway device is allowed to intercept the request and return a cached response on behalf of the specified target.

CC:0059.01.05.11.003 If this bit is not set to 1, a Z-Wave Gateway device MUST forward the request to the specified target.

CC:0059.01.05.12.001 A requesting node SHOULD allow caching to save network bandwidth.

CC:0059.01.05.11.004 A Z-Wave Gateway device MUST cache information for all nodes; also listening nodes. This will save network bandwidth.

CC:0059.01.05.11.005 A Z-Wave Gateway device MUST forward the request if it does not hold cached information for the specified target. The Z-Wave Gateway device MUST cache the data returned.

Reserved

CC:0059.01.05.11.006 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Grouping Identifier (1 byte)

This field is used to specify the requested association group identifier.

CC:0059.01.05.12.002 A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

3.2.8.9 Association Group Command List Report

This command is used to advertise the commands that are sent via an actual association group.

Table 3.34: Association Group Command List Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO (0x59)							
Command = ASSOCIATION_GROUP_COMMAND_LIST_REPORT (0x06)							
Grouping Identifier							
List Length							
Command Class 1 (1 or 2 bytes)							
Command 1							
...							
Command Class N (1 or 2 bytes)							
Command N							

Grouping Identifier (1 byte)

This field is used to advertise the actual association group identifier.

List Length (1 byte)

This field advertises the length in bytes of the command list (Command Class and Command fields).

Command Class and Command

CC:0059.01.06.13.001 Command Classes MAY be normal or extended Command Classes.

Normal Command Classes are represented as one byte while extended Command Classes are represented as two bytes. Thus, a command list entry (command class + command) is either 2 or 3 bytes long.

The receiving node must parse individual command list entries to determine if the individual Command Class is a normal or an extended Command Class.

The first byte of normal command classes is in the range 0x20 – 0xEE, while the first byte of extended command classes is in the range 0xF1 – 0xFF.

CC:0059.01.06.11.001 A sending node MUST NOT add any command payload in this field.

CC:0059.01.06.11.002 Command Classes already containing destination NodeID such as Wake Up Command Class MUST NOT be listed in the Association Group Command List Report. This also means that the Wake Up Command Class MUST NOT be covered by the Lifeline association group.

3.2.9 Association Group Information (AGI) Command Class, version 2

The Association Group Information (AGI) Command Class allows a node to advertise the capabilities of each association group supported by a given application resource.

3.2.9.1 Compatibility considerations

The Association Group Information (AGI) Command Class, version 2 introduces the Profile category "Meter".

The Association Group Information (AGI) Command Class, version 2 defines no new command fields or changes to the interpretation of Association Group Information (AGI) Command Class, version 1.

All sections and commands not described in this version remain unchanged from version 1.

3.2.9.2 The Association Group Information

CC:0059.02.00.13.001 A device MAY implement one or more association groups. If the device implements association groups,
CC:0059.02.00.12.001 the device SHOULD provide an Association Group Information (AGI) table as described in in Table 3.25.

3.2.9.2.1 Profile

The profile identifiers are referenced in Table 3.36. The profiles category "Meter" introduced in version 2 is described in Section 3.2.9.2.2.

3.2.9.2.2 Meter profiles

"Meter" profiles are intended for association groups of which commands are triggered by meter readings. As an example, a two-phase meter product could comprise three electric meters: one for each phase and one for the total consumption.

Meter profile identifiers are constructed by prepending the Meter Command Class identifier (referred to with AGI_PROFILE_METER) to a meter type defined in the Meter Command Class. The values in Table 8 only serve as examples to illustrate the construction of AGI Meter Profile identifiers. For the full list of available meter types, refer to the Meter Command Class.

CC:0059.02.00.12.002 In case of multiple logical functions in a device, the device SHOULD implement one Multi Channel End Point for each logical function; e.g. each meter instance. Table 7 gives an example of a Multi Channel Meter device with two normal and one aggregated meter functions.

Table 3.35: Example AGI Tables for Three-Function Meter

Root Device:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	General: Lifeline	Meter Report Device Reset Locally Notification	Lifeline
2	Meter:Electric	Basic Set	On/Off control (Phase 1)
3	Meter:Electric	Basic Set	On/Off control (Phase 1)
4	Meter:Electric	Basic Set	On/Off control (Total Consumption)
End Point 1:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	Meter:Electric	Meter Report	Phase 1 via Lifeline
2	Meter:Electric	Basic Set	On/Off control (Phase 1)
End Point 2:			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	Meter:Electric	Meter Report	Phase 1 via Lifeline
2	Meter:Electric	Basic Set	On/Off control (Phase 1)
End Point 3 (aggregated):			
Group identifier	**Profile 2 bytes**	**Command Class & Command (list) N bytes**	**Group Name (UTF-8) M bytes**
1	Meter:Electric	Meter Report	Total Consumption via Lifeline
2	Meter:Electric	Basic Set	On/Off control (Total Consumption)

The Profile identifier is Meter:Electric for all End Points but the Group Name allows a user to determine which End Point and which association group to use in order to configure the phase 2 meter to control an alarm lamp.

The Root Device in the above example advertises the Meter Report Command in the Lifeline group. Residing in a Multi Channel device, the Root Device does not actually implement any application functionality. The Meter Report command is a feature of the End Points which is advertised for backwards compatibility with legacy devices not supporting the Multi Channel Command Class.

Likewise, the Root device advertises the Basic Set Command in association groups 2..4. This is also a feature of the End Points. Refer to the Multi Channel Command Class for details on backwards compatibility.

End Points 1..3 in the above example advertise the Meter Report Command in association group 1. By advertising that zero NodeIDs are supported for association group 1, End Points indicate that this command is sent via the Root Device Lifeline group if a Multi Channel association is created for the Root Device Lifeline group.

3.2.10 Association Group Information (AGI) Command Class, version 3

The Association Group Information (AGI) Command Class allows a node to advertise the capabilities of each association group supported by a given application resource.

3.2.10.1 Compatibility considerations

The Association Group Information (AGI) Command Class, version 3 introduces the Profile category "Irrigation".

The Association Group Information (AGI) Command Class, version 3 defines no new command fields or changes to the interpretation of Association Group Information (AGI) Command Class, version 1 and version 2.

All sections and commands not described in this version remain unchanged from version 2.

3.2.10.2 The Association Group Information

A node may implement one or more association groups. If the device implements association groups,
CC:0059.03.00.12.001 the device SHOULD provide an Association Group Information (AGI) table as described in Table 3.25

3.2.10.2.1 Profile

The profile categories are referenced in Table 3.36. The profiles category "Irrigation" introduced in version 3 is described in Section 3.2.10.2.2.

3.2.10.2.2 Irrigation Profiles

"Irrigation" profiles are intended for association groups of which commands are triggered by irrigation events. As an example, an irrigation control device may provide 8 channels that can each control an external resource such as a valve or a pump.

Table 3.36: AGI Profiles

Profile	Explanation	Profile identifier	
		MSB	LSB
General:NA (v1)	"Not Applicable" There is no specific class of events for this association group	AGI_PROFILE_GENERAL = 0x00	AGI_GENERAL_NA = 0x00
General: Lifeline (v1)	"Lifeline" This association group is intended for all events relevant for the Lifeline group	AGI_PROFILE_GENERAL = 0x00	AGI_GENERAL_LIFELINE = 0x01
Control:Key01 (v1)	"Control Key 1" Members of this association group are controlled or receive reports in response to user input for key 1	AGI_PROFILE_CONTROL = 0x20	AGI_CONTROL_KEY01 = 0x01
Control:Key xx (v1)	…		
Control:Key32 (v1)	"Control Key 32" Members of this association group are controlled or receive reports in response to user input for key 32	AGI_PROFILE_CONTROL = 0x20	AGI_CONTROL_KEY32 = 0x20
Sensor: Air temperature (v1)	"Sensor, Air Temperature" Members of this association group are controlled or receive reports when the sensor value changes	AGI_PROFILE_SENSOR = 0x31	MULTILEVEL_SENSOR_TYPE_TEMPERATURE = 0x01
Sensor: Humidity (v1)	"Sensor, Humidity" Members of this association group are controlled or receive reports when the sensor value changes	AGI_PROFILE_SENSOR = 0x31	MULTILEVEL_SENSOR_TYPE_HUMIDITY = 0x05
…	(only examples above. Sensor profiles are built based on the Sensor Type as described in Section 3.2.8.3.5	AGI_PROFILE_SENSOR = 0x31	MULTILEVEL_SENSOR_TYPE_TEMPERATURE = 0x01
Notification: Smoke Alarm (v1)	"Notification, Smoke Alarm" Members of this association group are controlled or receive reports when an event or state change is detected.for the given Notification Type	AGI_PROFILE_NOTIFICATION = 0x71	NOTIFICATION_TYPE_SMOKE = 0x01
Notification: C02 Alarm (v1)	"Notification, CO2 Alarm" Members of this association group are controlled or receive reports when an event or state change is detected.for the given Notification Type	AGI_PROFILE_NOTIFICATION = 0x71	NOTIFICATION_TYPE_CO2 = 0x03
…	(only examples above. Notification profiles are built based on the Notification Type as described in Section 3.2.8.3.6.	AGI_PROFILE_NOTIFICATION = 0x71	…

Table 3.37: AGI Profiles Continued

Profile	Explanation	Profile identifier	
		MSB	LSB
Meter: Electric, kWH (v2)	"Meter, Electric" Members of this association group receive meter reports or are controlled when a metering event is detected.	AGI_PROFILE_METER = 0x32	METER_TYPE_ELECTRIC = 0x01
Meter: Gas (v2)	"Meter, Gas" Members of this association group receive meter reports or are controlled when a metering event is detected.	AGI_PROFILE_METER = 0x32	METER_TYPE_GAS = 0x02
Meter: Water (v2)	"Meter, Water" Members of this association group receive meter reports or are controlled when a metering event is detected.	AGI_PROFILE_METER = 0x32	METER_TYPE_WATER = 0x03
…	(only examples above. Meter profiles are built based on the Notification Type as described in Section 3.2.9.2.2.	AGI_PROFILE_METER = 0x32	MULTILEVEL_SENSOR_TYPE_TEMPERATURE = 0x01
Irrigation: Channel 01 (v3)	"Irrigation Channel 01" Member(s) of this association group are controlled by channel 1 of an irrigation control device	AGI_PROFILE_IRRIGATION = 0x6B	AGI_IRRIGATION_CHANNEL_01 = 0x01
Irrigation: Channel xx (v3)	…		
Irrigation:	"Irrigation Channel 32" Members of this association group are controlled or receive reports in response to user input for key 32	AGI_PROFILE_IRRIGATION = 0x6B	AGI_IRRIGATION_CHANNEL_32 = 0x20

3.2.11 Battery Command Class, version 1

The Battery Command Class is used to request and report battery levels for a given device.

3.2.11.1 Battery Get Command

The Battery Get Command is used to request the level of a battery.

CC:0080.01.02.11.001 The *Battery Report Command* MUST be returned in response to this command.

CC:0080.01.02.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.38: Battery Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY (0x80)							
Command = BATTERY_GET (0x02)							

3.2.11.2 Battery Report Command

This command is used to report the battery level of a battery operated device.

Table 3.39: Battery Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY (0x80)							
Command = BATTERY_REPORT (0x03)							
Battery Level							

Battery Level (8 bits)

This field is used to report the percentage indicating the battery level.

CC:0080.01.03.11.001 This field MUST be in the range 0x00..0x64 or set to 0xFF.

CC:0080.01.03.11.002 Values in the range 0x00..0x64 MUST indicate the battery percentage level from 0 to 100%.

CC:0080.01.03.11.003 The value 0xFF MUST indicate a low-battery warning which MUST only be sent unsolicited. When a controller requests the battery level using a BATTERY_GET command, the actual battery level (0-100) MUST be returned. Recommendation is to send the low-battery value (0xFF) when the device battery level indicates the battery should be replaced or recharged within a few days or weeks. The low-battery warning SHOULD be sent no more than once per day. The device manufacturer knows the chemistry and discharge characteristics of the battery but the controller does not. Thus the controller cannot use a simple battery percentage threshold to decide when to inform the user the battery need to be replaced. For example, the device manufacturer may choose a battery level of 20% for the low-battery warning point for a lithium battery which has a steep discharge curve at end-of-life or 5% for alkaline cells which has a more linear discharge curve. The point where the low-battery warning is sent is at the discretion of the device manufacturer. Controllers SHOULD inform the user that the battery needs replacement or recharging each time the low-battery warning (0xFF) is received.

3.2.12 Battery Command Class, version 2

This Command Class is used to request and report the battery types, status and levels of a given device.

3.2.12.1 Compatibility Considerations

The following commands has been extended to query status and levels of a battery:

- *Battery Get Command*
- *Battery Report Command*

The following commands are added for requesting the current health of the battery.

- *Battery Health Get Command*
- *Battery Health Report Command*

3.2.12.2 Battery Get Command

This command is used to request both the level of a battery and its status.

The *Battery Report Command* MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.40: Battery Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY (0x80)							
Command = BATTERY_GET (0x02)							

3.2.12.3 Battery Report Command

This command is used to report the level and status of a battery-operated device.

Table 3.41: Battery Level Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY (0x80)							
Command = BATTERY_REPORT (0x03)							
Battery Level							
Battery Charging Status		Recharge-able	Back-up battery	Over-heating	Low fluid	Replace/recharge status bitmask	
							Discon-nected

Battery Level (8 bits)

This field is used to report the percentage indicating the battery level. This field MUST be in the range 0x00..0x64 or set to 0xFF. Values in the range 0x00..0x64 MUST indicate the battery percentage level from 0 to 100%.

The value 0xFF MUST indicate a low-battery warning and this notification MUST only be sent as an unsolicited Report to the Lifeline destinations. When a controller requests the battery level, the actual battery level (0-100) MUST be returned. The percentage level of 0% represents the lowest operational voltage of the device. For example; if the device can no longer operate properly below a battery voltage of 2.2V, the device should report 0% level when the battery voltage reaches 2.2V.

Recommendation is to send the low-battery value (0xFF) when the device battery level indicates the battery should be replaced or recharged within a few days or weeks. The low-battery warning SHOULD be sent no more than once per day. The device manufacturer knows the chemistry and discharge characteristics of the battery but the controller does not. Thus the device chooses when the user should be informed of the low battery condition. The device manufacturer may choose a battery level of 20% for the low-battery warning point for a lithium battery which has a steep discharge curve at end-of-life or 5% for alkaline cells which has a more linear discharge curve. The point where the low-battery warning is sent is at the discretion of the device manufacturer. Controllers SHOULD inform the user that the battery needs replacement each time the low-battery warning (0xFF) is received.

The Replace/Recharge field MUST have at least bit 0 set when the low-battery warning is sent.

Battery Charging Status (2 bits)

CC:0080.02.03.11.001 The field describes the charging status of a battery if the *Rechargeable* flag field is 1. The field MUST be encoded according to Table 3.42.

CC:0080.02.03.11.002 If the *Rechargeable* flag field is set to 0, the battery charging status value MUST be 0x00.

Table 3.42: Charging status notification values: Battery Command Class

Value	Description
0x00	Discharging
0x01	Charging
0x02	Maintaining

Rechargeable (1 bit)

This field indicates if the battery is rechargeable or not.

CC:0080.02.03.11.003 If the battery is rechargeable this field MUST be set to 1.

CC:0080.02.03.11.004 If the battery is not rechargeable, the field value MUST set to 0.

Back-up battery (1 bit)

This field is used to illustrate if the battery is utilized for back-up purposes of a mains powered connected device.

CC:0080.02.03.11.005 If the battery is used for back-up purposes the field MUST be set to 1.

CC:0080.02.03.11.006 If battery is the primary means of power and is not for back-up, this field MUST set to 0.

Overheating (1 bit)

This field is used to indicate if overheating is detected at the battery.

CC:0080.02.03.11.007 The value 1 MUST indicate that the battery is overheating

CC:0080.02.03.11.008 The value 0 MUST indicate that the battery is operating within the normal temperature range.

Low fluid (1 bit)

This field is used to indicate if the battery fluid is low and should be refilled.

CC:0080.02.03.11.009 The value 1 MUST indicate that the battery fluid is low.

CC:0080.02.03.11.00A The value 0 MUST indicate that the battery fluid is within the normal range and no maintenance is required.

Replace/recharge status bitmask (2 bits)

This field is used to indicate if the battery needs to be recharged or replaced.

CC:0080.02.03.11.00B Bit 0 set to 1 MUST indicate that the battery MUST be replaced/recharged soon.

CC:0080.02.03.11.00C Bit 1 set to 1 MUST indicate that the battery MUST be replaced/recharged now.

CC:0080.02.03.11.00D If bit 1 is set to 1, bit 0 MUST also be set to 1.

Disconnected (1 bit)

This field is used to indicate if the battery is currently disconnected or removed from the node.

CC:0080.02.03.11.00E The value 1 MUST indicate that the battery is disconnected and the node is running on an alternative power source.

CC:0080.02.03.11.00F The value 0 MUST indicate that the battery correctly connected to the node.

CC:0080.02.03.11.010 If this field is set to 1, the *Battery Level* field MUST be set to 0x00.

3.2.12.4 Battery Health Get Command

This command is used to query the health of the battery, particularly the battery temperature and maximum capacity.

CC:0080.02.04.11.001 The *Battery Health Report Command* MUST be returned in response to this command.

CC:0080.02.04.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.43: Battery Health Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY (0x80)							
Command = BATTERY_HEALTH_GET (0x04)							

3.2.12.5 Battery Health Report Command

This command is used to report the maximum capacity of the battery as well as the temperature of the battery.

Table 3.44: Battery Health Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY (0x80)							
Command = BATTERY_HEALTH_REPORT (0x05)							
Maximum Capacity							
Precision			Scale		Size		
Battery Temperature 1							
...							
Battery Temperature N							

Maximum Capacity (8 bits)

This field is used to report the percentage indicating the maximum capacity of the battery.

CC:0080.02.05.11.001 This field MUST be in the range 0x00..0x64 or set to 0xFF. If the maximum capacity of the battery is unknown the value MUST set to 0xFF.

CC:0080.02.05.11.002 Values in the range 0x00..0x64 MUST indicate the maximum capacity of the battery in the percentage level from 0 to 100%.

For example, if the initial battery total capacity is 1500mAh, 100% represents this value. If subsequently the node detects that the total battery capacity when fully loaded is down to 1350, it must report 90%.

Precision (3 bits)

This field is used to indicate how many decimal places are included in the *Battery Temperature* field. For example, the *Battery Temperature* field set to 1025 with the Precision field set to 2 MUST be interpreted as 10.25.

Scale (2 bits)

CC:0080.02.05.11.003 The scale field indicates the scale used for the battery temperature value. This field MUST be encoded according to Table 10

Table 3.45: Battery Health Report: Scale field encoding

Value	Description
0x00	Celsius
0x01..0x03	Reserved

Size (3 bits)

CC:0080.02.05.11.004 The size field indicates the number of bytes used for the battery temperature value. This field MUST be set to 0, 1, 2 or 4.

CC:0080.02.05.11.005 The value 0 MUST indicate that the battery temperature is unknown. In this case, the *Scale* and
CC:0080.02.05.11.006 *Precision* fields MUST be set to 0 and the *Battery Temperature* field MUST be omitted.

CC:0080.02.05.11.007 Values 1, 2 and 4 MUST indicate the length in bytes of the *Battery Temperature* field.

Battery Temperature (N bytes)

This field advertises the temperature of the battery.

CC:0080.02.05.11.008 The length of this field in bytes MUST be according to the Size field.

CC:0080.02.05.11.009 The first byte MUST be the most significant byte.

CC:0080.02.05.11.010 This field MUST be encoded using signed representation and comply with Signed field encoding representation (Table 2.12).

3.2.13 Battery Command Class, version 3

This Command Class is used to request and report the battery types, status and levels of a given device. The Battery Command Class, version 3 is backwards compatible with the Battery Command
CC:0080.03.00.21.001 Class, version 2. Fields and commands not described in this version MUST remain unchanged from version 2.

The Battery Report Command has been extended to advertise if the device battery is stop charging due to low temperature.

3.2.13.1 Battery Report Command

This command is used to report the level and status of battery-operated device

Table 3.46: Battery Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY (0x80)							
Command = BATTERY_REPORT (0x03)							
Battery Level							
Battery Charging Status		Rechargeable	Back-up Battery	Overheating	Low Fluid	Replace/recharge status bitmask	
Reserved						Low Temperature Status	Disconnected

All fields not described below remain unchanged from version 2.

Low Temperature Status (1 bit)

This status is used to advertise if the battery of a device has stopped charging due to low temperature.

CC:0080.03.03.11.001 The value 1 MUST indicate that the battery is not charging due to low temperature.

The value 0 MUST indicate that the battery is operational.

CC:0080.03.03.11.002 If the battery is not rechargeable, the field value MUST set to 0.

3.2.14 Device Reset Locally Command Class, version 1

The Device Reset Locally Command Class is used to notify central controllers that a Z-Wave device is resetting its network specific parameters.

CC:005A.01.00.12.001 Any node SHOULD support this Command Class if it can be reset to factory default.

3.2.14.1 Device Reset Locally Notification Command

The Device Reset Locally Notification Command is used to advertise that the device will be reset to default.

CC:005A.01.01.11.001 The reset operation MUST reset protocol data (HomeID, NodeID, etc.). For Z-Wave Plus nodes, additional requirements upon node reset are provided in Section 8.

CC:005A.01.01.11.002 In case a Lifeline destination is configured in Association Group #1, the device MUST send a Device Reset Locally Notification Command to the Lifeline destinations.

CC:005A.01.01.11.004 The Device Reset Locally Notification Command MUST be issued by the Root Device. If Multi Channel encapsulation is used, the source End Point MUST be set to 0.

Table 3.47: Device Reset Locally Notification Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DEVICE_RESET_LOCALLY (0x5A)							
Command = DEVICE_RESET_LOCALLY_NOTIFICATION (0x01)							

3.2.15 Firmware Update Meta Data Command Class, version 1 [DEPRECATED]

Warning: THIS VERSION HAS BEEN DEPRECATED

CC:007A.01.00.13.001 CC:007A.01.00.12.001 A device MAY implement support for this version of the command class, but it is RECOMMENDED that new implementations implement the Firmware Update Meta Data Command Class, version 3.

If implementing support for this version of the command class, it is RECOMMENDED that support for Firmware Update Meta Data Command Class, version 3 is also implemented.

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to a Z-Wave device.

CC:007A.01.00.11.001 A device which supports the Firmware Update CC MUST support the Version CC and the Manufacturer Specific CC to enable other devices to select the correct firmware for a specific target. The Version Command Class may be used to verify that the intended firmware version is installed.

CC:007A.01.00.11.002 Devices implementing the Firmware Update CC MUST support a data rate of 40kbit/s or more.

CC:007A.01.00.12.002 CC:007A.01.00.12.003 A checksum SHOULD be appended to the firmware image to ensure the integrity of the image. A receiving device SHOULD verify the integrity of the received firmware image by matching the received firmware image checksum and the firmware image checksum that is indicated by the Firmware Meta Data Report when the firmware update is initiated.

CC:007A.01.00.12.004 It is RECOMMENDED that firmware update is enabled by out-of-band authentication (e.g. physical activation of a pushbutton). If out-of-band authentication is implemented, the device to receive a firmware update must be armed for firmware update before the Firmware Update Meta Data Request Get Command is issued.

3.2.15.1 Firmware Meta Data Get Command

The Firmware Meta Data Get Command is used to request information on the current firmware in the device.

CC:007A.01.01.11.001 The Firmware Meta Data Report Command MUST be returned in response to this command.

CC:007A.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.01.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.48: Firmware Meta Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_GET (0x01)							

3.2.15.2 Firmware Meta Data Report Command

The Firmware Meta Data Report Command is used to advertise the status of the current firmware in the device.

Table 3.49: Firmware Meta Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_REPORT (0x02)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device.

Manufacturer identifiers can be found in [32]

CC:007A.01.02.11.001 The first byte MUST be the most significant byte.

Firmware ID (16 bits)

CC:007A.01.02.12.001 A manufacturer SHOULD assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. When combined with the Manufacturer ID, the Firmware ID is a unique identification of a firmware image that is guaranteed to work with a particular product among all Z-Wave enabled products in the world.

If the product has no Firmware ID, the value 0x0000 MUST be returned.

CC:007A.01.02.11.002 The first byte MUST be the most significant byte.

A user may request Manufacturer Specific information to get more detailed information on the product.

CC:007A.01.02.12.002 A controlling device SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

Checksum (16 bits)

The checksum field is used to report a checksum value of the firmware image currently running in the
CC:007A.01.02.13.001 Z-Wave chip. As an alternative, a value of zero MAY be returned.

CC:007A.01.02.11.003 The first byte MUST be the most significant byte.

CC:007A.01.02.12.003 It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomial using initialization value equal to 0x1D0F and 0x1021 (normal representation).

3.2.15.3 Firmware Update Meta Data Request Get Command

The Firmware Update Meta Data Request Get Command is used to request that a firmware update is initiated.

CC:007A.01.03.11.001 The Firmware Update Meta Data Request Report Command MUST be returned in response to this command.

CC:007A.01.03.11.002 The firmware update MUST NOT be initiated if the Manufacturer ID and the Firmware ID do not match the actual firmware image values.

CC:007A.01.03.11.003 The firmware update MUST be aborted if the checksum does not match the calculated checksum after the firmware image has been transferred.

CC:007A.01.03.12.001 It is RECOMMENDED that firmware update is initiated by out-of-band authentication (e.g. physical activation of a pushbutton).

CC:007A.01.03.11.004 This command MUST NOT be issued via multicast addressing.

CC:007A.01.03.11.005 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.50: Firmware Update Meta Data Request Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET (0x03)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device.

CC:007A.01.03.11.006 Manufacturer identifiers can be found in [32]. The first byte MUST be the most significant byte.

Firmware ID (16 bits)

CC:007A.01.03.12.002 A manufacturer SHOULD assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. When combined with the Manufacturer ID, the Firmware ID is a unique identification of a firmware image that is guaranteed to work with a particular product among all Z-Wave enabled products in the world.

CC:007A.01.03.11.007 A receiving application MUST check that the Manufacturer ID and Firmware ID fields match the Manufacturer ID and Firmware ID values of the current firmware. While it is NOT RECOMMENDED, one MAY design a the product that has no Firmware ID.

CC:007A.01.03.11.008 A sending node MUST specify the Firmware ID value 0x0000 to indicate an absent Firmware ID.

CC:007A.01.03.11.009 A receiving node MUST interpret the value 0x0000 as “No Firmware ID available”.

CC:007A.01.03.11.00A The first byte MUST be the most significant byte.

Checksum (16 bits)

CC:007A.01.03.11.00B The checksum field MUST carry the checksum of the firmware image about to be transferred.

CC:007A.01.03.11.00C A receiving node MUST match this value and the checksum calculated from the received firmware image.

CC:007A.01.03.12.004 It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). Refer to *CRC-CCITT Source Code*.

3.2.15.4 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

Table 3.51: Firmware Update Meta Data Request Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT (0x04)							
Status							

Status (8 bits)

CC:007A.01.04.11.001 This field MUST comply with Table 3.81.

3.2.15.5 Firmware Update Meta Data Get Command

The Firmware Update Meta Data Get Command is used to request one or more Firmware Update Meta Data Report Commands.

CC:007A.01.05.11.001 The Firmware Update Meta Data Report Command MUST be returned in response to this command.

CC:007A.01.05.13.001 CC:007A.01.05.13.002 The transmission of the next Firmware Update Meta Data Get Command MAY be delayed if time is required to store the most recent firmware fragment in temporary non-volatile memory. A node MAY request multiple Firmware Update Meta Data Report Commands to improve throughput.

CC:007A.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Two exceptions may apply: A noise burst may interfere the transmission or the data source may stop returning Firmware Update Meta Data Report Commands.

CC:007A.01.05.12.001 To accommodate for bursts of RF noise, a device receiving data SHOULD repeatedly retransmit the same Firmware Update Meta Data Get Command every 10 seconds in case no Firmware Update Meta Data Report Commands are received.

CC:007A.01.05.12.002 A device receiving data SHOULD stop retransmitting Firmware Update Meta Data Get commands 2 minutes after the last successful reception of a Firmware Update Meta Data Report Command.

Table 3.52: Firmware Update Meta Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_GET (0x05)							
Number of Reports							
Res	Report number 1						
Report number 2							

Res

CC:007A.01.05.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Reports (8 bits)

Number of Firmware Update Meta Data Report Commands to be received in response to this Firmware Update Meta Data Get Command.

Report number 1 .. 2 (15 bits)

The Report number field indicates the Firmware Update Meta Data Report Command to be requested.
CC:007A.01.05.11.005 The report number values MUST be a sequence starting from 1. The first byte (Report number 1) is the most significant byte.

3.2.15.6 Firmware Update Meta Data Report Command

The Firmware Update Meta Data Report Command is used to transfer a firmware image fragment.

CC:007A.01.06.11.001 If sending more than a single Firmware Update Meta Data Report Command at a time, a node MUST apply a delay between each transmitted command. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 1 frame back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

CC:007A.01.06.12.001 If needed, a controlling node SHOULD abort an ongoing transfer by responding to a Firmware Update Meta Data Get Command with a Firmware Update Meta Data Report Command with the Last bit enabled and the Data fields intentionally corrupted.

This will invalidate the calculated firmware checksum, which should eventually cause the receiving device to return a Firmware Update Meta Data Status Report Command with the status code <checksum error>.

Table 3.53: Firmware Update Meta Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REPORT (0x06)							
Last	Report number 1						
Report number 2							
Data 1							
...							
Data N							

Last (1 bit)

The Last flag indicates if the requested Firmware Update Meta Data Report Command carries the last firmware image fragment.

CC:007A.01.06.11.002 The flag MUST be set to '1' if this is the last fragment. Otherwise the flag MUST be '0'.

Report number (15 bits)

The Report number field indicates the sequence number of the contained firmware fragment. The first
CC:007A.01.06.11.003 firmware fragment MUST be identified by the Report number value 1. The sequence number of each following firmware fragment MUST be incremented.

Report number 1 is the most significant byte.

The report number may be used to calculate the offset of the data by the formula:

Offset = (Report number – 1) x Number of Data fields (N)

CC:007A.01.06.11.004 Except for the last frame, each Report MUST carry the same number of Data bytes as the first
CC:007A.01.06.13.001 fragment. The last frame MAY carry a shorter Data field.

Data (N bytes)

The Data field is used to carry one firmware image fragment. Except for the last frame, each Firmware
CC:007A.01.06.11.005 Update Meta Data Report Command MUST carry the same number of Data bytes as the first frag-
CC:007A.01.06.13.002 ment. The last frame MAY carry a shorter Data field.

CC:007A.01.06.11.006 A sending device MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

CC:007A.01.06.11.007 A receiving device MUST determine the number of Data bytes from the length of the first received frame.

3.2.15.7 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

CC:007A.01.07.11.001 The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

CC:007A.01.07.12.001 A supporting node SHOULD reboot and apply the new firmware image before issuing this command.

CC:007A.01.07.11.002 A node MUST NOT issue Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

Table 3.54: Firmware Update Meta Data Status Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT (0x07)							
Status							

Status (8 bits)

CC:007A.01.07.11.003 This field MUST comply with Table 3.71.

3.2.15.8 Examples

Figure 3.3 shows a controlling node requesting the Manufacturer ID and Firmware ID of another node by issuing the Firmware Meta Data Get Command.

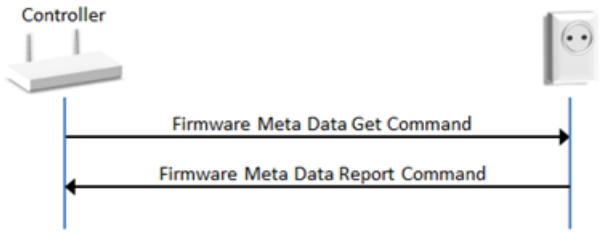


Figure 3.3: Requesting Manufacturer ID and Firmware ID of a Node

Figure 3.4 outlines the actual firmware update message flow. Prior to the firmware update, the supporting node may receive an out-of-band authentication (e.g. physical activation of a pushbutton). The controller sends a Firmware Meta Data Request Get Command to initiate the downloading a new firmware image.

The supporting node returns a Firmware Meta Data Request Report Command report to confirm the update request and the supporting node begins pulling firmware image fragments from the controller. Finally a Firmware Update Meta Data Status Report is returned to the controller to indicate the success (or failure) of the update process.

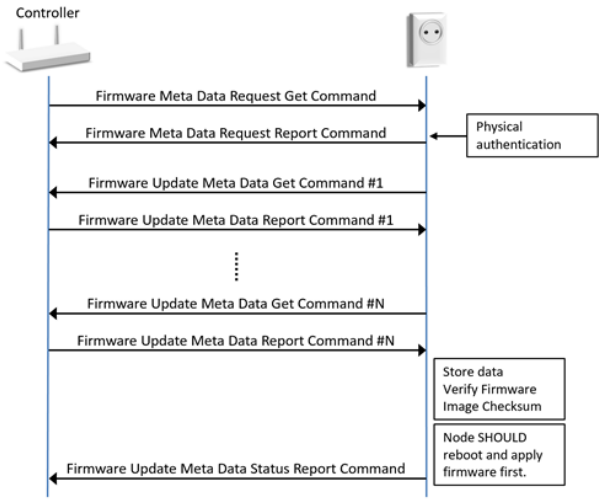


Figure 3.4: Transferring a Firmware Image to a Device

3.2.16 Firmware Update Meta Data Command Class, version 2

Firmware Update Meta Data Command Class, version 2 updates the Firmware Update Meta Data Report Command.

While support for version 1 has been deprecated, a controlling device implementing Firmware Update
CC:007A.02.00.11.001 Meta Data Command Class, version 2 MUST also be able to control Firmware Update Meta Data Command Class, version 1 based devices.

3.2.16.1 Compatibility considerations

CC:007A.02.00.21.001 A node supporting Firmware Update Meta Data Command Class, version 2 MUST also support Firmware Update Meta Data Command Class, version 1.

All commands and fields not described in this version remain unchanged from version 1.

3.2.16.2 Interoperability considerations

3.2.16.2.1 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report v2 command will consider the 16-bit Checksum field to be the last two bytes of the Data field.

Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report v1 command will consider the last two bytes of the Data field to be the 16-bit Checksum field.

In either case, the transfer will fail.

CC:007A.02.00.31.001 Therefore, a controlling device implementing the Firmware Update Meta Data Command Class, version 2 MUST do the following when initiating the transfer of a firmware image:

1. Request the version of the Firmware Update Meta Data Command Class from the target device
2. Use the version of the Firmware Update Meta Data Report implemented by the target device
3. Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

CC:007A.02.00.32.001 The controller SHOULD accept any binary image format and transfer that without any modifications. The image may be encrypted or carry checksums.

3.2.16.2.2 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory. For more details about the checksum calculation, refer to *CRC-CCITT Source Code*.

3.2.16.3 Firmware Update Meta Data Report Command

The Firmware Update Meta Data Report command is used to transfer a firmware image fragment.

CC:007A.02.06.11.001 If sending more than a single Firmware Update Meta Data Report Command at a time, a node MUST apply a delay between each transmitted command. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 1 frame back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

CC:007A.02.06.12.001 If needed, a controlling node SHOULD abort an ongoing transfer by responding to a Firmware Update Meta Data Get Command with a Firmware Update Meta Data Report Command with the Last bit enabled and the Data fields intentionally corrupted while the Checksum field of the Firmware Update Meta Data Report Command is valid, i.e. calculated over the corrupted Data fields.

This will invalidate the calculated firmware checksum, which should eventually cause the receiving device to return a Firmware Update Meta Data Status Report Command with the status code <checksum error>.

Table 3.55: Firmware Update Meta Data Report Command version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REPORT (0x06)							
Last	Report number 1						
Report number 2							
Data 1							
...							
Data N							
Checksum 1							
Checksum 2							

All fields not described below remain unchanged from version 1.

Checksum (16 bits)

CC:007A.02.06.11.002 The checksum field MUST be used to ensure the consistency of the entire command; including the command class and command identifiers.

CC:007A.02.06.12.002 It is RECOMMENDED to use a checksum algorithm that implements the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

The Checksum field is known to cause compatibility issues with version 1 devices. Refer to Section 3.2.16.1.

3.2.17 Firmware Update Meta Data Command Class, version 3

The Firmware Update Meta Data Command Class, version 3 allows a Z-Wave enabled device to receive one out of several firmware images.

As an example, one may construct a product comprising a Z-Wave chip and a secondary host processor that maintains a security certificate. With the capability to handle multiple firmware images enables the Z-Wave chip, the host processor and the security certificate may all be updated via individual firmware IDs.

3.2.17.1 Compatibility considerations

CC:007A.03.00.21.001 A device implementing Firmware Update Meta Data Command Class, version 3 MUST also implement Firmware Update Meta Data Command Class, version 2.

CC:007A.03.00.21.002 While support for version 1 has been deprecated, a controlling device implementing Firmware Update Meta Data Command Class, version 2 MUST also be able to control Firmware Update Meta Data Command Class, version 1 based devices.

All commands and fields not described in this version remain unchanged from version 2.

Note that the minimum granted firmware size that can be transferred using Firmware Update Meta Data Report command is around 900 kB.

3.2.17.1.1 New fields and values

The Firmware Update Meta Data Command Class, Version 3, introduces WaitTime parameter and a new Status code 0xFE for the Firmware Update Meta Data Status Report Command.

The status code 0xFE is intended for confirming the successful transfer of images which do not necessitate a restart, e.g. security certificates. If this code was returned to a controller implementing a previous version, the controller would report that an error had occurred. Therefore, a supporting
CC:007A.03.00.21.003 node MUST NOT return this code after having updated the firmware image for the Z-Wave chip image, i.e. the Firmware ID 0 target.

3.2.17.2 Interoperability considerations

CC:007A.03.00.32.001 Unintended modification of a firmware image SHOULD be prevented. It is RECOMMENDED that firmware update is enabled by out-of-band authentication (e.g. physical activation of a pushbutton).

CC:007A.03.00.31.001 A checksum MUST be appended to the Firmware 0 image to ensure the integrity of the image. A
CC:007A.03.00.32.002 receiving device SHOULD verify the integrity of the received Firmware 0 image by matching the received Firmware 0 image checksum and the Firmware 0 image checksum that is indicated by the Firmware Meta Data Report when the firmware update is initiated.

CC:007A.03.00.32.003 Any image transferred via the Firmware Update Meta Data Command Class SHOULD include a fingerprint value for validation of the integrity of the entire image after the transfer. The image and
CC:007A.03.00.31.002 the fingerprint value MUST be packed in one entity which can be stored in one file and transferred as one entity.

CC:007A.03.00.31.003 A manufacturer MUST assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. The combination of Manufacturer ID and Firmware ID provides unique identification of a firmware image that is guaranteed to work with a particular component of a particular product.

3.2.17.2.1 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command. Version 3 of the Firmware Update Meta Data Report command uses the v2 format.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report v2 command will consider the 16-bit Checksum field to be the last two bytes of the Data field.

Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report v1 command will consider the last two bytes of the Data field to be the 16-bit Checksum field.

In either case, the transfer will fail.

CC:007A.03.00.31.004 Therefore, a node controlling the Firmware Update Meta Data Command Class, version 3 MUST do the following when initiating the transfer of a firmware image:

1. Request the version of the Firmware Update Meta Data Command Class from the target device
2. Use the version of the Firmware Update Meta Data Report implemented by the target device
3. Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report

3.2.17.2.2 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory.

For more details about the checksum calculation, refer to *CRC-CCITT Source Code*.

3.2.17.3 Firmware Meta Data Report Command

This command is used to advertise the status of the current firmware in the device.

Table 3.56: Firmware Meta Data Report Command version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_REPORT (0x02)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware 0 ID 1							
Firmware 0 ID 2							
Firmware 0 Checksum 1							
Firmware 0 Checksum 2							
Firmware Upgradable							
Number of Firmware Targets							
Max Fragment Size 1							
Max Fragment Size 2							
Firmware 1 ID 1							
Firmware 1 ID 2							
...							
Firmware N ID 1							
Firmware N ID 2							

Fields not described below remain unchanged from version 2

Firmware 0 ID (16 bits)

The Firmware 0 ID field is dedicated to target 0, i.e. the Z-Wave chip.

CC:007A.03.02.11.001 A manufacturer MUST assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. The combination of Manufacturer ID and Firmware ID provides unique identification of a firmware image that is guaranteed to work with a particular component of a particular product.

CC:007A.03.02.11.002 The first byte MUST be the most significant byte.

A user may request Manufacturer Specific information to get more detailed information on the product.

CC:007A.03.02.12.001 A controlling node SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

Firmware 0 Checksum (16 bits)

CC:007A.03.02.11.003 The checksum field is used to ensure consistency of the Firmware 0 image currently in the device.

Supporting nodes MAY set this field to 0x00. In this case the field is unused.

Values in the range 0x0001..0xFFFF indicate that the supporting node advertises the checksum of its firmware image.

CC:007A.03.02.11.004 The first byte MUST be the most significant byte.

CC:007A.03.02.12.002 It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomial using initialization value equal to 0x1D0F and 0x1021 (normal representation).

Firmware Upgradable (8 bits)

This field defines whether the Z-Wave chip is firmware upgradable.

CC:007A.03.02.11.005 The value 0x00 MUST indicate that the firmware 0 image is not upgradable.

The value 0xFF MUST indicate that the firmware 0 image is upgradable.

Number of Firmware Targets (8 bits)

CC:007A.03.02.11.006 The Number of Firmware Targets field MUST report the number of firmware IDs following this field.

CC:007A.03.02.11.007 The Firmware 0 ID field is not included. The field MUST be zero if the device only implements a Firmware 0 target, i.e. the Z-Wave chip.

Max Fragment Size (16 bits)

CC:007A.03.02.11.008 CC:007A.03.02.13.001 The Max Fragment Size field MUST report the maximum number of Data bytes that a device is able to receive at a time. A sending node MAY send shorter fragments. The fragment size actually used is indicated in the Firmware Update Meta Data Request Get Command and confirmed in the Firmware Update Meta Data Request Report Command.

CC:007A.03.02.11.009 The Max Fragment Size may be longer than the supported frame length of Z-Wave if the image is to be transferred over e.g. IP. If the image is to be transferred over Z-Wave, the sending node MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

Firmware n ID (16 bits)

CC:007A.03.02.11.00A The Firmware 1 ID field indicates the Firmware ID that MUST be used for target 1. The Firmware 2 ID field represents target 2. And so on.

CC:007A.03.02.11.00B The first byte MUST be the most significant byte.

CC:007A.03.02.12.003 A controlling device SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

3.2.17.4 Firmware Update Meta Data Request Get Command

The Firmware Update Meta Data Request Get Command is used to request that a firmware update is initiated by the node receiving this command.

CC:007A.03.03.11.001 The Firmware Update Meta Data Request Report Command MUST be returned in response to this command.

CC:007A.03.03.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.03.03.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

CC:007A.03.03.11.004 The firmware update MUST NOT be initiated if the Manufacturer ID and the Firmware ID do not match the actual firmware image values for the specified Firmware Target.

CC:007A.03.03.11.005 The firmware update MUST be aborted if the checksum does not match the calculated checksum after the firmware image has been transferred.

CC:007A.03.03.12.001 It is RECOMMENDED that firmware update is enabled by out-of-band authentication (e.g. physical activation of a pushbutton) prior to the transmission of this command.

CC:007A.03.03.12.002 Any image transferred via the Firmware Update Meta Data Command Class SHOULD include a fingerprint value to enable the validation of the integrity of the entire image after the transfer. The CC:007A.03.03.11.006 image and the fingerprint value MUST be packed in one entity which can be stored in one file and transferred as one entity.

Table 3.57: Firmware Meta Data Request Get Command version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET (0x03)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							

All fields not described below remain unchanged from version 2.

Firmware ID (16 bits)

CC:007A.03.03.11.007 The Firmware ID MUST match the specified target of the actual device.

CC:007A.03.03.11.008 The firmware update MUST NOT be initiated if the Firmware ID does not match the specified target of the actual device.

CC:007A.03.03.11.009 The first byte MUST be the most significant byte.

Firmware Target (8 bits)

CC:007A.03.03.11.00A This field MUST be used to identify the firmware image to be updated. The firmware images MUST be identified as follows:

Table 3.58: Firmware Target (8 bits)

Firmware Target	Description
0x00	Firmware image targeted for the Z-Wave chip.
0x01	Firmware image intended for target 1 defined by the manufacturer.
...	...
0xFF	Firmware image intended for target 255 defined by the manufacturer.

Fragment Size (16 bits)

CC:007A.03.03.11.00B The Fragment Size field MUST report the fragment size that is to be used for firmware fragments. A receiving device MUST use this fragment size for the firmware update. The fragment size is not exchanged during the actual firmware update.

CC:007A.03.03.11.00C The Fragment Size MUST NOT exceed the Max Fragment Size value of the Firmware Meta Data Report.

CC:007A.03.03.11.00D A version 1 Firmware Update Meta Data Request Get Command does not carry a Fragment Size field. A receiving node MUST determine the number of Data bytes from the length of the first received frame, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

3.2.17.5 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

Table 3.59: Firmware Update Meta Data Request Report Command version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT (0x04)							
Status							

Status (8 bits)

CC:007A.03.04.11.001 This field MUST comply with Table 3.81.

3.2.17.6 Firmware Update Meta Data Get Command

The Firmware Update Meta Data Get Command is used to request one or more Firmware Update Meta Data Report Commands.

CC:007A.03.05.11.001 One or more Firmware Update Meta Data Report Commands MUST be returned in response to this command.

CC:007A.03.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.03.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

CC:007A.03.05.11.004 A node MUST verify the Manufacturer ID and Firmware ID fields received in a Firmware Update Meta Data Request Get Command before sending any Firmware Update Meta Data Get Commands.

CC:007A.03.05.11.005 A controlling node receiving a Firmware Update Meta Data Get Command MUST return fragments of the firmware image that was previously identified by the Manufacturer ID and Firmware ID fields received in a Firmware Update Meta Data Request Get Command.

CC:007A.03.05.13.001 The transmission of the next Firmware Update Meta Data Get Command MAY be delayed if time is
CC:007A.03.05.13.002 required to store the most recent firmware fragment in temporary non-volatile memory. A node MAY request multiple Firmware Update Meta Data Report Commands to improve throughput.

CC:007A.03.05.13.003 RAM may be a limited resource. A node MAY adjust the size of incoming Firmware Update Meta Data Report Commands to the available first level RAM receive buffer by reporting the desired Max Fragment Size when returning a Firmware Meta Data Report Command in response to a Firmware Update Meta Data Get Command.

Two exceptions may apply: A noise burst may interfere the transmission or the data source may stop returning Firmware Update Meta Data Report Commands.

CC:007A.03.05.12.001 To accommodate for bursts of RF noise, a device receiving data SHOULD repeatedly retransmit the same Firmware Update Meta Data Get Command every 10 seconds in case no Firmware Update Meta Data Report Commands are received.

CC:007A.03.05.12.002 A device receiving data SHOULD stop retransmitting Firmware Update Meta Data Get commands 2 minutes after the last successful reception of a Firmware Update Meta Data Report Command.

Table 3.60: Firmware Update Meta Data Get Command version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_GET (0x05)							
Number of Reports							
Res	Report Number 1						
Report Number 2							

Res

CC:007A.03.05.11.006 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Reports (8 bits)

Number of Firmware Update Meta Data Report Commands to be received in response to a single Firmware Update Meta Data Get Command.

CC:007A.03.05.11.007 The requesting node MUST keep track of missing Firmware Update Meta Data Report Commands.

Report Number (15 bits)

The Report number field indicates the sequence number of the requested firmware fragment. The first
CC:007A.03.05.11.008 firmware fragment MUST be identified by the Report Number value 1.

CC:007A.03.05.11.009 Report number 1 MUST be the most significant byte.

CC:007A.03.05.11.00A

If the Number of Reports field is larger than 1, the Report Number MUST identify the first firmware fragment in the requested sequence of firmware fragments.

3.2.17.7 Firmware Update Meta Data Report Command

The Firmware Update Meta Data Report command is used to transfer a firmware image fragment.

CC:007A.03.06.11.001 If sending more than a single Firmware Update Meta Data Report Command at a time, a node MUST apply a delay between each transmitted command. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 1 frame back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

After receiving all Firmware Update Meta Data Report Commands required for a complete firmware image, a receiving node MUST store the firmware image in the Firmware Target specified in a previously received Firmware Update Meta Data Request Get Command.

CC:007A.03.06.12.001 If needed, a controlling node SHOULD abort an ongoing transfer by responding to a Firmware Update Meta Data Get Command with a Firmware Update Meta Data Report Command with the Last bit enabled and the Data fields intentionally corrupted while the Checksum field of the Firmware Update Meta Data Report Command is valid, i.e. calculated over the corrupted Data fields.

This will invalidate the calculated firmware checksum, which should eventually cause the receiving device to return a Firmware Update Meta Data Status Report Command with the status code <checksum error>.

Table 3.61: Firmware Update Meta Report Command version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REPORT (0x06)							
Last	Report Number 1						
Report Number 2							
Data 1							
...							
Data N							
Checksum 1							
Checksum 2							

All fields not described below remain unchanged from version 2.

Last (1 bit)

This field indicates if the Firmware Update Meta Data Report Command is the last one.

CC:007A.03.06.11.002 The value 1 MUST indicate that this is the last report. Otherwise the field MUST be set to 0.

CC:007A.03.06.11.003 On the reception of the last image fragment, the receiving node MUST verify that the transferred image does match the indicated Manufacturer ID, Firmware ID and Firmware Target values.

Checksum (16 bits)

CC:007A.03.06.11.004 The checksum field MUST be used to ensure the consistency of the entire command; including the command class and command identifiers.

CC:007A.03.06.12.002 It is RECOMMENDED to use a checksum algorithm that implements the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

CC:007A.03.06.11.005 The checksum algorithm used for the Firmware Update Meta Data Command Class, version 3 MUST be the same as the algorithm used for the Firmware Update Meta Data Command Class, version 2 in the actual product.

The Checksum field is known to cause compatibility issues with version 1 devices. Refer to Section 3.2.16.1.

3.2.17.8 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

CC:007A.03.07.11.001 The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

CC:007A.03.07.11.002 A node MUST NOT issue the Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

Table 3.62: Firmware Update Meta Data Status Report Command version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT (0x07)							
Status							
WaitTime MSB							
WaitTime LSB							

Status (8 bits)

CC:007A.03.07.11.003 This field MUST comply with Table 3.71.

CC:007A.03.07.13.001 The status code 0xFE MAY be used for confirming the successful transfer of images which do not necessitate a restart, e.g. security certificates.

CC:007A.03.07.11.004 The status code 0xFE MUST NOT be advertised after the transfer of an image for the Firmware ID 0 target (the "Z-Wave chip" image).

Controlling nodes implementing earlier versions of the Firmware Update Meta Data CC do not support status codes defined for newer versions. Therefore, a node returning the Firmware Update Meta Data
CC:007A.03.07.11.005 Status Report Command MUST comply with the version implemented on the controlling device (device sending the image). The device returning the Firmware Update Meta Data Status Report can identify the version of the controlling device based on the Firmware Update Meta Data Request Get. This may be done as follows:

- If the Firmware Update Meta Data Request Get does not include Firmware Target and Fragment Size (8 bytes), the controller is version 1 or 2
- If the Firmware Update Meta Data Request Get includes Firmware Target and Fragment Size but not Activation (11 bytes), the controller is version 3

WaitTime (16 bits)

CC:007A.03.07.11.006 The WaitTime field MUST report the time that is needed before the receiving node again becomes available for communication after the transfer of an image. The unit is the second.

CC:007A.03.07.11.007 The value 0 (zero) MUST indicate that the node is ready (it SHOULD have rebooted if needed and applied the Firmware Image).

The value 0xFFFF is reserved and MUST NOT be returned.

CC:007A.03.07.12.001 A controlling node receiving this command SHOULD wait for the number of seconds specified in this field before trying to resume communication. When resuming communication, the controlling
CC:007A.03.07.12.002 application SHOULD issue a NOP command and wait for acknowledgement.

CC:007A.03.07.13.002 The controlling application MAY attempt resuming communication repeatedly. In that case, the NOP
CC:007A.03.07.11.008 interval SHOULD be five seconds and MUST be at least one second.

3.2.18 Firmware Update Meta Data Command Class, version 4

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to a Z-Wave device.

3.2.18.1 Compatibility considerations

All commands and fields not described in this version remain unchanged from version 3.

CC:007A.04.00.21.001 A device implementing Firmware Update Meta Data Command Class, version 4 MUST also implement Firmware Update Meta Data Command Class, versions 3.

CC:007A.04.00.21.002 While support for version 1 has been deprecated, a controlling device implementing Firmware Update Meta Data Command Class, version 2 MUST also be able to control Firmware Update Meta Data Command Class, version 1 based devices.

3.2.18.1.1 New commands, fields and values

The Firmware Update Meta Data Command Class version 4 supports delaying and scheduling a firmware update after downloading the image.

Version 4 updates the Firmware Update Meta Data Request Get Command and introduces the following commands:

- New Firmware Update Activation Set Command
- New Firmware Update Activation Status Report

After downloading an image, a device may use the Firmware Update Meta Data Status Report Command to indicate to the controlling device that the device is waiting for an activation command.

3.2.18.2 Interoperability considerations

3.2.18.2.1 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command. Version 4 of the Firmware Update Meta Data Report command uses the v2 format.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report v2 command will consider the 16-bit Checksum field to be the last two bytes of the Data field.

Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report v1 command will consider the last two bytes of the Data field to be the 16-bit Checksum field.

In either case, the transfer will fail.

Therefore, a controlling device implementing the Firmware Update Meta Data Command Class, ver-
CC:007A.04.00.31.001 sion 2 MUST do the following when initiating the transfer of a firmware image:

1. Request the version of the Firmware Update Meta Data Command Class from the target device
2. Use the version of the Firmware Update Meta Data Report implemented by the target device
3. Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the

use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report

3.2.18.2.2 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory.

For more details about the checksum calculation, refer to *CRC-CCITT Source Code*.

3.2.18.3 Firmware Update Meta Data Request Get Command

Table 3.63: Firmware Update Meta Data Request Get Command version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET (0x03)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Reserved							Activation

All fields not described below remain unchanged from version 3.

Activation (1 bit)

The Activation field is used to advertise if the receiving node may delay the actual firmware update.

CC:007A.04.03.11.001 The field MUST be interpreted as:

CC:007A.04.03.13.001 CC:007A.04.03.11.002
- '1': The receiving device MAY delay the actual firmware update.

  If the receiving node delays the firmware update, the delay MUST be advertised via the Status code 0xFD in the Firmware Update Meta data Status Report Command.

CC:007A.04.03.11.003
- '0': The receiving device MUST NOT delay the firmware update.

3.2.18.4 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

CC:007A.04.07.11.001 The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

CC:007A.04.07.11.002 A node MUST NOT issue the Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

Table 3.64: Firmware Update Meta Data Status Report Command version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT (0x07)							
Status							
WaitTime MSB							
WaitTime LSB							

All fields not described below remain unchanged from version 3.

Status (8 bits)

CC:007A.04.07.11.003 This field MUST comply with Table 3.71.

CC:007A.04.07.13.001 The status code 0xFD MAY be used by a version 4 device for confirming that an image has been successfully transferred but that the actual firmware update will not be performed until a Firmware Update Activation Set Command is received.

CC:007A.04.07.13.002 The Firmware Update Activation Set Command MAY be delayed for any period of time. The delay may be controlled via the Schedule Command Class.

CC:007A.04.07.13.003 The status code 0xFE MAY be used for confirming the successful transfer of an image which does not necessitate a restart, e.g. security a certificate.

CC:007A.04.07.11.004 The status code 0xFE MUST NOT be advertised after the transfer of an image for the Firmware ID 0 target (the “Z-Wave chip” image).

Controlling nodes implementing earlier versions of the Firmware Update Meta Data CC do not support status codes defined for newer versions. Therefore, a node returning the Firmware Update Meta Data CC:007A.04.07.11.005 Status Report Command MUST comply with the version implemented on the controlling device (device sending the image). The device returning the Firmware Update Meta Data Status Report can identify the version of the controlling device based on the Firmware Update Meta Data Request Get. This may be done as follows:

- If the Firmware Update Meta Data Request Get does not include Firmware Target and Fragment Size (8 bytes), the controller is version 1 or 2
- If the Firmware Update Meta Data Request Get includes Firmware Target and Fragment Size but not Activation (11 bytes), the controller is version 3
- If the Firmware Update Meta Data Request Get includes Activation (12 bytes), the controller is version 4

3.2.18.5 Firmware Update Activation Set Command

This command is used to initiate the programming of a previously transferred firmware image. Refer to the Firmware Update Meta Data Status Report Command Status code 0xFD.

CC:007A.04.08.13.001 This command MAY be issued directly by a controlling node or MAY be scheduled for later execution via the Schedule Command Class.

CC:007A.04.08.11.001 The Firmware Update Activation Status Report Command MUST be returned in response to this command.

Table 3.65: Firmware Update Activation Set Command version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_ACTIVATION_SET (0x08)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							

For fields' description, refer to the Firmware Update Meta Data Request Get Command.

3.2.18.6 Firmware Update Activation Status Report

This command is used to advertise the result of a firmware update operation initiated by the Firmware Update Activation Set Command.

CC:007A.04.09.11.001 The Firmware Update Activation Status Report fields MUST reflect the values specified in the Firmware Update Activation Set Command.

Table 3.66: Firmware Update Activation Status Report Command version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_ACTIVATION_STATUS_REPORT (0x09)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Firmware Update Status							

For fields not described below, refer to the Firmware Update Meta Data Request Get Command.

Firmware Update Status (8 bits)

CC:007A.04.09.11.002 This field MUST comply with Table 3.74.

3.2.19 Firmware Update Meta Data Command Class, version 5

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to or from a Z-Wave node.

3.2.19.1 Compatibility considerations

CC:007A.05.00.21.001 A device implementing Firmware Update Meta Data Command Class, version 5 MUST also implement Firmware Update Meta Data Command Class, version 4.

Firmware Update Meta Data Command Class, version 5 is backwards compatible with Firmware Update Meta Data Command Class, version 4.

All commands and fields not mentioned in this version remain unchanged from version 4.

3.2.19.1.1 New commands, fields and values

The Firmware Update Meta Data Command Class, version 5 introduces the support of a hardware identifier to uniquely identify firmware images that can be loaded on devices. This allows devices running identical software version on different hardware revisions to receive the correct firmware image.

The following commands are extended from version 4:

- Firmware Meta Data Report Command
- Firmware Update Meta Data Request Get Command
- Firmware Update Meta Data Request Report Command
- Firmware Update Meta Data Status Report Command
- Firmware Update Activation Set Command
- Firmware Update Activation Status Report Command

The Firmware Update Meta Data Command Class, version 5 also introduces the support firmware download from the Z-Wave node to the controller.

The following commands are introduced:

- Firmware Update Meta Data Prepare Get Command
- Firmware Update Meta Data Prepare Report Command

3.2.19.2 Interoperability Considerations

3.2.19.2.1 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command. Version 5 of the Firmware Update Meta Data Report command uses the version 2 format.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report version 2 Command will consider the 16-bit Checksum field to be the last two bytes of the Data field.

Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report version 1 Command will consider the last two bytes of the Data field to be the 16-bit Checksum field.

In either case, the transfer will fail.

Therefore, a controlling device implementing the Firmware Update Meta Data Command Class, version 2 MUST do the following when initiating the transfer of a firmware image:

CC:007A.05.00.31.001

1. Request the version of the Firmware Update Meta Data Command Class from the target device
2. Use the version of the Firmware Update Meta Data Report implemented by the target device
3. Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report

3.2.19.2.2 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. There is no way to identify what method has been used for calculating the checksum for nodes implementing version 1 to version 4. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory.

For more details about the checksum calculation, refer to *CRC-CCITT Source Code*.

3.2.19.3 Firmware Meta Data Report Command

This command is used to advertise the status of the current firmware in the device.

Table 3.67: Firmware Meta Data Report Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_REPORT (0x02)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware 0 ID 1							
Firmware 0 ID 2							
Firmware 0 Checksum 1							
Firmware 0 Checksum 2							
Firmware Upgradable							
Number of Firmware Targets							
Max Fragment Size 1							
Max Fragment Size 2							
Firmware 1 ID 1							
Firmware 1 ID 2							
...							
Firmware N ID 1							
Firmware N ID 2							
Hardware Version							

All fields not described below remain unchanged from version 4.

Firmware 0 Checksum (16 bits)

The checksum field is used to carry the checksum of the Firmware 0 Image.

CC:007A.05.02.13.001 Supporting nodes MAY set this field to 0x00. In this case the field is unused.

CC:007A.05.02.11.006 Values in the range 0x0001..0xFFFF indicate that the supporting node MUST advertise the checksum of its firmware 0 image.

CC:007A.05.02.11.001 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*.

Hardware Version (8 bits)

CC:007A.05.02.11.002 This field MUST report a value which is unique to this particular version of the product.

CC:007A.05.02.11.003 It MUST be possible to uniquely identify applicable firmware images via the Manufacturer ID, Firmware ID and the Hardware Version fields. This information allows selecting a firmware image that is guaranteed to work with this particular version of the product.

CC:007A.05.02.11.004 The Hardware Version field MUST apply to the entire product and not only to the version of the Z-Wave radio chip.

CC:007A.05.02.11.005 This field MUST report the same value as the Version CC Version Report Command: Hardware Version field.

3.2.19.4 Firmware Update Meta Data Request Get Command

This command is used to request that a firmware update is initiated by the node receiving this command.

Table 3.68: Firmware Update Meta Data Request Get Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET (0x03)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Reserved							Activation
Hardware Version							

All fields not described below remain unchanged from version 4.

Firmware Checksum (16 bits)

CC:007A.05.03.11.001 The checksum field MUST carry the checksum of the firmware image about to be transferred.

CC:007A.05.03.11.002 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*.

Hardware Version (8 bits)

Refer to Section 3.2.19.3 Firmware Meta Data Report Command.

3.2.19.5 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

Table 3.69: Firmware Update Meta Data Request Report Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT (0x04)							
Status							

Status (8 bits)

CC:007A.05.04.11.001 This field MUST comply with Table 3.81.

3.2.19.6 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

CC:007A.05.07.11.001 The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

CC:007A.05.07.11.002 A device MUST NOT issue the Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

Table 3.70: Firmware Update Meta Data Status Report Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT (0x07)							
Status							
WaitTime MSB							
WaitTime LSB							

All fields not described below are the same as in version 4.

Status (8 bits)

CC:007A.05.07.11.003 This field MUST comply with Table 3.71.

CC:007A.05.07.13.001 The status code 0xFD MAY be used by a version 4 device for confirming that an image has been successfully transferred but that the actual firmware update will not be performed until a Firmware Update Activation Set Command is received.

CC:007A.05.07.13.002 The Firmware Update Activation Set Command MAY be delayed for any period of time. The delay may be controlled via the Schedule Command Class.

CC:007A.05.07.13.003 The status code 0xFE MAY be used for confirming the successful transfer of an image which does not necessitate a restart, e.g. security a certificate.

CC:007A.05.07.11.004 The status code 0xFE MUST NOT be advertised after the transfer of an image for the Firmware ID 0 target (the "Z-Wave chip" image).

CC:007A.05.07.11.005 Controlling nodes implementing earlier versions of the Firmware Update Meta Data CC do not support status codes defined for newer versions. Therefore, a device returning the Firmware Update Meta Data Status Report Command MUST comply with the version implemented on the controlling device (device sending the image). The device returning the Firmware Update Meta Data Status Report can identify the version of the controlling device based on the Firmware Update Meta Data Request Get. This may be done as follows:

- If the Firmware Update Meta Data Request Get does not include Firmware Target and Fragment Size (8 bytes), the controller is version 1 or 2
- If the Firmware Update Meta Data Request Get includes Firmware Target and Fragment Size but not Activation (11 bytes), the controller is version 3
- If the Firmware Update Meta Data Request Get includes Activation but not Hardware Version (12 bytes), the controller is version 4
- If the Firmware Update Meta Data Request Get includes Hardware Version (13 bytes), the controller is version 5.

Table 3.71: Firmware Update Meta Data Status Report::Status Encoding

Status	Description	Version
0x00	The device was unable to receive the requested firmware data without checksum error. Number of retries and request sequence of missing frames are implementation specific. The image is not stored.	1
0x01	The device was unable to receive the requested firmware data. Number of retries and request sequence of missing frames are implementation specific. The image is not stored.	1
0x02	The transferred image does not match the Manufacturer ID. The image is not stored.	4
0x03	The transferred image does not match the Firmware ID. The image is not stored.	4
0x04	The transferred image does not match the Firmware Target. The image is not stored.	4
0x05	Invalid file header information. The image is not stored.	4
0x06	Invalid file header format. The image is not stored.	4
0x07	Insufficient memory. The image is not stored.	4
0x08	The transferred image does not match the Hardware version. The image is not stored.	5
...	*Reserved*	...
0xFD	Firmware image downloaded successfully, waiting for activation command.	4
0xFE	New image was successfully stored in temporary non-volatile memory. The device does not restart itself. This Status code MUST NOT be used when updating the Z-Wave chip image	3
0xFF	New image was successfully stored in temporary non-volatile memory and/or applied successfully. The supporting node MAY restart itself. In this case, version 3 or newer supporting nodes SHOULD use the WaitTime to indicate that reboot has not been performed yet.	1

CC:007A.05.07.11.006 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

3.2.19.7 Firmware Update Activation Set Command

This command is used to initiate the programming of a previously transferred firmware image.

Refer to the Firmware Update Meta Data Status Report Command Status code 0xFD.

CC:007A.05.08.13.001 This command MAY be issued directly by a controlling node or MAY be scheduled for later execution via the Schedule Command Class.

CC:007A.05.08.11.001 The Firmware Update Activation Status Report Command MUST be returned in response to this command.

Table 3.72: Firmware Update Activation Set Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_ACTIVATION_SET (0x08)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Hardware Version							

All fields not described below remain unchanged from version 4.

Firmware Checksum (16 bits)

CC:007A.05.08.11.002 The checksum field MUST carry the checksum of the firmware image about to be activated.

CC:007A.05.08.11.003 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*.

Hardware Version (8 bits)

Refer to Section 3.2.19.3 Firmware Meta Data Report Command.

3.2.19.8 Firmware Update Activation Status Report Command

This command is used to advertise the result of a firmware update operation initiated by the Firmware Update Activation Set Command.

Table 3.73: Firmware Update Activation Status Report Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_ACTIVATION_STATUS_REPORT (0x09)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Firmware Update Status							
Hardware Version							

All fields not described below remain unchanged from version 4.

Firmware Checksum (16 bits)

CC:007A.05.09.11.001 The checksum field MUST carry the checksum of the firmware image activated by the Firmware Update Activation Set Command.

CC:007A.05.09.11.002 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*.

Hardware Version (8 bits)

Refer to Section 3.2.19.3 Firmware Meta Data Report Command.

Firmware Update Status (8 bits)

CC:007A.05.09.11.003 The Firmware Update Status field MUST comply with Table 3.74.

Table 3.74: Firmware Update Activation Status Report::Firmware Update Status Encoding

Status	Description	Version
0x00	Invalid combination of manufacturer ID, firmware ID and Hardware Version or Firmware Target. The received image will not be stored.The device was unable to receive the requested firmware data without checksum error.	4
0x01	Error activating the firmware. Last known firmware image has been restored.	4
...	*Reserved*	...
0xFF	Firmware update completed successfully.	4

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

3.2.19.9 Firmware Update Meta Data Prepare Get Command

This command is used to request that a firmware download is initiated by the node sending this command.

CC:007A.05.0A.11.001 The Firmware Update Meta Data Prepare Report Command MUST be returned in response to this command when the receiving node is ready to send the requested firmware image.

CC:007A.05.0A.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.05.0A.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

CC:007A.05.0A.12.001 Any image transferred via the Firmware Update Meta Data Command Class SHOULD include a fingerprint value to enable the validation of the integrity of the entire image after the transfer. The

CC:007A.05.0A.11.004 image and the fingerprint value MUST be packed in one entity which can be stored in one file and transferred as one entity.

Table 3.75: Firmware Update Meta Data Prepare Get Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_PREPARE_GET (0x0A)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Hardware Version							

Refer to Section 3.2.19.4 Firmware Update Meta Data Request Get Command for field description.

3.2.19.10 Firmware Update Meta Data Prepare Report Command

This command is used to advertise if the firmware image has been prepared and is ready to be transferred.

Table 3.76: Firmware Update Meta Data Prepare Report Command version 5

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_PREPARE_REPORT (0x0B)							
Status							
Firmware Checksum 1							
Firmware Checksum 2							

Status (8 bits)

CC:007A.05.0B.11.001 The Status field MUST comply with Table 3.77.

Table 3.77: Firmware Update Meta Data Prepare Report::Status encoding

Status	Description	Version
0x00	ERROR. Invalid combination of Manufacturer ID and Firmware ID. The receiving node MUST NOT initiate the firmware download.	5
0x01	ERROR. Device expected an authentication event to enable firmware update. The receiving node MUST NOT initiate the firmware download.	5
0x02	ERROR. The requested Fragment Size exceeds the Max Fragment Size. The receiving node MUST NOT initiate the firmware download.	5
0x03	ERROR. This firmware target is not downloadable. The receiving node MUST NOT initiate the firmware download.	5
0x04	ERROR. Invalid Hardware Version. The receiving node MUST NOT initiate the firmware download.	5
0xFF	OK. The receiving node can initiate the firmware download of the target specified in the Firmware Update Meta Data Prepare Get Command.	5

CC:007A.05.0B.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Firmware Checksum (16 bits)

CC:007A.05.0B.11.003 The checksum field MUST carry the checksum of the firmware image about to be transferred.

CC:007A.05.0B.12.001 The checksum field value SHOULD be 0x00 if the Status field is set to a different value than 0xFF.

CC:007A.05.0B.11.004 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*.

3.2.19.11 Examples

3.2.19.11.1 Identifying firmware revisions

The different fields Manufacturer ID, Firmware ID and Hardware version are used to identify compatible firmware image with the product.

The Version Command Class is also used for identifying the Firmware version/subversion

An example of the different fields' usage for a wall outlet is shown in Table 3.78.

Table 3.78: Labeling Different Firmware Revisions

	Firmware Identification			
Product	Manufacturer ID	Hardware Version	Firmware ID	Firmware version/ subversion
Initial Wall Outlet	0x0001	0x01	0x0001	0x01/0x01
New Revision				
Fix previous bugs	0x0001	0x01	0x0001	0x01/0x02
New revision				
Introducing new features (e.g. multiple press)	0x0001	0x01	0x0001	0x02/0x01
New revision				
US version (initial version)	0x0001	0x01	0x0001	0x02/0x01
China version (matching local requirements)	0x0001	0x01	0x0002	0x02/0x01
New revision				
Proximity Sensor added (US)	0x0001	0x02	0x0001	0x02/0x01
Proximity Sensor added (China)	0x0001	0x02	0x0002	0x02/0x01
New revision				
Adding S2 Capability (without proximity sensor, US)	0x0001	0x01	0x0001	0x03/0x01
Adding S2 Capability (without proximity sensor, China)	0x0001	0x01	0x0002	0x03/0x01
Adding S2 Capability (with proximity sensor, US)	0x0001	0x02	0x0001	0x03/0x01
Adding S2 Capability (with proximity sensor, China)	0x0001	0x02	0x0002	0x03/0x01

An illustration of a controller retrieving the firmware information is given in Figure 3.5

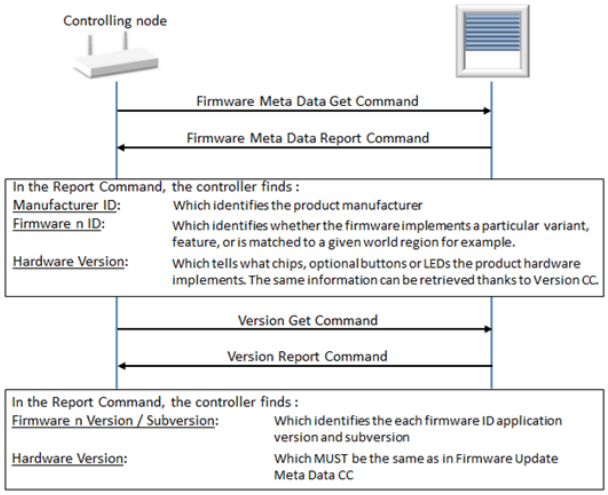


Figure 3.5: Identifying Compatible Firmware Images

3.2.19.11.2 Firmware download

The use of the Firmware Update Meta Data Prepare Get and Firmware Update Meta Data Prepare Report commands is illustrated in Figure 3.6.

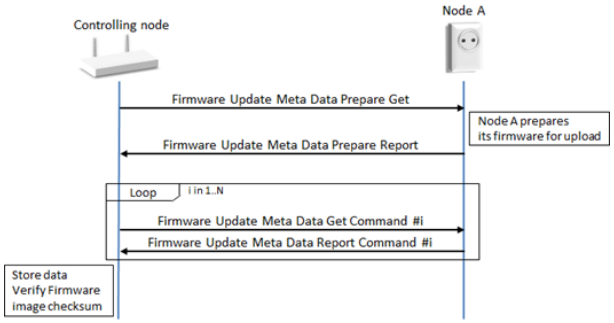


Figure 3.6: Firmware Download Flow Diagram

3.2.20 Firmware Update Meta Data Command Class, version 6

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to or from a Z-Wave node.

3.2.20.1 Compatibility considerations

This version introduces new status codes for firmware update request responses in the Firmware Meta Data Report Command and Firmware Update Meta Data Request Report Command. It also allows a supporting node to advertise whether the application functionalities will be available during a Firmware Update or not.

Firmware Update Meta Data Command Class, version 6 is backwards compatible with Firmware Update Meta Data Command Class, version 5.

CC:007A.06.00.21.001 All commands and fields not mentioned in this version MUST remain unchanged from version 5.

3.2.20.2 Firmware Meta Data Report Command

This command is used to advertise the status of the current firmware in the device.

Table 3.79: Firmware Meta Data Report Command version 6

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_REPORT (0x02)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware 0 ID 1							
Firmware 0 ID 2							
Firmware 0 Checksum 1							
Firmware 0 Checksum 2							
Firmware Upgradable							
Number of Firmware Targets							
Max Fragment Size 1							
Max Fragment Size 2							
Firmware 1 ID 1							
Firmware 1 ID 2							
...							
Firmware N ID 1							
Firmware N ID 2							
Hardware Version							
Reserved							CC

All fields not described below remain unchanged from version 5.

CC (1 bits)

This field is used to advertise if the supporting node's Command Classes functionality will continue to function normally during Firmware Update transfer.

If all other Command Classes functionalities will function normally during firmware update image
CC:007A.06.03.11.001 transfer, this field MUST be set to 1.

If any Command Class functionality will not function normally during firmware update image transfer,
CC:007A.06.03.11.002 this field MUST be set to 0.

For example, if a node is unable to issue Notification Reports during firmware update, this field MUST be set to 0.

3.2.20.3 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

Table 3.80: Firmware Update Meta Data Request Report Command version 6

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT (0x04)							
Status							

Status (8 bits)

CC:007A.06.04.11.001 This field MUST comply with Table 3.81.

Table 3.81: Firmware Update Meta Data Request Report::Status Encoding

Status	Description	Version
0x00	ERROR. Invalid combination of Manufacturer ID and Firmware ID. The device will not initiate the firmware update.	1
0x01	ERROR. Device expected an authentication event to enable firmware update. The device will not initiate the firmware update.	1
0x02	ERROR. The requested Fragment Size exceeds the Max Fragment Size. The device will not initiate the firmware update.	3
0x03	ERROR. This firmware target is not upgradable. The device will not initiate the firmware update.	3
0x04	ERROR. Invalid Hardware Version. The device will not initiate the firmware update.	5
0x05	ERROR: Another firmware image is current being transferred. The device will not initiate the firmware update.	6
0x06	ERROR: Insufficient battery level The device will not initiate the firmware update.	6
0xFF	OK. The device will initiate the firmware update of the target specified in the Firmware Update Meta Data Request Get Command.	1

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

3.2.21 Firmware Update Meta Data Command Class, version 7

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to or from a Z-Wave node.

3.2.21.1 Compatibility considerations

This version introduces the ability to advertise if subsequent firmware activation is supported or not by a node.

Firmware Update Meta Data Command Class, version 7 is backwards compatible with Firmware Update Meta Data Command Class, version 6.

CC:007A.07.00.21.001 All commands and fields not mentioned in this version MUST remain unchanged from version 6.

3.2.21.2 Firmware Meta Data Report Command

This command is used to advertise the status of the current firmware in the device.

Table 3.82: Firmware Meta Data Report Command version 7

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_REPORT (0x02)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware 0 ID 1							
Firmware 0 ID 2							
Firmware 0 Checksum 1							
Firmware 0 Checksum 2							
Firmware Upgradable							
Number of Firmware Targets							
Max Fragment Size 1							
Max Fragment Size 2							
Firmware 1 ID 1							
Firmware 1 ID 2							
...							
Firmware N ID 1							
Firmware N ID 2							
Hardware Version							
Reserved						Activa-tion	CC

All fields not described below remain unchanged from version 6.

Activation (1 bits)

This field is used to advertise if the supporting node supports subsequent activation after Firmware Update transfer.

If the sending node supports the subsequent activation of firmware after downloading, this field MUST be set to 1.

If the sending node does not support the subsequent activation of firmware after downloading, this field MUST be set to 0.

3.2.21.3 Firmware Update Meta Data Request Get Command

Table 3.83: Firmware Meta Data Request Get Command version 7

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET (0x03)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Reserved							Activation
Hardware Version							

All fields not described below remain unchanged from version 3.

Activation (1 bit)

The Activation field is used to advertise if the receiving node must delay the actual firmware activation after the file transfer.

A node advertising no support for “activation” in the Firmware Meta Data Report Command MUST ignore this field and activate the Firmware immediately after download.

If the receiving node advertises support for Activation in the Firmware Meta Data Report Command:

- The value ’1’ MUST indicate to delay the actual firmware update.

  When delaying the firmware activation, the supporting node MUST set the Status code field to 0xFD in the Firmware Update Meta data Status Report Command.
- The value ’0’ MUST indicate that the receiving device MUST NOT delay the firmware update and activate the firmware immediately after download.

3.2.22 Firmware Update Meta Data Command Class, version 8

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to or from a Z-Wave node.

3.2.22.1 Compatibility Considerations

This version introduces the ability to perform firmware transfer without security encapsulation and resume aborted firmware update attempts.

Firmware Update Meta Data Command Class, version 8 is backwards compatible with the Firmware Update Meta Data Command Class, version 7.

3.2.22.2 Interoperability Considerations

3.2.22.2.1 Interoperability with v1 devices

Version 2 of the *Firmware Update Meta Data Report Command* introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the *Firmware Update Meta Data Report Command*. Subsequent versions of the Firmware Update Meta Data Command Class command use the version 2 format.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a *Firmware Update Meta Data Report Command* using version 2 format will consider the 16-bit Checksum field to be the last two bytes of the Data field.

Similarly, a version 2 implementation receiving a *Firmware Update Meta Data Report Command* version 1 will consider the last two bytes of the Data field to be the 16-bit Checksum field.

In either case, the transfer will fail.

CC:007A.08.00.31.001 Therefore, a node controlling the Firmware Update Meta Data Command Class, version 2 MUST do the following when initiating the transfer of a firmware image:

1. Request the version of the Firmware Update Meta Data Command Class from the target device
2. Use the version of the *Firmware Update Meta Data Report Command* implemented by the supporting node.
3. Determine the number of Data bytes that can fit into the *Firmware Update Meta Data Report Command* so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the *Firmware Update Meta Data Report Command*.

3.2.22.2.2 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. There is no way to identify what method has been used for calculating the checksum for nodes implementing version 1 to version 4. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

CC:007A.08.00.31.002 From Version 5 onwards, the checksum calculation method is mandatory. A version 5 supporting node MUST calculate checksums using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). For more details about the checksum calculation, refer to *CRC-CCITT Source Code*.

3.2.22.2.3 Firmware Activation

CC:007A.08.00.31.003 Nodes supporting version 7 and newer MUST wait for a *Firmware Update Activation Set Command* before activating a firmware after download, if instructed to use activation in the *Firmware Update Meta Data Request Get Command*.

CC:007A.08.00.32.001 Nodes supporting version 6 or older are not mandated to so do. Controlling nodes SHOULD NOT rely on the activation functionality for these versions.

3.2.22.3 Firmware Data fields

Firmware are identified using the following fields:

3.2.22.3.1 Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device. Manufacturer identifiers can be found in **Z-Wave Manufacturer ID List.xlsx**.

CC:007A.08.00.11.001 The first byte MUST be the most significant byte.

3.2.22.3.2 Firmware ID (16 bits)

Each firmware target has a Firmware ID. Firmware IDs may be identical for multiple targets.

CC:007A.08.00.11.002 The first byte MUST be the most significant byte.

3.2.22.3.3 Firmware Checksum (16 bits)

CC:007A.08.00.11.003 The checksum field MUST carry the checksum of the firmware image.

CC:007A.08.00.11.004 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*

3.2.22.3.4 Firmware target (8 bits)

This field MUST be used to identify the firmware image to be updated. The firmware images MUST be identified according to Table 3.84.

Table 3.84: Firmware Update Meta Data Request Get Command - Firmware Target encoding

Firmware Target	Description
0 (0x00)	Firmware image targeted for the Z-Wave chip.
1 (0x01)	Firmware image intended for target 1 defined by the manufacturer.
...	...
255 (0xFF)	Firmware image intended for target 255 defined by the manufacturer.

3.2.22.3.5 Hardware Version (8 bits)

CC:007A.08.00.11.005 This field MUST report a value which is unique to this particular version of the product.

CC:007A.08.00.11.006 It MUST be possible to uniquely identify applicable firmware images via the Manufacturer ID, Firmware ID and the Hardware Version fields. This information allows selecting a firmware image that is guaranteed to work with this particular version of the product.

CC:007A.08.00.11.007 The Hardware Version field MUST apply to the entire product and not only to the version of the Z-Wave radio chip.

CC:007A.08.00.11.008 This field MUST report the same value as the Version CC Version Report Command: Hardware Version field.

3.2.22.4 Firmware Meta Data Get Command

This command is used to request firmware information and capabilities for a supporting node.

This command was introduced in version 1.

CC:007A.08.01.11.001 The *Firmware Meta Data Report Command* MUST be returned in response to this command.

CC:007A.08.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.08.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.85: Firmware Meta Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_GET (0x01)							

3.2.22.5 Firmware Meta Data Report Command

This command is used to advertise firmwares and capabilities of the supporting nodes.

This command was introduced in version 1.

Table 3.86: Firmware Meta Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_MD_REPORT (0x02)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware 0 ID 1							
Firmware 0 ID 2							
Firmware 0 Checksum 1							
Firmware 0 Checksum 2							
Firmware Upgradable							
Number of Firmware Targets							
Max Fragment Size 1							
Max Fragment Size 2							
Firmware 1 ID 1							
Firmware 1 ID 2							
...							
Firmware N ID 1							
Firmware N ID 2							
Hardware Version							
Reserved				Resume	Non-secure	Activation	CC

Manufacturer ID (16 bits)

Refer to *Manufacturer ID (16 bits)*.

Firmware 0 ID (16 bits)

The Firmware 0 ID field is dedicated to target 0, i.e. the Z-Wave chip.

CC:007A.08.02.11.001 A manufacturer MUST assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. The combination of Manufacturer ID and Firmware ID provides unique identification of a firmware image that is guaranteed to work with a particular component of a particular product.

CC:007A.08.02.11.002 The first byte MUST be the most significant byte.

A user may request Manufacturer Specific information to get more detailed information on the product.

CC:007A.08.02.12.001 A controlling node SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

Firmware 0 Checksum (16 bits)

The checksum field is used to carry the checksum of the Firmware 0 Image.

CC:007A.08.02.13.001 Supporting nodes MAY set this field to 0x00. In this case the field is unused.

CC:007A.08.02.11.003 Values in the range 0x0001..0xFFFF indicate that the supporting node MUST advertise the checksum of its firmware 0 image.

CC:007A.08.02.11.004 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*

Firmware Upgradable (8 bits)

This field defines whether the Z-Wave chip is firmware upgradable.

CC:007A.08.02.11.005 The value 0x00 MUST indicate that the firmware 0 image is not upgradable.

CC:007A.08.02.11.006 The value 0xFF MUST indicate that the firmware 0 image is upgradable.

Number of Firmware Targets (8 bits)

CC:007A.08.02.11.007 The Number of Firmware Targets field MUST report the number of firmware IDs following this field. The Firmware 0 ID field is not included. The field MUST be zero if the device only implements a Firmware 0 target, i.e. the Z-Wave chip.

Max Fragment Size (16 bits)

CC:007A.08.02.11.008 CC:007A.08.02.13.002 The Max Fragment Size field MUST report the maximum number of Data bytes that a device is able to receive at a time. A sending node MAY send shorter fragments. The fragment size actually used is indicated in the *Firmware Update Meta Data Request Get Command* and confirmed in the *Firmware Update Meta Data Request Report Command*.

CC:007A.08.02.11.009 The Max Fragment Size may be longer than the supported frame length of Z-Wave if the image is to be transferred over e.g. IP. If the image is to be transferred over Z-Wave, the sending node MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

Firmware 1..N ID (16 bits)

The Firmware 1 ID field indicates the Firmware ID that MUST be used for target 1. The Firmware 2 ID field represents target 2. And so on.

CC:007A.08.02.11.00A The first byte MUST be the most significant byte.

CC:007A.08.02.12.002 A controlling device SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

Hardware Version (8 bits)

Refer to *Hardware Version (8 bits)*.

CC (1 bit)

This field is used to advertise if the supporting node's Command Classes functionality will continue to function normally during Firmware Update transfer.

CC:007A.08.02.11.00B If all other Command Classes functionalities will function normally during firmware update image transfer, this field MUST be set to 1.

CC:007A.08.02.11.00C If any Command Class functionality will not function normally during firmware update image transfer, this field MUST be set to 0. For example, if a node is unable to issue Notification Reports during firmware update, this field MUST be set to 0.

Activation (1 bit)

This field is used to advertise if the supporting node supports subsequent activation after Firmware Update transfer.

CC:007A.08.02.11.00D If the sending node supports the subsequent activation of firmware after downloading, this field MUST be set to 1.

CC:007A.08.02.11.00F If the sending node does not support the subsequent activation of firmware after downloading, this field MUST be set to 0.

Non-secure (1 bit)

This field is used to indicate if the node supports non-secure firmware transfers.

CC:007A.08.02.11.010 The value 0 MUST indicate that the non-secure transfer functionality is not supported.

CC:007A.08.02.11.011 The value 1 MUST indicate that the non-secure transfer functionality is supported.

Resume (1 bit)

This field is used to indicate if the node supports resuming aborted firmware transfers.

CC:007A.08.02.11.012 The value 0 MUST indicate that the Resume functionality is not supported.

CC:007A.08.02.11.013 The value 1 MUST indicate that the Resume functionality is supported.

3.2.22.6 Firmware Update Meta Data Request Get Command

The Firmware Update Meta Data Request Get Command is used to request that a firmware update is initiated.

This command was introduced in version 1.

CC:007A.08.03.11.001 The *Firmware Update Meta Data Request Report Command* MUST be returned in response to this command.

CC:007A.08.03.11.002 The firmware update MUST NOT be initiated if the Manufacturer ID and the Firmware ID do not match the actual firmware image values.

CC:007A.08.03.11.003 The firmware update MUST be aborted if the checksum does not match the calculated checksum after the firmware image has been transferred.

CC:007A.08.03.11.004 This command MUST NOT be issued via multicast addressing.

CC:007A.08.03.11.005 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.87: Firmware Update Meta Data Request Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET (0x03)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Checksum 1							
Firmware Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Reserved					Resume	Non-secure	Activation
Hardware Version							

Manufacturer ID (16 bits)

Refer to *Manufacturer ID (16 bits)*.

Firmware ID (16 bits)

CC:007A.08.03.11.006 The Firmware ID MUST match the specified target of the actual device.

CC:007A.08.03.11.007 The firmware update MUST NOT be initiated if the Firmware ID does not match the specified target of the actual device. The first byte MUST be the most significant byte.

Firmware Checksum (16 bits)

Refer to *Firmware Checksum (16 bits)*.

Firmware Target (8 bits)

Refer to *Firmware target (8 bits)*.

Fragment Size (16 bits)

CC:007A.08.03.11.008 The Fragment Size field MUST report the fragment size that is to be used for firmware fragments. A receiving device MUST use this fragment size for the firmware update. The fragment size is not exchanged during the actual firmware update.

CC:007A.08.03.11.009 The Fragment Size MUST NOT exceed the Max Fragment Size value of the *Firmware Meta Data Report Command*. A version 1 *Firmware Update Meta Data Request Get Command* does not carry a CC:007A.08.03.11.00A Fragment Size field. A receiving node MUST determine the number of Data bytes from the length of the first received frame, the use of security encapsulation as well as the presence of Checksum bytes in the *Firmware Meta Data Report Command*.

Hardware Version (8 bits)

Refer to *Firmware Meta Data Report Command*.

Activation (1 bit)

CC:007A.08.03.11.014 The Activation field is used to advertise if the receiving node MUST delay the actual firmware activation after the file transfer.

CC:007A.08.03.11.00B A node advertising no support for "activation" in the *Firmware Meta Data Report Command* MUST ignore this field and activate the Firmware immediately after download.

If the receiving node advertises support for Activation in the *Firmware Meta Data Report Command*:

CC:007A.08.03.11.00C
- The value '1' MUST indicate to delay the actual firmware update. When delaying the firmware activation, the supporting node MUST set the Status code field to 0xFD in the Firmware Update Meta data Status Report Command.

CC:007A.08.03.11.00D
- The value '0' MUST indicate that the receiving device MUST NOT delay the firmware update and activate the firmware immediately after download.

Resume (1 bit)

This field is used to request the supporting node to resume the firmware transfer from the last attempt.

CC:007A.08.03.11.00E Supporting nodes advertising no support for the Resume functionality in the *Firmware Meta Data Report Command* MUST ignore this field and start the firmware update transfer from the Report number 1.

CC:007A.08.03.11.00F The value 0 MUST indicate that the Firmware Update transfer MUST start from Report Number 1.

CC:007A.08.03.11.010 The value 1 MUST indicate that the Firmware Update transfer SHOULD start from The last received report number in a previous transfer attempt for this firmware image.

Non-Secure (1 bit)

This field is used to request the supporting node to accept receiving firmware fragments sent without security encapsulation.

CC:007A.08.03.11.011 Supporting nodes advertising no support for the Non-Secure functionality in the *Firmware Meta Data Report Command* MUST ignore this field and only accept *Firmware Update Meta Data Report Command* received at their highest granted security class.

CC:007A.08.03.11.012 The value 0 MUST indicate that the Firmware Update transfer MUST be performed securely and the supporting node MUST only accept *Firmware Update Meta Data Report Command* received at its highest granted security class.

CC:007A.08.03.11.013 The value 1 MUST indicate that the Firmware Update transfer SHOULD be done without security encapsulation and the supporting node SHOULD accept *Firmware Update Meta Data Report Command* received without security encapsulation.

3.2.22.7 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

This command was introduced in version 1.

Table 3.88: Firmware Update Meta Data Request Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT (0x04)							
Status							
Reserved					Resume	Non-secure	Reserved

Status (8 bits)

CC:007A.08.04.11.001 This field MUST comply with Table 3.89.

Table 3.89: Firmware Update Meta Data Request Report Command - Status encoding

Status	Description	Version
0x00	*ERROR*. Invalid combination of Manufacturer ID and Firmware ID. The device will not initiate the firmware update.	1
0x01	*ERROR*. Device expected an authentication event to enable firmware update. The device will not initiate the firmware update.	1
0x02	*ERROR*. The requested Fragment Size exceeds the Max Fragment Size. The device will not initiate the firmware update.	3
0x03	*ERROR*. This firmware target is not upgradable. The device will not initiate the firmware update.	3
0x04	*ERROR*. Invalid Hardware Version. The device will not initiate the firmware update.	5
0x05	*ERROR*. Another firmware image is current being transferred. The device will not initiate the firmware update.	6
0x06	*ERROR*. Insufficient battery level. The device will not initiate the firmware update.	6
0xFF	*OK*. The device will initiate the firmware update of the target specified in the *Firmware Update Meta Data Request Get Command*.	1

CC:007A.08.04.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Resume (1 bit)

This field is used to indicate if the supporting node accepted to resume the firmware transfer from the last attempt.

CC:007A.08.04.11.003 Supporting nodes advertising no support for the Resume functionality in the *Firmware Meta Data Report Command* MUST set this field to 0 and start the firmware update transfer from the Report number 1.

CC:007A.08.04.11.004 The value 0 MUST indicate that the supporting node did not (or cannot) accept to restart the previous firmware update and it will start from Report Number 1.

CC:007A.08.04.11.005 The value 1 MUST indicate that the supporting node accepted to resume the previous firmware update and it will start from the last received report number in a previous transfer attempt for this firmware image.

Non-Secure (1 bit)

This field is used to indicate if the supporting node accepted to receive firmware fragments sent without security encapsulation.

CC:007A.08.04.11.006 Supporting nodes advertising no support for the Non-Secure functionality in the *Firmware Meta Data Report Command* MUST set this field to 0 and and only accept *Firmware Update Meta Data Report Command* received at their highest granted security class.

CC:007A.08.04.11.007 The value 0 MUST indicate that the Firmware Update transfer will be performed securely and the supporting node MUST only accept *Firmware Update Meta Data Report Command* received at its highest granted security class.

CC:007A.08.04.11.008 The value 1 MUST indicate that the Firmware Update transfer will be performed without security encapsulation and the supporting node MUST accept *Firmware Update Meta Data Report Command* received without security encapsulation.

3.2.22.8 Firmware Update Meta Data Get Command

This command is used to request one or more *Firmware Update Meta Data Report Command*.

This command was introduced in version 1.

CC:007A.08.05.11.001 The *Firmware Update Meta Data Report Command* MUST be returned in response to this command.

CC:007A.08.05.13.001 CC:007A.08.05.13.002 The transmission of the next *Firmware Update Meta Data Get Command* MAY be delayed if time is required to store the most recent firmware fragment in temporary non-volatile memory. A node MAY request multiple *Firmware Update Meta Data Report Command* to improve throughput.

CC:007A.08.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.08.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Two exceptions may apply: A noise burst may interfere the transmission or the data source may stop returning *Firmware Update Meta Data Report Command*.

CC:007A.08.05.12.001 To accommodate for bursts of RF noise, a device receiving data SHOULD repeatedly retransmit the same *Firmware Update Meta Data Get Command* every 10 seconds in case no *Firmware Update Meta Data Report Command* are received.

CC:007A.08.05.12.002 A device receiving data SHOULD stop retransmitting Firmware Update Meta Data Get Commands and abort the ongoing firmare update 2 minutes after the last successful reception of a *Firmware Update Meta Data Report Command*.

Table 3.90: Firmware Update Meta Data Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_GET (0x05)							
Number of Reports							
Res	Report number 1						
Report number 2							

Res (1 bit)

CC:007A.08.05.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Reports (8 bits)

Number of *Firmware Update Meta Data Report Command* to be received in response to this *Firmware Update Meta Data Get Command*.

Report number (15 bits)

CC:007A.08.05.11.005 The Report number field indicates the *Firmware Update Meta Data Report Command* to be requested. The report number values MUST be a sequence starting from 1. The first byte (Report number 1) is the most significant byte.

3.2.22.9 Firmware Update Meta Data Report Command

The Firmware Update Meta Data Report Command is used to transfer a firmware image fragment.

This command was introduced in version 1.

CC:007A.08.06.11.001 If sending more than a single *Firmware Update Meta Data Report Command* at a time, a node MUST apply a delay between each transmitted command. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 1 frame back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

CC:007A.08.06.12.001 If needed, a controlling node SHOULD abort an ongoing transfer by responding to a *Firmware Update Meta Data Get Command* with a *Firmware Update Meta Data Report Command* with the Last bit enabled and the Data fields intentionally corrupted.

This will invalidate the calculated firmware checksum, which will eventually cause the receiving device to return a *Firmware Update Meta Data Status Report Command* with the status code 0x00 (checksum error).

Table 3.91: Firmware Update Meta Data Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_REPORT (0x06)							
Last	Report number 1						
Report number 2							
Data 1							
...							
Data N							
Checksum 1							
Checksum 2							

Last (1 bit)

This field indicates if this *Firmware Update Meta Data Report Command* is the last one.

CC:007A.08.06.11.002 The value 1 MUST indicate that this is the last report. Otherwise the field MUST be set to 0.

CC:007A.08.06.11.003 On the reception of the last image fragment, the receiving node MUST verify that the transferred image does match the indicated Manufacturer ID, Firmware ID and Firmware Target values.

Report number (15 bits)

CC:007A.08.06.11.004 The Report number field indicates the sequence number of the contained firmware fragment. The first firmware fragment MUST be identified by the Report number value 1.

CC:007A.08.06.11.005 The sequence number of each following firmware fragment MUST be incremented. Report number 1 is the most significant byte.

The report number may be used to calculate the offset of the data by the formula:

$$Offset = (Report\ Number - 1) \times Number\ of\ Data\ fields\ (N)$$

CC:007A.08.06.11.006 Except for the last frame, each Report MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field.

Data (N bytes)

The Data field is used to carry one firmware image fragment.

CC:007A.08.06.11.007 Except for the last frame, each *Firmware Update Meta Data Report Command* MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field.

CC:007A.08.06.11.008

A sending device MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

CC:007A.08.06.11.009 A receiving device MUST determine the number of Data bytes from the length of the first received frame.

Checksum (16 bits)

CC:007A.08.06.11.00A The checksum field MUST be used to ensure the consistency of the entire command; including the command class and command identifiers.

CC:007A.08.06.11.00B The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to *CRC-CCITT Source Code*

The Checksum field is known to cause compatibility issues with version 1 devices. Refer to Section 3.2.22.2.1 and Section 3.2.22.2.2

3.2.22.10 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

This command was introduced in version 1.

CC:007A.08.07.11.001 The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

CC:007A.08.07.12.001 A supporting node SHOULD reboot and apply the new firmware image before issuing this command.

CC:007A.08.07.11.002 A node MUST NOT issue Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

Table 3.92: Firmware Update Meta Data Status Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT (0x07)							
Status							
WaitTime 1 (MSB)							
WaitTime 2 (LSB)							

Status (8 bits)

CC:007A.08.07.11.003 This field MUST comply with Table 3.93.

Table 3.93: Firmware Update Meta Data Status Report Command - Status encoding

Status	Description	Version
0x00	The device was unable to receive the requested firmware data without checksum error. Number of retries and request sequence of missing frames are implementation specific. The image MAY be stored for subsequent retries.	1
0x01	The device was unable to receive the requested firmware data. Number of retries and request sequence of missing frames are implementation specific. The image MAY be stored for subsequent retries.	1
0x02	The transferred image does not match the Manufacturer ID. The image is not stored.	4
0x03	The transferred image does not match the Firmware ID. The image is not stored.	4
0x04	The transferred image does not match the Firmware Target. The image is not stored.	4
0x05	Invalid file header information. The image is not stored.	4
0x06	Invalid file header format. The image is not stored.	4
0x07	Insufficient memory. The image is not stored.	4
0x08	The transferred image does not match the Hardware version The image is not stored.	5
0x09..0xFC	*Reserved*	*N/A*
0xFD	Firmware image downloaded successfully, waiting for activation command.	4
0xFE	New image was successfully stored in temporary non-volatile memory. The device does not restart itself. This Status code MUST NOT be used when updating the Z-Wave chip image	3
0xFF	New image was successfully stored in temporary non-volatile memory and/or applied successfully. The supporting node MAY restart itself. In this case, version 3 or newer supporting nodes SHOULD use the WaitTime to indicate that reboot has not been performed yet.	1

CC:007A.08.07.11.004 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

WaitTime (16 bits)

CC:007A.08.07.11.005 The WaitTime field MUST report the time that is needed before the receiving node again becomes available for communication after the transfer of an image. The unit is the second.

CC:007A.08.07.11.006 The value 0 (zero) MUST indicate that the node is ready (it SHOULD have rebooted if needed and applied the Firmware Image).

CC:007A.08.07.11.007 The value 0xFFFF is reserved and MUST NOT be returned.

CC:007A.08.07.12.002 A controlling node receiving this command SHOULD wait for the number of seconds specified in this field before trying to resume communication. When resuming communication, the controlling application SHOULD issue a NOP command and wait for acknowledgement.

CC:007A.08.07.12.003 The controlling application MAY attempt resuming communication repeatedly. In that case, the NOP interval SHOULD be five seconds and MUST be at least one second.

3.2.22.11 Firmware Update Activation Set Command

This command is used to initiate the programming of a previously transferred firmware image.

This command was introduced in version 4.

Refer to the *Firmware Update Meta Data Status Report Command* *Status* field value 0xFD.

CC:007A.08.08.13.001 This command MAY be issued directly by a controlling node or MAY be scheduled for later execution via the Schedule Command Class.

CC:007A.08.08.11.001 The *Firmware Update Activation Status Report Command* MUST be returned in response to this command.

Table 3.94: Firmware Update Activation Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_ACTIVATION_SET (0x08)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Checksum 1							
Firmware Checksum 2							
Firmware Target							
Hardware Version							

For fields' description, refer to *Firmware Data fields*.

3.2.22.12 Firmware Update Activation Status Report Command

This command is used to advertise the result of a firmware update operation initiated by the *Firmware Update Activation Set Command*.

This command was introduced in version 4.

Table 3.95: Firmware Update Activation Status Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_ACTIVATION_STATUS_REPORT (0x09)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Checksum 1							
Firmware Checksum 2							
Firmware Target							
Firmware Update Status							
Hardware Version							

For fields not described below, refer to the *Firmware Data fields*.

Firmware Update Status (8 bits)

CC:007A.08.09.11.001 The Firmware Update Status field MUST comply with Table 3.96.

Table 3.96: Firmware Update Activation Status Report Command - Firmware Update Status encoding

Status	Description	Version
0x00	Invalid combination of manufacturer ID, firmware ID and Hardware Version or Firmware Target. The received image will not be stored.	4
0x01	Error activating the firmware. Last known firmware image has been restored. The received image will not be stored.	4
0x02..0xFE	*Reserved*	*N/A*
0xFF	Firmware update completed successfully.	1

CC:007A.08.09.11.002 Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

3.2.22.13 Firmware Update Meta Data Prepare Get Command

This command is used to request that a firmware download is initiated by the node sending this command.

The purpose of this command is to download and backup the existing firmware from the device before starting firmware upgrade.

This command was introduced in version 5.

CC:007A.08.0A.11.001 The *Firmware Update Meta Data Prepare Report Command* Command MUST be returned in response to this command when the receiving node is ready to send the requested firmware image or to advertise that firmware is not downloadable.

CC:007A.08.0A.11.002 This command MUST NOT be issued via multicast addressing.

CC:007A.08.0A.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

CC:007A.08.0A.12.001 Any image transferred via the *Firmware Update Meta Data Command Class, version 8* SHOULD include a fingerprint value to enable the validation of the integrity of the entire image after the CC:007A.08.0A.11.004 transfer. The image and the fingerprint value MUST be packed in one entity which can be stored in one file and transferred as one entity.

Table 3.97: Firmware Update Meta Data Prepare Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_PREPARE_GET (0x0A)							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Hardware Version							

Refer to *Firmware Data fields* for field description.

Fragment Size (16 bits)

CC:007A.08.0A.11.005 The Fragment Size field MUST report the fragment size that is to be used for firmware fragments. A receiving device MUST use this fragment size for the firmware update. The fragment size is not exchanged during the actual firmware update.

CC:007A.08.0A.11.006 The Fragment Size MUST NOT exceed the Max Fragment Size value of the *Firmware Meta Data Report Command*. A version 1 *Firmware Update Meta Data Request Get Command* does not carry a Fragment Size field. A receiving node MUST determine the number of Data bytes from the length of

the first received frame, the use of security encapsulation as well as the presence of Checksum bytes in the *Firmware Meta Data Report Command*.

3.2.22.14 Firmware Update Meta Data Prepare Report Command

This command is used to advertise if the firmware image has been prepared and is ready to be transferred.

This command was introduced in version 5.

Table 3.98: Firmware Update Meta Data Prepare Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD (0x7A)							
Command = FIRMWARE_UPDATE_MD_PREPARE_REPORT (0x0B)							
Status							
Firmware Checksum 1							
Firmware Checksum 2							

Status (8 bits)

CC:007A.08.0B.11.001 The Firmware Update Status field MUST comply with Table 3.99.

Table 3.99: Firmware Update Meta Data Prepare Report Command - Status encoding

Status	Description	Version
0x00	ERROR. Invalid combination of Manufacturer ID and Firmware ID. The receiving node MUST NOT initiate the firmware download.	5
0x01	ERROR. Device expected an authentication event to enable firmware update. The receiving node MUST NOT initiate the firmware download.	5
0x02	ERROR. The requested Fragment Size exceeds the Max Fragment Size. The receiving node MUST NOT initiate the firmware download.	5
0x03	ERROR. This firmware target is not downloadable. The receiving node MUST NOT initiate the firmware download.	5
0x04	ERROR. Invalid Hardware Version. The receiving node MUST NOT initiate the firmware download.	5
0x05..0xFE	*Reserved*	*N/A*
0xFF	OK. The receiving node can initiate the firmware download of the target specified in the *Firmware Update Meta Data Prepare Get Command*.	5

Firmware Checksum (16 bits)

Refer to *Firmware Checksum (16 bits)*.

3.2.22.15 Firmware Update Examples and frame flows

3.2.22.15.1 Requesting Firmware information

Figure 3.7 shows a controlling node requesting the Firmware capabilities of another node by issuing the *Firmware Meta Data Get Command* .

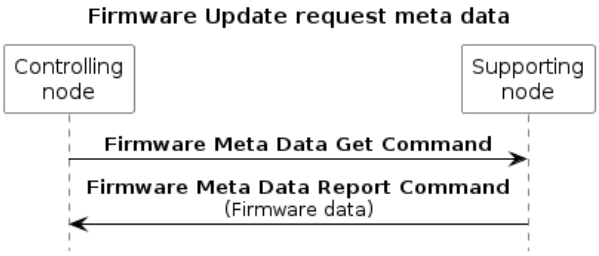


Figure 3.7: Requesting firmware data from a node

3.2.22.15.2 Performing a Firmware update

Figure 3.8 outlines the actual firmware update message flow. Prior to the firmware update, the supporting node may receive an out-of-band authentication (e.g. physical activation of a pushbutton).

The controller sends a *Firmware Update Meta Data Request Get Command* to initiate the downloading a new firmware image.

The supporting node returns a *Firmware Update Meta Data Request Report Command* to confirm the update request and the supporting node begins pulling firmware image fragments from the controlling node.

CC:007A.08.00.12.001 It is RECOMMENDED for controlling nodes to time out after a few minutes if no more *Firmware Update Meta Data Get Command* is received before the end of the transfer. In case of time out, the controlling node SHOULD consider the firmware update as aborted/failed.

Finally a *Firmware Update Meta Data Status Report Command* is returned to the controlling node to indicate the success (or failure) of the update process.

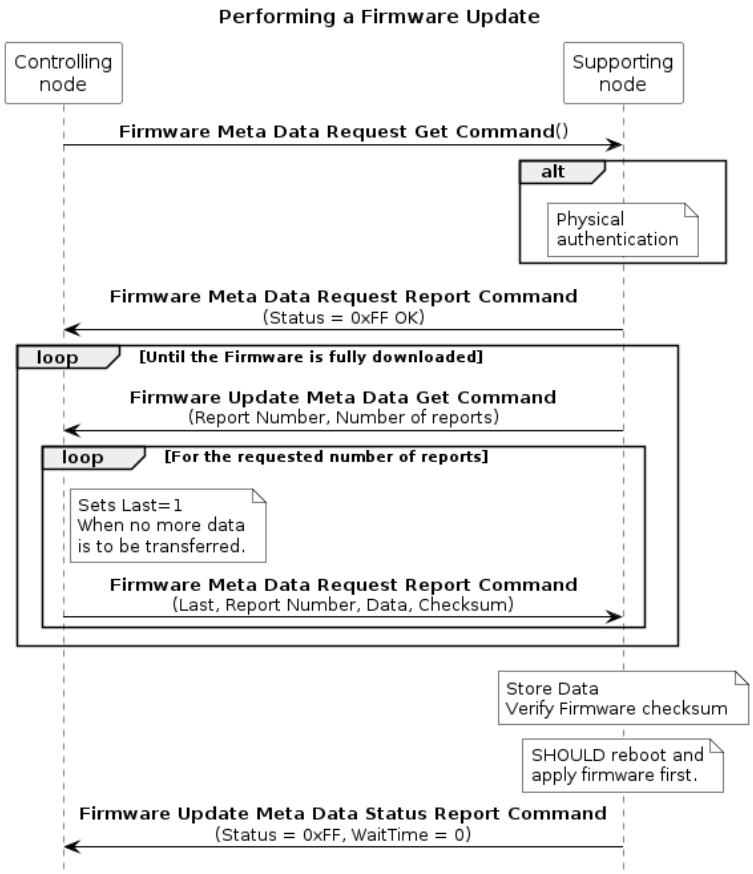


Figure 3.8: Transferring a firmware image to a device

3.2.22.15.3 Identifying Firmware revisions

The different fields Manufacturer ID, Firmware ID and Hardware version are used to identify compatible firmware image with the product. The *Version Command Class, version 1 [OBSOLETED]* is also used for identifying the Firmware version/subversion An example of the different fields' usage for a wall outlet is shown in Table 3.100

Table 3.100: Labelling different Firmware revisions

Product	Firmware identification			
	Manufacturer ID	Hardware Version	Firmware ID	Firmware version/sub-version
Initial Wall outlet	0x0001	0x01	0x0001	0x01 / 0x01
New Revision				
Fixing bugs	0x0001	0x01	0x0001	0x01 / 0x02
New Revision				
New features	0x0001	0x01	0x0001	0x02 / 0x01
New Revision				
US version (as before)	0x0001	0x01	0x0001	0x02 / 0x01
China version	0x0001	0x01	0x0002	0x02 / 0x01
New Revision				
Hardware change (US)	0x0001	0x02	0x0001	0x02 / 0x01
Hardware change (China)	0x0001	0x02	0x0002	0x02 / 0x01
New Revision				
Adding S2 (initial HW, US)	0x0001	0x01	0x0001	0x03 / 0x01
Adding S2 (initial HW, China)	0x0001	0x01	0x0002	0x03 / 0x01
Adding S2 (new HW, US)	0x0001	0x02	0x0001	0x03 / 0x01
Adding S2 (new HW, China)	0x0001	0x02	0x0002	0x03 / 0x01

An illustration of a controller retrieving the firmware information is given in Figure 3.9

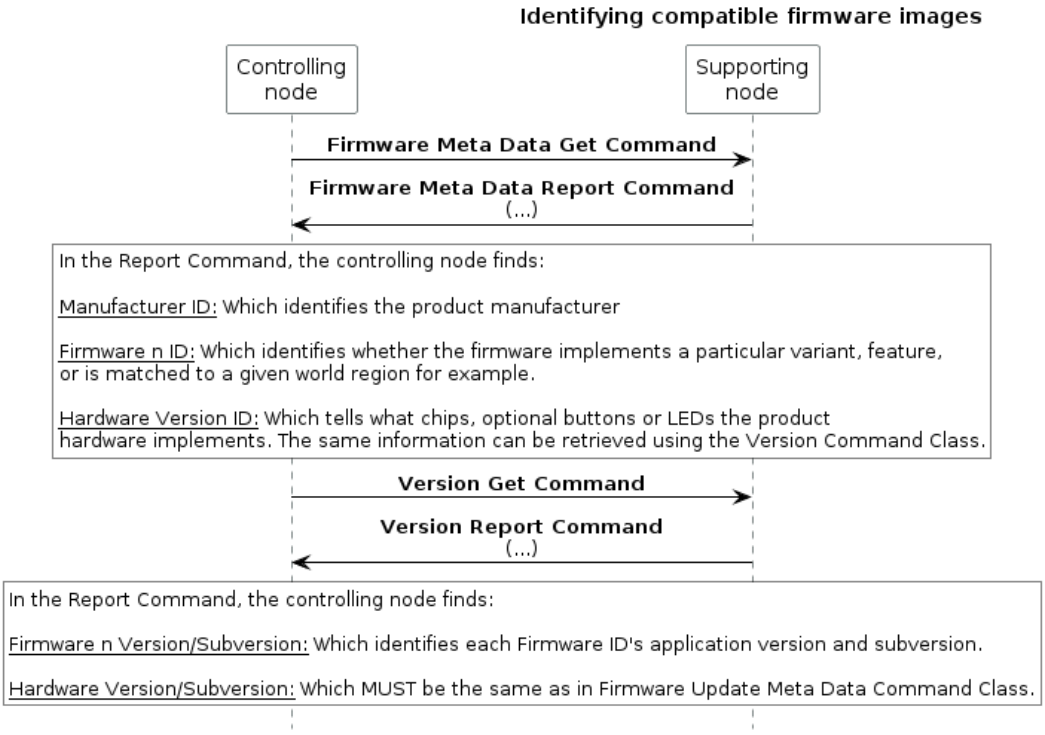


Figure 3.9: Identifying compatible firmware images

3.2.22.15.4 Firmware Activation

A controlling node can instruct a supporting node to wait for activating its firmware after download.

Note: nodes supporting version 6 or older may not wait for the activation command after receiving
CC:007A.08.00.12.002 the *Firmware Update Meta Data Request Get Command* with the *Activation* field set to 1. It is RECOMMENDED to use this functionality with nodes supporting version 7 or newer.

An example of a firmware activation is given in Figure 3.10.

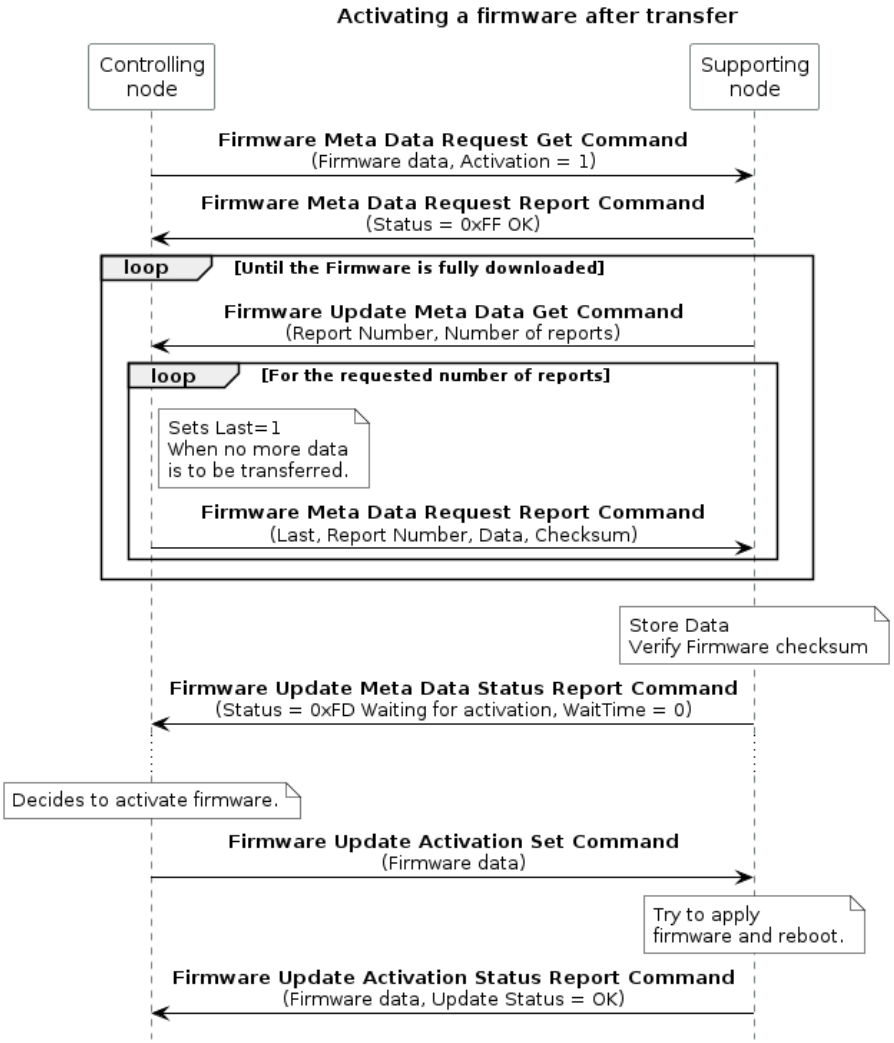


Figure 3.10: Activating a firmware after transfer

3.2.22.15.5 Firmware download

Some supporting nodes can allow to download their firmware for back-up reasons.

Note: Very few supporting nodes are capable of downloading their firmware.

Firmware downloads cannot be performed non-securely and cannot be resumed. An example of a firmware download is given in Figure 3.11.

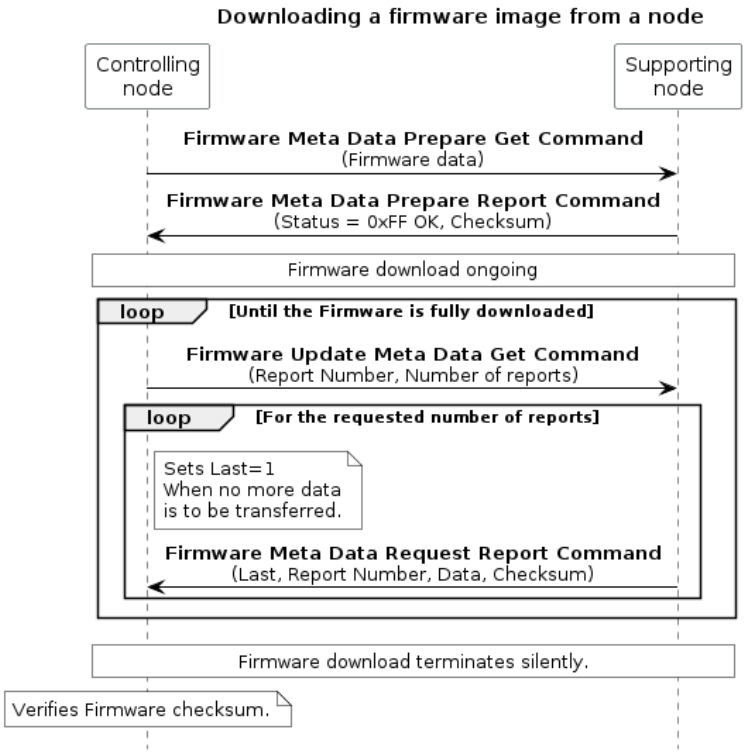


Figure 3.11: Downloading a firmware image from a node

3.2.22.15.6 Firmware Resume

If a previous firmware update failed or was interrupted, the controlling and suppporting nodes can agree on resuming the previous firmware update attempt.

CC:007A.08.00.12.003 The controlling node SHOULD request to resume firmware update if they actively decided to abort a firmware update setting the *Last* field to 1 in a *Firmware Update Meta Data Report Command*.

An example of resuming firmware update is shown in Figure 3.12

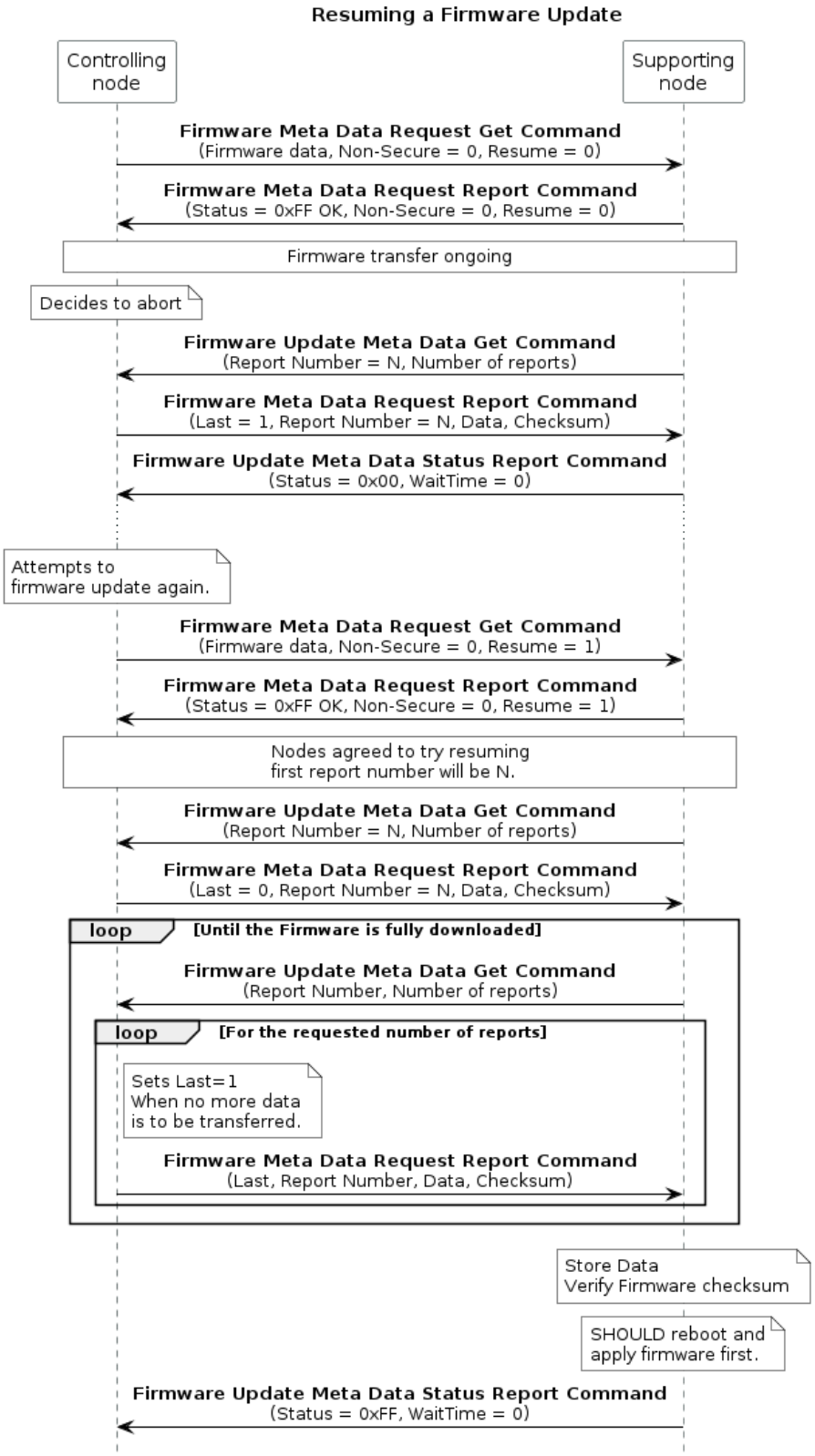


Figure 3.12: Resuming a Firmware Update

CC:007A.08.00.12.004 A supporting node SHOULD accept to resume firmware transfer regardless of the Non-Secure setting, i.e. resuming securely a non-secure transfer or vice-versa SHOULD be allowed.

3.2.22.15.7 Non-secure Firmware update

To increase the throughput of a firmware update, a controlling node and suppporting node can agree on performing the Firmware Transfer without any Security encapsulation.

An illustration of a non-secure firmware update is shown in Figure 3.13

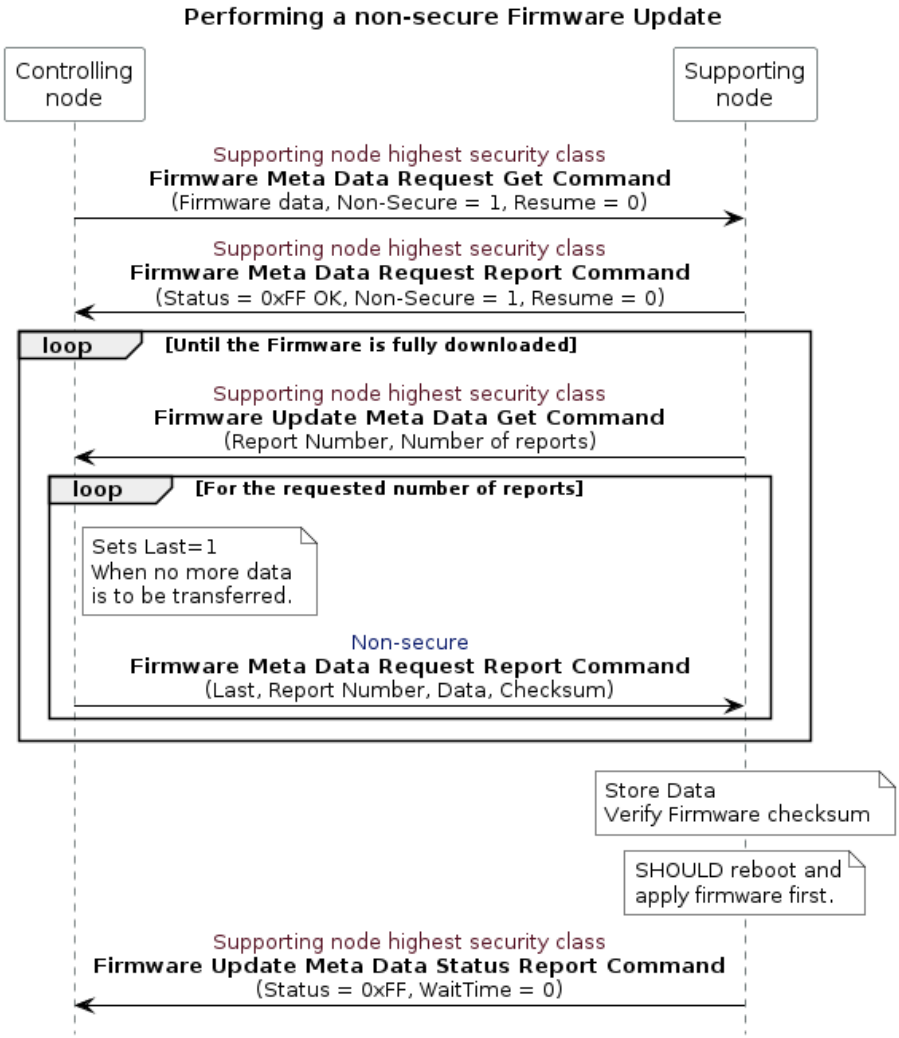


Figure 3.13: Performing a non-secure Firmware Update

3.2.22.15.8 Rejecting firmware updates or downloads

Supporting nodes can reject firmware updates or downloads if they do not support the respective functionality.

An illustration of a non-upgradable firmware is shown in Figure 3.14 and a non-downloadable firmware is shown in Figure 3.15

Figure 3.14: Rejecting a Firmware Update Request for non upgradable firmwares images

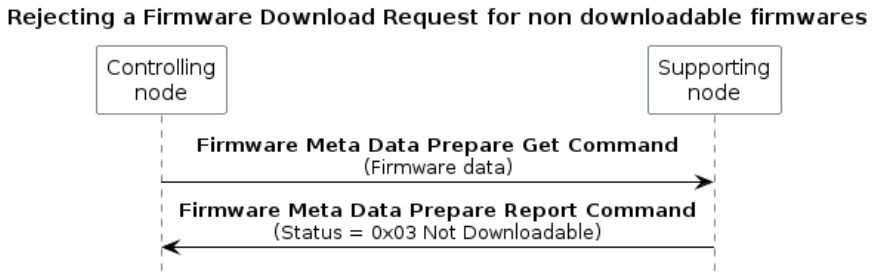


Figure 3.15: Rejecting a Firmware Download Request for non downloadable firmwares

3.2.23 Grouping Name Command Class, version 1 [DEPRECATED]

Warning: THIS COMMAND HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Association Group Information (AGI) Command Class for naming association groups.

If implementing this command class, it is RECOMMENDED that the Association Group Information (AGI) Command Class is also implemented.

The Grouping Name Command Class is used to transfer name of groupings (as defined by the grouping identifier in the Association Command Class).

3.2.23.1 Grouping Name Set Command

The Grouping Name Set Command used to set a grouping Identifier name.

Table 3.101: Grouping Name Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GROUPING_NAME (0x7B)							
Command = GROUPING_NAME_SET (0x01)							
Grouping Identifier							
Reserved					Char. Presentation		
Grouping Name 1							
...							
Grouping Name N							

Grouping Identifier (8 bits)

The field specifies the Grouping ID.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Char. Presentation (3 bits)

The char presentation identifier MAY be set to the following values:

Table 3.102: Grouping Name Set Command::Char Presentation

Char. Presentation	Description
0	Using standard ASCII codes, see *ASCII Codes* (values 128-255 are ignored)
1	Using standard and OEM Extended ASCII codes, see *appendix_ascii_codes*.
2	Unicode UTF-16

Note: Devices supporting Unicode UTF-16 characters can have strings of a maximum of 8 characters because each character is described by a 2 byte long decimal representation. The first byte is the most significant byte. I.e. if there is one Unicode character in the set frame the char 1 will be MSB and char 2 will be LSB of the Unicode character.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Grouping Name (N bytes)

Grouping name using specified character representation.

The Grouping name MAY have a maximum of 16 characters and a minimum of 0 characters. The number of character fields transmitted MUST be determined from the frame length. If a frame with more than 16 characters is received, the receiving node MUST ignore any characters following the 16th character.

3.2.23.2 Grouping Name Get Command

The Grouping Name Get Command is used to request a grouping Identifier name.

The Grouping Name Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.103: Grouping Name Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GROUPING_NAME (0x7B)							
Command = GROUPING_NAME_GET (0x02)							
Grouping Identifier							

Grouping Identifier (8 bits)

The field specifies the grouping identifier.

3.2.23.3 Group Name Report Command

The Grouping Name Report Command is used to report the grouping Identifier name.

Table 3.104: Grouping Name Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GROUPING_NAME (0x7B)							
Command = GROUPING_NAME_REPORT (0x03)							
Grouping Identifier							
Reserved					Char. Presentation		
Grouping Name 1							
...							
Grouping Name N							

Grouping Identifier (8 bits)

The field specifies the grouping identifier.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Char. Presentation (3 bits)

Refer to description under the Grouping Name Set Command

Grouping Name (N bytes)

The Grouping Name fields contain the assigned group name. The Group Name can have a maximum of 16 characters and a minimum of 0 characters.

3.2.24 Hail Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND HAS BEEN OBSOLETED

New implementations MUST report local state updates via the Lifeline Association Group or use dedicated Command Classes for application specific purposes. Refer to [34].

The Hail Command Class used by applications to hail other devices in the Z-Wave network. The usage of the Hail Command Class is application specific.

3.2.24.1 Hail Command

Application can send unsolicited Hail Command to other devices in a Z-Wave network.

Table 3.105: Hail Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HAIL (0x82)							
Command = HAIL (0x01)							

3.2.25 Indicator Command Class, version 1

The Indicator Command Class is used to help end users to monitor the operation or condition of the application provided by a supporting node.

3.2.25.1 Interoperability considerations

Version 1 supporting nodes implement a single indicator resource. The indicator resource can only be turned on or off.

Nodes supporting the Wake-Up Command Class SHOULD keep the indicator in the same state (On/Off) as last instructed by an Indicator Set Command.

3.2.25.2 Indicator Set Command

This command is used to enable or disable the indicator resource.

Table 3.106: Indicator Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_SET (0x01)							
Value							

Value (8 bits)

This field is used to enable or disable the indicator resource.

This field MUST be in the range 0x00..0x63 or 0xFF.

The value 0x00 MUST indicate that the indicator MUST be turned off/disabled.

Values in the range 0x01..0x63 MUST indicate that the indicator MUST be turned on/enabled.

The value 0xFF MUST indicate that the indicator MUST be turned on/enabled.

3.2.25.3 Indicator Get Command

This command is used to request the state of the indicator resource.

The Indicator Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.107: Indicator Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_GET (0x02)							

3.2.25.4 Indicator Report Command

This command is used to advertise the current state of the indicator resource.

Table 3.108: Indicator Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_REPORT (0x03)							
Value							

Value (8 bits)

This field is used to advertise the current state of the indicator resource. This field MUST be in the range 0x00..0x63 or set to 0xFF.

The value 0x00 MUST indicate that the indicator is turned off/disabled.

Values in the range 0x01..0x63 MUST indicate that the indicator is turned on/enabled.

The value 0xFF MUST indicate that the indicator is turned on/enabled.

3.2.26 Indicator Command Class, version 2

The Indicator Command Class, version 2 is used to manipulate indicator resources in a supporting node. An indicator may be an LED, an LCD display or a buzzer.

3.2.26.1 Compatibility Considerations

The Indicator Command Class, version 1 provides a single unspecified indicator resource. Version 2 introduces support for multiple indicator resources. Each indicator resource may implement a number of properties (or capabilities) such as turning on/off, set on a specific level or toggling state.

A controlling node may discover the supported indicators ID and their property IDs via the Indicator Supported Report Command.

The Indicator Command Class, version 2 renames the Value field of version 1 to "Indicator 0 Value".

The Indicator 0 Value field properties are unspecified as in version 1.

A version 2 supporting node MUST map the Indicator 0 to one of its supported Indicator ID and Property ID. The Indicator 0 SHOULD be mapped to an indicator 0x01 (Multilevel) or 0x02 (Binary) Property ID.

A supporting node advertising no support for the "Low power" indication Property ID and supporting the Wake Up Command Class MUST keep awake as long as one of the property ID is not 0x00 (off) for the actual Indicator ID. Such a node MUST return to sleep after both the Wake Up No More Information Command is received and all Property IDs are back to 0x00 for the actual Indicator ID.

A supporting node advertising support for the "Low power" indication Property ID and supporting the Wake Up Command Class MAY return to sleep even if an indication is ongoing for the actual Indicator ID.

3.2.26.2 Interoperability considerations

A supporting device MAY use its indicator resources for local status reporting.

After receiving an Indicator Set Command, a supporting node MUST NOT use its indicator resources for local status reporting until it has completed the operation requested by the Set Command.

An exception to this MAY be interface feedback functionality like a short beep on each button press. Likewise, transmission failure to a controlling node MAY be advertised via the use of local indications. As an example, an entry control keypad device may feature an LCD backlight. If the backlight is advertised as an indicator resource, a central control application also has to control the backlight of the keypad, e.g. in response to local user activity and to temporarily draw attention to all keypads when an alarm is enabled.

3.2.26.3 Indicator Set Command

This command is used to manipulate one or more indicator resources at a supporting node.

Table 3.109: Indicator Set Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_SET (0x01)							
Indicator 0 Value (Indicator ID 0 = 0x00, Property ID 0 = 0x01)							
Reserved			Indicator Object Count				
Indicator ID 1							
Property ID 1							
Value 1							
...							
Indicator ID N							
Property ID N							
Value N							

Indicator 0 Value (8 bits)

This field provides backwards compatibility for version 1 supporting nodes.

A receiving node MUST ignore this field if the *Indicator Object Count* field is not set to 0.

A receiving node MUST map this value to a supported indicator resource if the *Indicator Object Count* field is set to 0 or not included in the command.

Refer to Section 3.2.26.1 Compatibility Considerations.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Indicator Object Count (5 bits)

This field is used to advertise the number of indicator objects carried in the actual command. An indicator object MUST comprise an *Indicator ID*, a *Property ID* and a *Value* field.

Indicator ID (8 bits, N times)

This field is used to identify the actual indicator resource. Indicator IDs values are defined in [24].

A receiving node MUST ignore the entire indicator object if an unsupported Indicator ID is specified.

A receiving node MUST NOT ignore other indicator objects included in the command if an unsupported Indicator ID is specified.

Property ID (8 bits, N times)

This field is used to identify the specific property of the indicator resource identified by the corresponding *Indicator ID* field.

A receiving node MUST ignore the entire indicator object if an unsupported Property ID for the corresponding Indicator ID is specified. The receiving node MUST NOT ignore other indicator objects included in the command.

If an *Indicator ID* is specified in an object but not all *Property IDs* are included in the command, a supporting node MUST assume non-specified Property IDs values to be set to 0x00.

If several Property IDs not belonging to the same Property group defined in [24] are specified in the command, the Property IDs from one Property group MUST be applied and all other Property IDs MUST be ignored. For example, if Multilevel and Binary property IDs are defined, only one Property ID value MUST be applied and the other MUST be ignored.

Properties marked as "ADVERTISE ONLY:" MUST be ignored if received in a controlling command.

Value (8 bits, N times)

This field is used to specify the value to assign to the specific indicator property identified by the *Indicator ID* and the *Property ID* fields.

This field MUST be set according to the defined values and requirements for Property IDs in [24].

3.2.26.4 Indicator Get Command

This command is used to request the state of an indicator.

The Indicator Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.110: Indicator Get Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_GET (0x02)							
Indicator ID							

Indicator ID (8 bits)

This field is used to specify the actual indicator resource.

If an unsupported Indicator ID is specified in this command, a receiving node MUST return an indicator object with the specified *Indicator ID* and the *Property ID* and Value fields set to 0x00.

3.2.26.5 Indicator Report Command

This command is used to advertise the state of an indicator resource.

Table 3.111: Indicator Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_REPORT (0x03)							
Indicator 0 Value (Indicator ID 0 = 0x00, Property ID 0 = 0x01)							
Reserved			Indicator Object Count				
Indicator ID 1							
Property ID 1							
Value 1							
...							
Indicator ID N							
Property ID N							
Value N							

Indicator 0 Value (8 bits)

This field provides backwards compatibility for version 1 controlling nodes.

A sending node MUST advertise the current value of the mapped Indicator ID and Property ID.

A version 2 controlling node MUST ignore the Indicator 0 Value field if other values are advertised. Refer to Section 3.2.26.1 Compatibility Considerations.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Indicator Object Count (5 bits)

This field is used to advertise the number of indicator objects carried in this command. An indicator object MUST comprise an *Indicator ID*, a *Property ID* and a *Value* field.

Indicator ID (N bytes)

This field is used to identify the actual indicator resource.

All indicator objects MUST carry the same Indicator ID.

Property ID (N bytes)

This field is used to identify the specific property of the indicator resource identified by the corresponding *Indicator ID* field.

If not all Property IDs are included in the command for the actual Indicator ID, a controlling node MUST assume non-specified Property IDs values to be 0x00.

Value (N bytes) This field is used to specify the current value of the specific indicator property identified by the *Indicator ID* and the *Property ID* fields.

This field MUST be set according to the defined values for Property IDs in [24].

3.2.26.6 Indicator Supported Get Command

This command is used to request the supported properties of an indicator.

The Indicator Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.112: Indicator Supported Get Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_SUPPORTED_GET (0x04)							
Indicator ID							

Indicator ID (1 byte)

This field is used to specify the actual indicator resource.

A controlling node SHOULD set this field to zero to discover the supported Indicator IDs. A supporting node receiving this field set to 0x00 MUST advertise the first supported Indicator ID in response.

A supporting node receiving a non-zero Indicator ID that is not supported MUST set all fields (*Indicator ID*, *Next Indicator ID*, *Property Supported Bit Mask Length*) to 0x00 in the returned response.

3.2.26.7 Indicator Supported Report Command

This command is used to advertise the supported properties for a given indicator.

Table 3.113: Indicator Supported Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_SUPPORTED_REPORT (0x05)							
Indicator ID							
Next Indicator ID							
Reserved			Property Supported Bit Mask Length				
Property Supported Bit Mask 1							
...							
Property Supported Bit Mask N							

Indicator ID (1 byte)

This field is used to specify the actual indicator ID resource for the supported properties are being advertised.

The supported Indicator IDs MUST be according to the Indicator IDs values defined in [24].

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Next Indicator ID (8 bits)

This field is used to advertise if more Indicator IDs are supported after the actual Indicator ID advertised by the *Indicator ID* field.

If a sending node supports additional Indicator IDs, this field MUST advertise the next supported Indicator ID.

If the sending node does not support additional Indicator IDs this field MUST be set to 0x00.

The supported Indicator IDs MUST be according to the Indicator IDs values defined in [24].

Property Supported Bit Mask Length (5 bits)

This field is used to advertise the length of the *Property Supported Bit Mask* field in bytes.

The value MUST be in the range 0..32.

Property Supported Bit Mask (N bytes)

This field is used to advertise the properties supported by the actual Indicator ID.

The length of this field in bytes MUST be according to the *Property Supported Bit Mask Length* field value. This field MUST be encoded as a bitmask as follow:

- Bit 0 in Bit Mask 1 is not allocated to any property and MUST be set to zero.
- Bit 1 in Bit Mask 1 MUST indicate if Property ID = 1 (Multilevel) is supported.
- Bit 2 in Bit Mask 1 MUST indicate if Property ID = 2 (On/Off) is supported.
- ...

If the Property ID is supported, the corresponding bit MUST be set to 1.

If the Property ID is not supported the corresponding bit MUST be set to 0.

The supported Property IDs MUST be according to the Property IDs values defined in [24].

3.2.27 Indicator Command Class, version 3

The Indicator Command Class, version 3 is used to manipulate indicator resources in a supporting node. An indicator may be an LED, an LCD display or a buzzer.

3.2.27.1 Compatibility Considerations

The Indicator Command Class, version 3 is backwards compatible with the Indicator Command Class, version 2. All commands and fields not mentioned in this version MUST remain unchanged from the Indicator Command Class, version 2.

This version introduces new Indicator IDs and Property IDs. The list of supported Indicator IDs and Property IDs is moved to [24]. Values not defined in [24] are reserved and MUST NOT be used by a supporting node.

New values MAY be added to [24] and will be labelled under version 3.

3.2.28 Indicator Command Class, version 4

3.2.28.1 Compatibility Considerations

The Indicator Command Class, version 4 is backwards compatible with the Indicator Command Class, version 3. All commands and fields not mentioned in this version MUST remain unchanged from the Indicator Command Class, version 3.

This version introduces 2 new commands used to request the detailed information, appearance and use of manufacturer defined Indicator IDs.

- Indicator Description Get Command
- Indicator Description Report Command

Supporting nodes supporting Indicator IDs in the range 0x80..0x9F (manufacturer defined indicators) MUST have a description available that can be requested via the version 4 commands.

Indicator IDs in the range 0x80..0x9F that are supported by a node MUST NOT replace or be achievable by any combination of Indicator IDs defined in [24].

3.2.28.2 Indicator Description Get Command

This command is used to request a detailed description of the appearance and use of an Indicator ID

The Indicator Description Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.114: Indicator Description Get Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_DESCRIPTION_GET (0x06)							
Indicator ID							

Indicator ID (8 bits)

This field is used to specify the actual indicator resource.

A supporting node MUST return a report for the Indicator ID value specified in this field.

This field MUST be in the range 0x80..0x9F.

A supporting node receiving this command for an Indicator ID outside the 0x80..0x9F range MUST return an Indicator Description Report with the Description Length set to 0.

3.2.28.3 Indicator Description Report Command

This command is used to advertise appearance and use of an indicator ID resource.

Table 3.115: Indicator Description Report Command, version 4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR (0x87)							
Command = INDICATOR_DESCRIPTION_REPORT (0x07)							
Indicator ID 1							
Description Length							
Description 1							
...							
Description N							

Indicator ID (8 bits)

This field is used to specify the actual indicator resource for which the description is being advertised.

Description Length (1 byte)

This field indicates the length in bytes of the *Description* field.

The value 0 MUST indicate that no description is available for the Indicator ID.

Description (N bytes)

This field is used to advertise the appearance and use of the Indicator ID.

It MUST replace the "Appearance and use" column information from [24] for manufacturer defined Indicator IDs.

The length of this field in bytes MUST comply with the advertised value in the *Description Length* field.

This field MUST be omitted if the *Description Length* field is set to 0.

This field MUST be encoded in UTF-8 format.

3.2.29 IP Association Command Class, version 1

The IP Association Command Class is used to create and maintain Z-Wave application bindings between IPv6 hosts. An association group sends a predefined command to the configured destinations when triggered by an event.

IP Association identifies nodes via IPv6 addresses.

IP Association is natively Z-Wave Multi Channel End Point aware.

IP Association groups are specific to a particular End Point or Root Device.

3.2.29.1 Compatibility considerations

The IP Association Command Class extends the functionality of the Z-Wave Multi Channel Association Command Class. Z-Wave nodes may be represented as Z/IP nodes in an IP subnet. IP Associations may be mapped to classic Z-Wave Multi Channel associations.

The IP Association Remove Command extends the Multi Channel Association Remove Command by adding the ability to clear all groups in all End Points of a node. Only one IP association can be created or maintained with a single command.

In order to respond quickly and to conserve battery, it is RECOMMENDED that a Z/IP Gateway caches IP associations and corresponding Z-Wave associations and only looks up all associations when explicitly instructed to do so.

3.2.29.2 Terminology

The IP Association Command Class operates on IP primitives but works for all generations of Z-Wave technology. In the following, the term "Classic Z-Wave" refers to operations that are possible with Z-Wave only releases.

The term "Z-Wave node" is used to identify a Z-Wave node as used in Classic Z-Wave while the term "Z/IP node" denotes an IP enabled Z-Wave node that implements the ICMP echo service and the Z-Wave UDP service. A Z/IP node may implement other IP services. Such IP services are out of scope of this document.

An IP association is created by sending an IP Association Set command from a configuration tool to the association source, instructing the association source to add an association destination to an association group maintained in the association source. The association destination has no knowledge that an association is created.

A Z/IP Gateway represents classic Z-Wave nodes as Z/IP nodes in an IP environment. The Z/IP Gateway therefore maps IP association commands from Z/IP clients to relevant Association and Multi Channel Association commands for Z-Wave nodes.

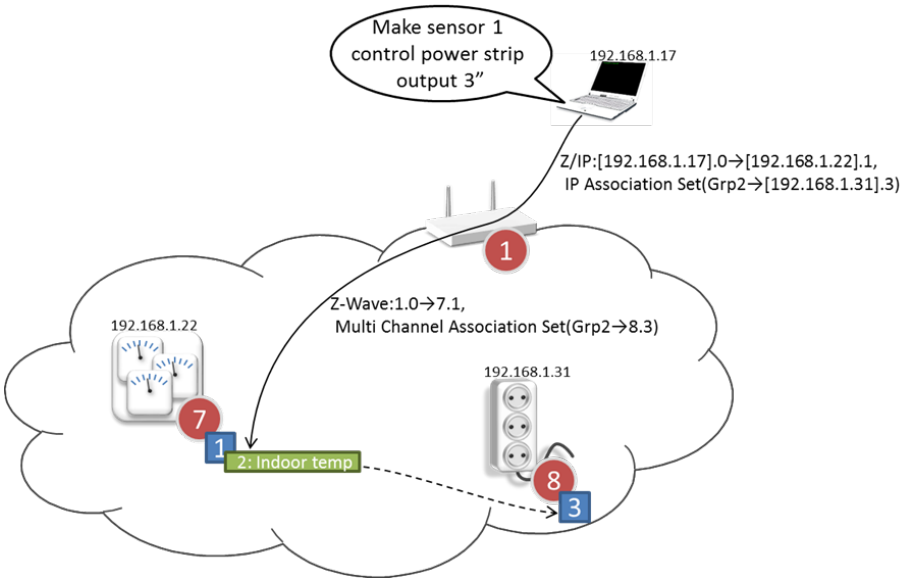


Figure 3.16: IP Association Example

3.2.29.3 Z-Wave Multi Channel compatibility considerations

IP Association is natively Multi Channel aware. The same message format is used to create an association to a logical End Point as to the Root Device.

In the Z/IP framework, End Point 0 represents the physical entity.

IP Associations are conceptually created between two Z/IP resources identified by an IPv6 address and an End Point. In case one or both parties are Z-Wave nodes represented as Z/IP nodes, the Z/IP Gateway MUST create Z-Wave associations in the following way, based on the Resource Directory information pertaining to the actual nodes.

A Z/IP gateway MUST map IP associations to Z-Wave associations according to Table 3.116.

Table 3.116: IP Association Mapping to Z-Wave Association

Association Source Multi Channel Capability	Association Destination Multi Channel Capability	Association Type	Creating the Association
✗	✗	Root Device -> Root Device	An Association Set Command MUST be sent un-encapsulated to the association source node
✓	✓	End Point m -> End Point n m, n MAY be 0	A Multi Channel Association Set Command MUST be sent Multi Channel encapsulated to the association source End Point. A Multi Channel Association from the Lifeline association group of a sensor Root Device enables the transmission of sensor reports with different Source End Points to the Lifeline destination.
✓	✗	End Point m -> Root Device	An Association Set Command MUST be sent Multi Channel encapsulated to the association source End Point.
✗	✓	Root Device -> End Point n	An Association Set Command MUST be sent un-encapsulated to the association source node. This first association MUST target a NodeID owned by the Z/IP Gateway. A second association MUST be created from the abovementioned Z/IP Gateway to the Multi Channel End Point specified in the IP Association Set Command.

3.2.29.4 Advertising the IP Association Command Class

The Node Information Frame emitted by Classic Z-Wave nodes does not include IP Association. To allow IP clients to issue IP Association commands, the Z/IP Gateway MUST rewrite the list of supported command classes from the node, replacing the Association and Multi Channel Association Command Classes with the IP Association Command Class in the following situations:

- When a Node Info Cached Report is sent to an IP client.
- When a Node Add Status is being sent to an IP client.

In both cases, rewriting MUST NOT be performed if the report is being sent to a native Z-Wave node.

Furthermore, mDNS responses to queries for nodes supporting the IP Association Command Class MUST also advertise nodes supporting the Association and Multi Channel Association Command Classes (but the command classes must be replaced with COMMAND_CLASS_IP_ASSOCIATION before advertised via mDNS).

3.2.29.5 IP Association Set Command

This command is used to add one resource to a given association group.

The receiving IPv6 host SHOULD add the specified destination resource to the specified association group.

This command MAY be ignored if the association group is already full.

Unless the association destination is a gateway, a controlling node SHOULD NOT create an association if the association destination node does not support the controlling commands (Set/Get type) that the actual association group will be sending. The AGI Command Class SHOULD be used to probe the commands that a given association group will be sending.

A controlling node SHOULD NOT create an association if the source and destination nodes are bootstrapped with different security levels.

Table 3.117: IP Association Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION (0x5C)							
Command = IP_ASSOCIATION_SET (0x01)							
Grouping Identifier							
IPv6 Address 1							
...							
IPv6 Address 16							
End Point							

Grouping identifier (8 bits)

This field is used to specify the actual association group.

Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

IPv6 Address (16 bytes)

This field specifies the full IPv6 address for the association destination. The IPv6 address MUST NOT be compressed.

The IPv6 address SHOULD be in the ULA IPv6 prefix or in a globally routable IPv6 prefix.

The IPv6 address MAY be an IPv4-mapped IPv6 address.

The field MUST NOT carry a link-local IPv6 address.

End Point (8 bits)

This field is used in combination with the IPv6 Address to identify the actual resource.

The field represents the Z/IP Bit Address flag (bit 7) as well as the Z/IP 7-bit End Point identifier (bits 6..0).

The receiving node MUST treat this field as one scalar value when adding or removing associations; i.e. any 8-bit End Point value causes one association to be added.

3.2.29.6 IP Association Get Command

This command is used to request active associations for a given association group. The IP Association Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.118: IP Association Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION (0x5C)							
Command = IP_ASSOCIATION_GET (0x02)							
Grouping Identifier							
Index							

Grouping identifier (8 bits)

This field is used to specify the actual association group.

Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

If an unsupported Grouping Identifier is specified, the IP Association Report Command returned in response to this command MUST carry IPv6 address and End Point fields which are set to all-zeros.

Index (8 bits)

This field is used to specify an entry in the association table for the group identified by the Grouping Identifier.

A requesting host SHOULD start specifying the index value 1.

The actual number of nodes in the association group may be determined from the Actual Nodes field of the IP Association Report.

3.2.29.7 IP Association Report Command

This command is used to advertise one association group entry.

Table 3.119: IP Association Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION (0x5C)							
Command = IP_ASSOCIATION_REPORT (0x03)							
Grouping Identifier							
Index							
Actual Nodes							
IPv6 Address 1							
...							
IPv6 Address 16							
End Point							

Grouping identifier (8 bits)

This field is used to identify the actual group of nodes.

Index (8 bits)

This field is used to advertise the actual entry in the association group identified by the Grouping Identifier.

Actual Nodes (8 bits)

This value indicates the number of nodes in the association group advertised by the Grouping Identifier field.

IPv6 Address (16 bytes)

This field carries a full IPv6 address with no compression. The address SHOULD be in the ULA IPv6 prefix or in a globally routable IPv6 prefix. The address MAY be an IPv4-mapped IPv6 address. The field MUST NOT carry a link-local IPv6 address.

If the Actual Nodes value is zero, the IPv6 Address field MUST be all zeros.

End Point (8 bits)

This field is used in combination with the IPv6 Address to identify the actual resource

If the Actual Nodes value is zero, the End Point field MUST be all zeros.

The field represents the Z/IP Bit Address flag (bit 7) as well as the Z/IP 7-bit End Point identifier (bits 6..0).

3.2.29.8 IP Association Remove Command

This command is used to remove one resource from a given association group.

Table 3.120: IP Association Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION (0x5C)							
Command = IP_ASSOCIATION_REMOVE (0x04)							
Grouping Identifier							
IPv6 Address 1							
...							
IPv6 Address 16							
End Point							

Grouping identifier (8 bits)

This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

A receiving node MUST ignore an unsupported Grouping Identifier.

This field MUST be interpreted in combination with the IPv6 Address and End Point fields.

IPv6 Address (16 bytes)

End Point (1 byte)

These fields specify the resources that are to be removed.

The Grouping identifier and these fields MUST be interpreted as follows.

Table 3.121: IP Association Remove, V1::Parameter Interpretation

Command Properties			
Grouping Identifier	**IPv6 Address**	**End Point ID**	**Removal Operation**
> 0	<> 0	= 0 (Root Device)	Remove destination from association group in the association source Root Device
> 0	<> 0	> 0 (End Point)	Remove destination from association group in the specified association source End Point
> 0	= 0	= 0 (Root Device)	Remove all destinations from association group in the association source Root Device
> 0	= 0	> 0 (End Point)	Remove all destinations from association group in the specified association source End Point
= 0	= 0	= 0 (Root Device)	Remove all destinations from all association groups in the association source Root Device **and** in all association source End Points
= 0	= 0	> 0 (End Point)	Remove all destinations from all groups in the specified association source End Point
= 0	<> 0	>= 0	*Reserved*

*) The End Point field represents the Z/IP Bit Address flag (bit 7) as well as the Z/IP 7-bit End Point identifier (bits 6..0).

The receiving node MUST treat the End Point field as one scalar value when removing associations; i.e. any 8-bit End Point value causes one association to be removed.

3.2.30 Manufacturer Specific Command Class, version 1

The Manufacturer Specific Command Class is used to advertise manufacturer specific information. Version 2 of this command class further allows device specific information to be advertised.

3.2.30.1 Security Considerations

This Command Class provides information about the device implementation. This information can potentially be used by an attacker to find vulnerabilities.

CC:0072.01.00.41.004 In order to increase interoperability with legacy controlling nodes, a node supporting the Security 0 Command Class MUST reply to Manufacturer Specific Get Commands received non-securely if it was granted the S0 network key as its highest Security Class.

CC:0072.01.00.43.001 In this case, it is OPTIONAL for the node to advertise the Manufacturer Specific Command Class in its NIF.

3.2.30.2 Manufacturer Specific Get Command

This command is used to request manufacturer specific information from another node.

CC:0072.01.04.11.001 The Manufacturer Specific Report Command MUST be returned in response to this command.

CC:0072.01.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0072.01.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.122: Manufacturer Specific Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC (0x72)							
Command = MANUFACTURER_SPECIFIC_GET (0x04)							

3.2.30.3 Manufacturer Specific Report Command

This command is used to advertise manufacturer specific device information.

Table 3.123: Manufacturer Specific Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC (0x72)							
Command = MANUFACTURER_SPECIFIC_REPORT (0x05)							
Manufacturer ID 1							
Manufacturer ID 2							
Product Type ID 1							
Product Type ID 2							
Product ID 1							
Product ID 2							

Manufacturer ID (16 bits)

CC:0072.01.05.11.001 The Manufacturer ID field MUST carry the unique ID identifying the manufacturer of the device.

Manufacturer identifiers can be found in [32]. The first byte is the most significant byte.

Product Type ID (16 bits)

CC:0072.01.05.11.002 The Product Type ID field MUST carry a unique ID identifying the actual product type.

The first byte is the most significant byte.

CC:0072.01.05.11.003 A specific Product Type ID MUST be defined by the manufacturer for each type of product.

Product ID (16 bits)

CC:0072.01.05.11.004 The Product ID field MUST carry a unique ID identifying the actual product.

The first byte is the most significant byte.

CC:0072.01.05.11.005 A specific Product ID MUST be defined by the manufacturer for each product of a given product type. Thus, the same Product ID value may appear for different Product Type ID values.

3.2.31 Manufacturer Specific Command Class, version 2

Manufacturer Specific Command Class, version 2 adds a set of commands to communicate unique identification, e.g. the serial number, of the product.

Commands not mentioned here remains unchanged as specified for Manufacturer Specific Command Class, Version 1.

3.2.31.1 Security Considerations

A supporting node MUST comply with Section 3.2.30.1 Security Considerations

3.2.31.2 Device Specific Get Command

This command is used to request device specific information.

CC:0072.02.06.11.001 The Device Specific Report Command MUST be returned in response to this command.

CC:0072.02.06.11.002 This command MUST NOT be issued via multicast addressing.

CC:0072.02.06.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.124: Device Specific Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC (0x72)							
Command = DEVICE_SPECIFIC_GET (0x06)							
Reserved					Device ID Type		

Device ID Type (3 bits)

This field contains values for the Device ID Type.

Table 3.125: Device ID Type (3 bits)

Device ID Type	Value
Return OEM factory default Device ID Type	0
Serial Number	1
Pseudo Random	2
Reserved	3-7

CC:0072.02.06.12.001 A sending node SHOULD specify a value of zero when issuing the Device Specific Get command since the responding node may only be able to return one Device ID Type.

3.2.31.3 Device Specific Report Command

This command is used to advertise device specific information.

Table 3.126: Device Specific Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC (0x72)							
Command = DEVICE_SPECIFIC_REPORT (0x07)							
Reserved					Device ID Type		
Device ID Format			Device ID Length				
Device ID Data 1							
...							
Device ID Data N							

Device ID Data Format (3 bits)

This command field defines the format used for Device ID Data.

Table 3.127: Device ID Data Format (3 bits)

Device ID Data Format	Value	Description
CC:0072.02.07.11.001 UTF-8	0x00	The Device ID Data MUST be in UTF-8 format.
CC:0072.02.07.11.002 Binary	0x01	The Device ID Data is in plain binary format and MUST be displayed as hexadecimal values e.g. 0x30, 0x31, 0x32, 0x33 MUST be displayed as h'30313233.

CC:0072.02.07.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Device ID Type (3 bits)

This field contains values for the Device ID Type. See Device Specific Get Command for details.

CC:0072.02.07.13.001 In case the Device ID Type specified in a Device Specific Get command is not supported by the responding node, the responding node MAY return the factory default Device ID Type (as if receiving the value 0 in the Device Specific Get command).

Device ID Data Length (5 bits)

CC:0072.02.07.11.004 This field contains the length of the Device ID Data field. The field MUST NOT carry the value zero.

Device ID Data (N bytes)

Data fields for Device ID. "Device ID Data Format" defines data format and "Device ID Data Length Indicator" defines length.

CC:0072.02.07.11.005 The Manufacturer ID and Device ID combination MUST be globally unique.

3.2.32 Multi Instance Association Command Class, version 1 [OBSOLETED]

Warning: THIS VERSION HAS BEEN OBSOLETED

CC:008E.01.00.11.001 New implementations MUST use the Multi Channel Association Command Class, version 2 or newer.

3.2.33 Multi Channel Association Command Class, version 2 [OBSOLETED]

> **Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED**
>
> New implementations MUST NOT support this Command Class.
>
> New implemenetations MUST use the *Association Command Class, version 3*, or newer instead.

The Multi Channel Association Command Class is used to manage associations to Multi Channel End Point destinations as well as to NodeID destinations.

An association group sends an unsolicited command to the configured destinations when triggered by an event. The parameters of the command may be dynamic, e.g. the temperature of a sensor reading or the light level for a dimmer.

A NodeID association identifies its destination by a NodeID.

An End Point association identifies its destination by a combination of a NodeID and an End Point.

Refer to Section 4 for an introduction to the Multi Channel concept.

3.2.33.1 Compatibility Considerations

CC:008E.02.00.21.001 The Multi Channel Association Command Class extends the functionality of the Association command class. A device supporting this Command Class version MUST also support the (non-Multi Channel) Association Command Class, version 2.

CC:008E.02.00.22.001 A controlling device SHOULD NOT create End Point associations to dynamic Multi Channel End Points.

CC:008E.02.00.22.002 The Association Group Information (AGI) Command Class SHOULD be supported to enable automated discovery of association group properties.

CC:008E.02.00.21.002 The Association and Multi Channel Association Command Classes MUST access the same collection of association groups. Further, the following applies:

CC:008E.02.00.21.003 The two command classes MUST advertise the total number of association groups.

CC:008E.02.00.21.004 The two command classes MUST advertise the total number of supported nodes for a given association group.

CC:008E.02.00.21.005 Any advertised association group MUST support NodeID destinations as well as End Point destinations.

CC:008E.02.00.21.006 NodeID associations maintained via the Multi Channel Association Command Class MUST be identical to NodeID associations maintained via the Association Command Class.

CC:008E.02.00.21.007 CC:008E.02.00.23.001 The Multi Channel Command Class specifies that, for backwards compatibility with non-Multi Channel devices, the Root Device of a Multi Channel device MUST mirror the functionality of End Point 1 and it MAY mirror the functionality of more End Points. This principle also applies to association groups advertised by the Root Device.

CC:008E.02.00.22.003 Each association group advertised by the Root Device, except for association group 1, SHOULD mirror an association group of an End Point.

CC:008E.02.00.22.004 If a Root Device association group mirrors an End Point association group, the Root Device SHOULD map all Association commands for that group to the mirrored End Point association group.

CC:008E.02.00.22.005 Except for association group 1 (the Z-Wave Plus Lifeline), a Multi Channel aware controlling device SHOULD ignore all Root Device association groups since they are just mirrored End Point association groups.

CC:008E.02.00.21.008 The use of encapsulation MUST comply with Table 3.128.

Table 3.128: V2 Associations and the Transmissions they may Trigger

Source -> Destination	Association Type	Allowed Transmissions
NodeID -> NodeID (V2)	NodeID	Non-encapsulated
End Point -> NodeID (V2)	NodeID	Non-encapsulated
NodeID -> End Point (V2)	End Point	Encapsulated *) (src 0, dst>0)
End Point -> End Point (V2)	End Point	Encapsulated (src>0, dst>0)
REMOVED NodeID -> Root Device	End Point	*None* **)

*) The source End Point MAY be different than 0 if the associated group is an End Point group mapped to the Root Device

**) Version 2 of this command class does not allow the destination End Point to be zero.

3.2.33.2 Z-Wave Plus Considerations

The Z-Wave Plus certification program mandates that Association group 1 is reserved for the Lifeline
CC:008E.02.00.11.001 association group. Group 1 MUST NOT be assigned to any other use than the Lifeline group. The actual Device Type specifies a mandatory list of commands which the device must be able to send to
CC:008E.02.00.13.001 all lifeline group destinations. A manufacturer MAY add additional commands to the lifeline group.

CC:008E.02.00.12.001 End Points SHOULD NOT implement the Lifeline Association Group. End Points SHOULD report that zero NodeIDs are supported for association group 1.

The Z-Wave Plus certification program mandates support for the Association Group Information (AGI) Command Class if a device supports the Multi Channel Association Command Class.

3.2.33.3 Security Considerations

CC:008E.02.00.41.001 A node that is included securely MUST NOT accept Association commands unless the commands are received via the highest security key assigned to the device.

CC:008E.02.00.41.002 A supporting node issuing commands via association groups MUST send those commands with its highest granted Security Class.

CC:008E.02.00.42.001 A supporting node issuing Set or Report type commands via association groups SHOULD use Supervision encapsulation only if sending commands with S2 (or higher security) encapsulation.

3.2.33.4 Multi Channel Association Set Command

This command is used to request that one or more destinations are added to a given association group.

CC:008E.02.01.13.001 The destinations MAY be a mix of NodeID destinations and End Point destinations.

CC:008E.02.01.12.001 The receiving node SHOULD add the specified destinations to the specified association group.

CC:008E.02.01.13.002 This command MAY be ignored if the association group is already full.

CC:008E.02.01.11.001 Routing end nodes MUST have return routes assigned to all association destinations.

Table 3.129: Multi Channel Association Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION (0x8E)							
Command = MULTI_CHANNEL_ASSOCIATION_SET (0x01)							
Grouping Identifier							
NodeID 1							
...							
NodeID M							
Marker = MULTI_CHANNEL_ASSOCIATION_SET_MARKER							
Multi Channel NodeID 1							
Bit Address 1	End Point 1						
...							
Multi Channel NodeID N							
Bit Address N	End Point N						

Grouping Identifier (8 bits)

CC:008E.02.01.11.002 This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

CC:008E.02.01.11.003 A node that receives an unsupported Grouping Identifier MUST ignore this command

NodeID (M bytes)

This field specifies a list of NodeID destinations that are to be added to the specified association group as a NodeID association.

CC:008E.02.01.11.004 A NodeID association created via this field MUST be identical to a NodeID association created with the (non-Multi Channel) Association Set command.

Marker (8 bits)

This field is used to indicate the end of NodeID destinations and the start of End Point destinations.
CC:008E.02.01.11.005 The Marker field MUST be set to the value MULTI_CHANNEL_ASSOCIATION_SET_MARKER.
CC:008E.02.01.13.003 The field MAY be omitted if no End Point destinations are specified.

Multi Channel NodeID (N bytes)

The Multi Channel NodeID, Bit Address and End Point fields specify a list of End Points which are to be added to the specified association group as an End Point association.

The complete identification of an End Point destination requires a NodeID as well as an End Point
CC:008E.02.01.13.004 identifier. This command MAY carry multiple copies of the same Multi Channel NodeID in combination with different End Point identifiers.

Bit Address + End Point (N bytes)

CC:008E.02.01.11.006 These fields MUST be processed in combination with the Multi Channel NodeID field.

CC:008E.02.01.12.005 The receiving node SHOULD treat the Bit Address flag and the End Point identifier as one scalar value; thus creating only one association group entry. Using a single scalar value enables:

- Better utilization of association group capacity

- Simple transmission of command for a multi-End Point destination
- Well-defined removal of bit addressed End Points

Refer to the Multi Channel Command Class for the actual encoding of bit addressed End Points.

CC:008E.02.01.11.007 The 7-bit End Point value MUST be in the range 1..127.

3.2.33.5 Multi Channel Association Get Command

This command is used to request the current destinations of a given association group.

CC:008E.02.02.11.001 The Multi Channel Association Report Command MUST be returned in response to this command.

CC:008E.02.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:008E.02.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.130: Multi Channel Association Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION (0x8E)							
Command = MULTI_CHANNEL_ASSOCIATION_GET (0x02)							
Grouping Identifier							

Grouping Identifier (8 bits)

CC:008E.02.02.11.004 This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

CC:008E.02.02.12.001 A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

3.2.33.6 Multi Channel Association Report Command

This command is used to advertise the current destinations for a given association group.

Table 3.131: Multi Channel Association Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION (0x8E)							
Command = MULTI_CHANNEL_ASSOCIATION_REPORT (0x03)							
Grouping Identifier							
Nax Nodes Supported							
Reports to Follow							
NodeID 1							
...							
NodeID M							
Marker = MULTI_CHANNEL_ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID 1							
Bit Address 1	End Point 1						
...							
Multi Channel NodeID N							
Bit Address N	End Point N						

Grouping Identifier (8 bits)

CC:008E.02.03.11.001 This field is used to advertise the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

Max Nodes Supported (8 bits)

The maximum number of destinations supported by the advertised association group. Each destination
CC:008E.02.03.13.001 MAY be a NodeID destination or an End Point destination.

Reports to Follow (8 bits)

The entire list destinations of the advertised association group may be too long for one command.

CC:008E.02.03.11.002 This field MUST advertise how many report frames will follow this report.

NodeID (M bytes)

This field advertises a list of NodeID destinations of the advertised association group.

CC:008E.02.03.11.003 The list of NodeIDs MUST be empty if there are no NodeID destinations configured for the advertised association group.

Marker (8 bits)

Refer to description under the Multi Channel Association Set command.

Multi Channel NodeID (N bytes)

The Multi Channel NodeID, Bit Address and End Point fields specify a list of End Points which are currently in the advertised association group.

The complete identification of an End Point requires a NodeID as well as an End Point identifier. The
CC:008E.02.03.13.002 Multi Channel Association Report command MAY carry multiple copies of the same Multi Channel NodeID in combination with different End Point identifiers.

CC:008E.02.03.11.004 The list of Multi Channel NodeID and End Point identifiers MUST be empty if there are no End Point destinations configured for the advertised association group.

Bit address + End Point (N bytes)

CC:008E.02.03.11.005 These fields MUST be processed in combination with the Multi Channel NodeID field.

Refer to the Multi Channel Command Class for the actual encoding of bit addressed End Points.

3.2.33.7 Multi Channel Association Remove Command

This command is used to remove NodeID and End Point destinations from a given association group.

CC:008E.02.04.11.001 This command MUST manipulate the same list of NodeID destinations as the Association Remove Command.

Table 3.132: Multi Channel Association Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION (0x8E)							
Command = MULTI_CHANNEL_ASSOCIATION_REMOVE (0x04)							
Grouping Identifier							
NodeID 1							
...							
NodeID M							
Marker = MULTI_CHANNEL_ASSOCIATION_REMOVE_MARKER							
Multi Channel NodeID 1							
Bit Address 1	End Point 1						
...							
Multi Channel NodeID N							
Bit Address N	End Point N						

Grouping Identifier (8 bits)

This field is used to specify from which association group the specified destinations should be removed.

CC:008E.02.04.11.002 Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

CC:008E.02.04.11.003 A receiving node MUST ignore an unsupported Grouping Identifier; except for the value 0.

CC:008E.02.04.11.004 This field MUST be interpreted in combination with the NodeID and End Point fields.

(NodeID Destination) NodeID (M bytes)

(End Point Destination) Multi Channel NodeID (N bytes)

(End Point Destination) End Point (N bytes)

These fields specify the destinations that are to be removed.

CC:008E.02.04.11.005 The Grouping Identifier and these fields MUST be interpreted as follows.

"NodeID Destinations" denote the NodeID fields.

"End Point destinations" denote the combined Multi Channel NodeID and End Point fields found after the marker.

CC:008E.02.04.13.001 A receiving node MAY interpret the empty command (only Command Class and Command fields) as an instruction to Remove all NodeID destinations and End Point destinations from all association groups.

CC:008E.02.04.11.006 This field MUST be interpreted according to Table 3.133

Table 3.133: Multi Channel Association Remove, V2::Parameter Interpretation

Grouping identifier	Number of NodeID Destinations	Number of End Point Destinations	Interpretation
> 0	> 0	(don't care)	Remove NodeID destinations from association group. Then assess End Points (next line) – if any specified
> 0	(don't care)	> 0 *)	Remove End Point destinations from association group *).
> 0	= 0	= 0	Remove all NodeID destinations and End Point destinations from association group.
= 0	> 0	(don't care)	Remove NodeID destinations from all association groups. Then assess End Point destinations (next line) – if any specified
= 0	(don't care)	> 0 *)	Remove End Point destinations from all association groups *).
= 0	= 0	= 0	Remove all NodeID destinations and End Point destinations from all association groups.

CC:008E.02.04.11.007
CC:008E.02.04.11.008

*) The receiving node MUST treat the Bit Address flag and the End Point identifier as one scalar value. The receiving node MUST remove End Points if an exact match can be found, while it MAY ignore the removal request if an exact match cannot be found; even though there may exist associations for individual End Point destinations which are covered by the End Point bitmap address.

Marker (8 bits)

Refer to the description in Section 3.2.33.4 in the Multi Channel Association Set Command.

3.2.33.7.1 Examples

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Remove NodeID = 12 from Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							
NodeID = 12							

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Figure 3.17: Remove specified NodeID from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Remove Multi Channel NodeID = 16,3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 3							

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 17							
End Point = 2							

Figure 3.18: Remove Specified End Point from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 134 (b1000 0110)							
Multi Channel NodeID = 17							
End Point = 2							

Remove Multi Channel NodeID = 16,2 and 16,3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 134 (b1000 0110)							

Note: The Multi Channel Node MUST report end points as bit addressable in order to be removed in the same manners.

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 17							
End Point = 2							

Figure 3.19: Remove Bit Addressable End Points from Group 3

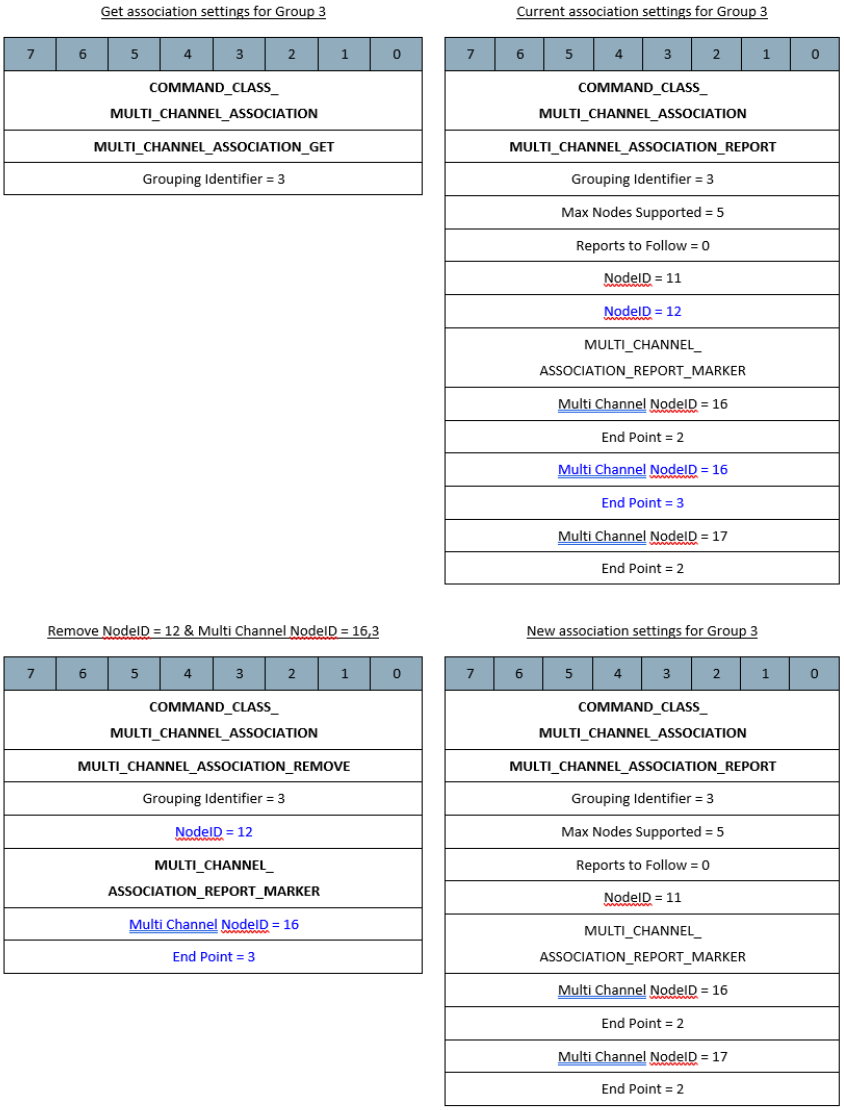


Figure 3.20: Remove Specified NodeID and End Point from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Remove all NodeIDs from Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							

Figure 3.21: Remove all Associations from Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 0							
NodeID = 12							

Figure 3.22: Remove Specific NodeID from all Groups

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 0							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 3							

Figure 3.23: Remove Specific Multi Channel NodeID from all Groups

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							

or

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 0							

Figure 3.24: Remove all Associations from all Groups

3.2.33.8 Multi Channel Association Supported Groupings Get Command

This command is used to request the number of association groups that this node supports.

CC:008E.02.05.11.001 The Multi Channel Association Supported Groupings Report Command MUST be returned in response to this command.

CC:008E.02.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:008E.02.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.134: Multi Channel Association Supported Groupings Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION (0x8E)							
Command = MULTI_CHANNEL_ASSOCIATION_GROUPINGS_GET (0x05)							

3.2.33.9 Multi Channel Association Supported Groupings Report Command

This command is used to advertise the maximum number of association groups implemented by this node.

Table 3.135: Multi Channel Association Supported Groupings Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION (0x8E)							
Command = MULTI_CHANNEL_ASSOCIATION_REMOVE (0x04)							
Supported Groupings							

Supported Groupings (8 bits)

This field is used to advertise the number of association groups that this node supports.

CC:008E.02.06.11.001 Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

CC:008E.02.06.11.002 The value advertised by this field MUST be the total number of supported association groups; whether associations are managed via the Association Command Class or the Multi Channel Association Command Class.

CC:008E.02.06.11.003 Each of these association groups MUST support NodeID destinations as well as End Point destinations.

3.2.34 Multi Channel Association Command Class, version 3

The Multi Channel Association Command Class is used to manage associations to Multi Channel End Point destinations as well as to NodeID destinations.

The following sections specify commands which were extended or added in version 3.

3.2.34.1 Compatibility Considerations

The considerations of section Section 3.2.33.1 also apply to this version the following additions:

CC:008E.03.00.21.001 A device supporting this Command Class version MUST also

- support the Multi Channel Association Command Class, version 2
- support the (non-Multi Channel) Association Command Class, version 2

This version introduces support for the creation of an End Point Association to the Root Device of another device, e.g. a gateway, by allowing the destination End Point 0 as a valid value.

This allows a gateway to create a single Lifeline association from a Multi Channel device and subsequently receive status messages or sensor reports from the Root Device as well as the End Points of that device. Refer to section Section 3.2.34.2.

CC:008E.03.00.21.002 The use of Multi Channel encapsulation MUST comply with Table 3.136.

Table 3.136: V3 Associations and the Transmissions they may Trigger

Source -> Destination	Association Type	Allowed Transmissions
NodeID -> NodeID (V2)	NodeID	Non-encapsulated
End Point -> NodeID (V2)	NodeID	Non-encapsulated
NodeID -> End Point (V2)	End Point	Encapsulated *) (src 0, dst>0)
End Point -> End Point (V2)	End Point	Encapsulated (src>0, dst>0)
End Point -> Root Device (V3)	End Point	Encapsulated (src>0, dst=0)
Root Device -> Root Device (V3)	End Point	Non-encapsulated **) or Encapsulated (src>0, dst=0)

*) The source End Point MAY be different than 0 if the associated group is an End Point Group mapped to the Root Device

**) Command must be sent non-encapsulated if both End Points are zero. However, the End Point association from a Root Device Lifeline group to a Root Device also allows a source End Point to send encapsulated commands to the Root Device Lifeline destination. Refer to Section 3.2.34.2.

3.2.34.2 Z-Wave Plus Considerations

The considerations of version 2 also apply to this version.

Version 3 adds support for creation of an End Point Association to the Root Device of another device, e.g. a gateway.

CC:008E.03.00.11.001 A controlling device MUST verify that a device supports Multi Channel Association Command Class, Version 3 before creating an End Point Association to the Root Device of another device.

Multi Channel End Points may implement functionality relevant to a Lifeline destination, e.g. individual meter reports from a power strip. Multi Channel encapsulation allows the Lifeline destination to distinguish between apparently identical commands from different End Points.

As specified by the Multi Channel Command Class, a Root Device must not use Multi Channel encapsulation when communicating to another Root Device, i.e. if both End Points are zero. It does

however still make sense to create an End Point Association from one Root Device to another Root Device, e.g. a gateway.

Consider, as an example, a two-channel temperature sensor. By creating an End Point association from the Lifeline Association Group to the Root Device of a gateway, the gateway may receive unsolicited Multi Channel encapsulated sensor reports from each of the sensor End Points as well as non-encapsulated battery status messages from the sensor Root Device.

CC:008E.03.00.12.001 The Root Device of a Multi Channel device SHOULD implement support for the Multi Channel Association Command Class if any of the End Points provide association capabilities. If the Root Device
CC:008E.03.00.13.001 implements such support, the Root Device MAY forward Multi Channel encapsulated commands to the association destination on behalf of End Points.

CC:008E.03.00.11.002 A NodeID association MUST NOT be created in response to a request for an End Point association to the Root Device (destination End Point 0).

If a NodeID Association is created from the Root Device Lifeline Association Group, End Point
CC:008E.03.00.11.003 commands MUST NOT be transmitted via the Lifeline Association Group of the root device. All
CC:008E.03.00.11.004 commands originating from the Root Device (Tamper Alarm, Device Reset Locally, etc.) MUST be sent non-encapsulated to the Lifeline destination.

CC:008E.03.00.11.005 If a NodeID association is created, two End Points MUST NOT forward identical commands (e.g. Multilevel Sensor Report::Temperature) via the Lifeline group, as the lack of source End Point information will prevent a receiving application from distinguishing the individual commands from each other.

CC:008E.03.00.11.006 If an End Point Association is created from the Lifeline group, End Points MUST send relevant Multi Channel encapsulated commands to the Lifeline group destination.

CC:008E.03.00.13.002 A Multi Channel device MAY forward commands from multiple Multi Channel End Points to the Lifeline group destination.

3.2.34.3 Security Considerations

The considerations of section Section 3.2.33.3 also apply to this version.

3.2.34.4 Multi Channel Association Set Command

This command is used to request that one or more destinations are added to a given association group.

CC:008E.03.01.13.001 The destinations MAY be a mix of NodeID destinations and End Point destinations.

CC:008E.03.01.12.001 The receiving node SHOULD add the specified destinations to the specified association group.

CC:008E.03.01.13.002 This command MAY be ignored if the association group is already full.

CC:008E.03.01.11.001 Routing end nodes MUST have return routes assigned to all association destinations.

Table 3.137: Multi Channel Association Set Command version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION (0x8E)							
Command = MULTI_CHANNEL_ASSOCIATION_SET (0x01)							
Grouping Identifier							
NodeID 1							
...							
NodeID M							
Marker = MULTI_CHANNEL_ASSOCIATION_SET_MARKER							
Multi Channel NodeID 1							
Bit Address 1	End Point 1						
...							
Multi Channel NodeID N							
Bit Address N	End Point N						

Grouping Identifier (8 bits)

This field is used to specify the actual association group.

CC:008E.03.01.11.002 A node receiving an unsupported Grouping Identifier MUST ignore this command.

NodeID (M bytes)

The NodeID fields specify a list of NodeID destinations that are to be added to the specified association group as a NodeID association.

CC:008E.03.01.11.003 A NodeID association created via this field MUST be identical to a NodeID association created with the (non-Multi Channel) Association Set command.

Marker (8 bits)

This field is used to indicate the end of NodeID destinations and the start of End Point destinations.
CC:008E.03.01.11.004 The Marker field MUST be set to the value MULTI_CHANNEL_ASSOCIATION_SET_MARKER. The field MAY be omitted if no End Point destinations are specified.

Multi Channel NodeID (N bytes)

The Multi Channel NodeID, Bit Address and End Point fields specify a list of End Point destinations which are to be added to the specified association group as an End Point association.

The complete identification of an End Point destination requires a NodeID as well as an End Point
CC:008E.03.01.13.003 identifier. This command MAY carry multiple copies of the same Multi Channel NodeID in combination with different End Point identifiers.

Bit address + End Point (N bytes)

CC:008E.03.01.11.005 These fields MUST be processed in combination with the Multi Channel NodeID field.

CC:008E.03.01.12.005 The receiving node SHOULD treat the Bit Address flag and the End Point identifier as one scalar value; thus creating only one association group entry. Using a single scalar value enables:

- Better utilization of association group capacity
- Simple transmission of command for a multi-End Point destination

- Well-defined removal of bit addressed End Points

Refer to the Multi Channel Command Class for the actual encoding of bit addressed End Points.

CC:008E.03.01.11.006 The 7-bit End Point value MUST be in the range 0..127.

3.2.35 Multi Channel Association Command Class, version 4

The Multi Channel Association Command Class version 4 introduces the capability to discover the highest security class of a target when issuing controlling commands via association groups.

3.2.35.1 Compatibility Considerations

CC:008E.04.00.21.001 If a node supports the version 4 of this Command Class and the Association Command Class, it MUST support the Association Command Class, version 3 or newer.

3.2.35.2 Interoperability Considerations

CC:008E.04.00.31.001 A supporting node MUST respect the requirement defined in Section 3.2.5.2 Interoperability Considerations.

3.2.36 Multi Channel Association Command Class, version 5

The Multi Channel Association Command Class version 5 introduces the capability to discover the highest security class of a target when issuing unsolicited supporting commands (Unsolicited Reports and Notifications) via association groups.

3.2.36.1 Compatibility Considerations

CC:008E.05.00.21.001 If a node supports the version 5 of this Command Class and the Association Command Class, it MUST support the Association Command Class, version 4 or newer.

3.2.36.2 Interoperability Considerations

CC:008E.05.00.31.001 A supporting node MUST respect the requirement defined in Section 3.2.6.2 Interoperability Considerations.

3.2.37 Node Naming and Location Command Class, version 1

The Node Naming and Location Command Class is used to assign a name and a location text string to a supporting node.

3.2.37.1 Node Name Set Command

This command is used to set the name of the receiving node.

Table 3.138: Node Name Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_NAMING (0x77)							
Command = NODE_NAMING_NODE_NAME_SET (0x01)							
Reserved					Char. Presentation		
Node name char 1							
...							
Node name char N							

Node name char (N bytes)

This field is used to indicate the actual assigned name for the node.

This field MUST be encoded using the specified character representation in the Char.Presentation field.

The Node name string MUST NOT contain any appended termination characters.

The length of this field MUST be in the range 1..16 bytes. The length of this field MUST be determined from the frame length.

A node receiving this command with a Node name char longer than 16 bytes MUST ignore any byte/character following the 16th byte.

Devices using the Unicode UTF-16 characters representation can set Name strings using a maximum of 8 characters because each character is encoded with 2 bytes. In this case, the first byte MUST be the most significant byte.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Char. Presentation (3 bits)

This field is used to indicate the character encoding for the Node name char field.

This field MUST comply with Table 3.139:

Table 3.139: Node Name Set::Char. Presentation Encoding

Source -> Value	Description
0x00	Using standard ASCII codes, see *ASCII Codes* (values 128-255 are ignored)
0x01	Using standard and OEM Extended ASCII codes, see *ASCII Codes*.
0x02	Unicode UTF-16

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

3.2.37.2 Node Name Get Command

This command is used to request the stored name from a node.

The *Node Name Report Command* MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

This command is used to set the name of the receiving node.

Table 3.140: Node Name Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_NAMING (0x77)							
Command = NODE_NAMING_NODE_NAME_GET (0x02)							

3.2.37.3 Node Name Report Command

This command is used to advertise the name assigned to the sending node.

Table 3.141: Node Name Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_NAMING (0x77)							
Command = NODE_NAMING_NODE_NAME_REPORT (0x03)							
Reserved					Char. Presentation		
Node name char 1							
...							
Node name char N							

For fields' description, refer to Section 3.2.37.1 Node Name Set Command

A sending node MUST comply with fields' description from Section 3.2.37.1 Node Name Set Command

3.2.37.4 Node Location Set Command

The Node Location Set Command is used to set a location name in a node in a Z-Wave network.

Table 3.142: Node Location Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_NAMING (0x77)							
Command = NODE_NAMING_NODE_LOCATION_SET (0x04)							
Reserved					Char. Presentation		
Node location char 1							
...							
Node location char N							

Node location char (N bytes)

This field is used to indicate the actual assigned location for the node.

This field MUST be encoded using the specified character representation in the Char. Presentation field.

The Node location string MUST NOT contain any appended termination characters.

The length of this field MUST be in the range 1..16 bytes. The length of this field MUST be determined from the frame length.

A node receiving this command with a Node location char longer than 16 bytes MUST ignore any byte/character following the 16th byte.

Devices using the Unicode UTF-16 characters representation can set Location strings using a maximum of 8 characters because each character is encoded with 2 bytes. In this case, the first byte MUST be the most significant byte.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Char. Presentation (3 bits)

This field is used to indicate the character encoding for the Node location char field. This field MUST comply with Table 3.139.

3.2.37.5 Node Location Get Command

This command is used to request the stored node location from a node.

The *Node Location Report Command* MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.143: Node Location Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_NAMING (0x77)							
Command = NODE_NAMING_NODE_LOCATION_GET (0x05)							

3.2.37.6 Node Location Report Command

This command is used to advertise the node location.

Table 3.144: Node Location Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_NAMING (0x77)							
Command = NODE_NAMING_NODE_LOCATION_REPORT (0x06)							
Reserved					Char. Presentation		
Node location char 1							
...							
Node location char N							

For fields' description, refer to Section 3.2.37.4 Node Location Set Command

A sending node MUST comply with fields' description from Section 3.2.37.4 Node Location Set Command.

3.2.38 Remote Association Activation Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS VERSION HAS BEEN OBSOLETED

New implementations MUST NOT use this command class.

The Remote Association Activation Command Class is used to remote activations of Association grouping identifiers in other nodes.

Mandatory requirement: Both 'local' and 'remote' node MUST implement the Association Command Class as illustrated below. In addition the 'local' node MUST implement the Remote Association Configuration Command Class as 'Supported'.

The Remote Association Activation Command Class and the Remote Association Configuration Command Class are additions to the functionality to the existing Association Command Class.

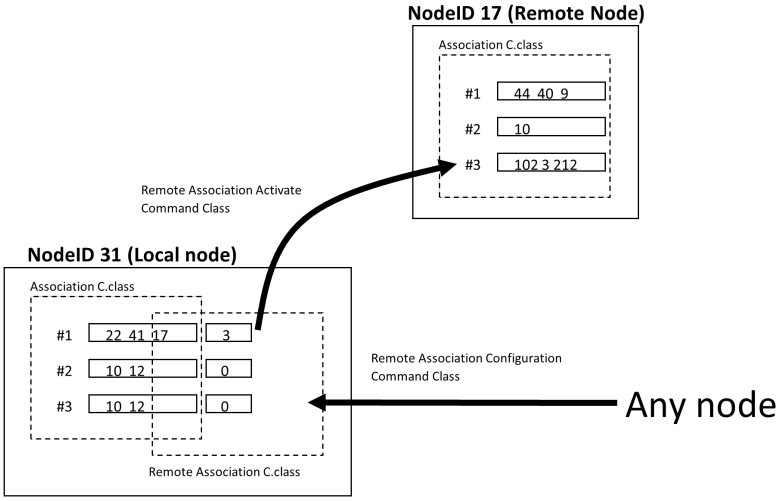


Figure 3.25: Remote Association Activation Command Class

The Remote Association Configuration Command Class enables a 1st node (any node) to configure a 2nd node (local node) to issue a Remote Association Activate Command to a 3rd node (remote node), which instruct the 3rd node to activate one of its locally stored association group identifiers as defined by the Association Command Class.

3.2.38.1 Remote Association Activate Command

The Remote Association Activate Command is used to instruct a 'remote' node to activate one of its locally stored association group identifiers as defined by the Association Command Class. This will subsequently generate a number of Commands being issued from the 'remote node to the NodeIDs associated to the grouping identifier.

Table 3.145: Remote Association Activate Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_REMOTE_ASSOCIATION_ACTIVATE (0x7C)							
Command = REMOTE_ASSOCIATION_ACTIVATE (0x01)							
Grouping identifier							

Grouping identifier (8 bits)

This group identifier used to specify the grouping identifier on the remote node.

3.2.39 Remote Association Configuration Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS VERSION HAS BEEN OBSOLETED

New implementations MUST NOT use this command class.

The Remote Association Configuration Command Class is used to configuration of the Remote Association Activation Command Class.

Mandatory requirement: Both 'local' and 'remote' node MUST implement the Association Command Class as 'supported'. In addition the 'local' node MUST implement the Remote Association Configuration Command Class as 'supported' and the node used to make the configuration ('any node' in the description below) MUST implement it as 'controlled'.

The Remote Association Configuration Command Class is an addition to the functionality of the existing Association Command Class.

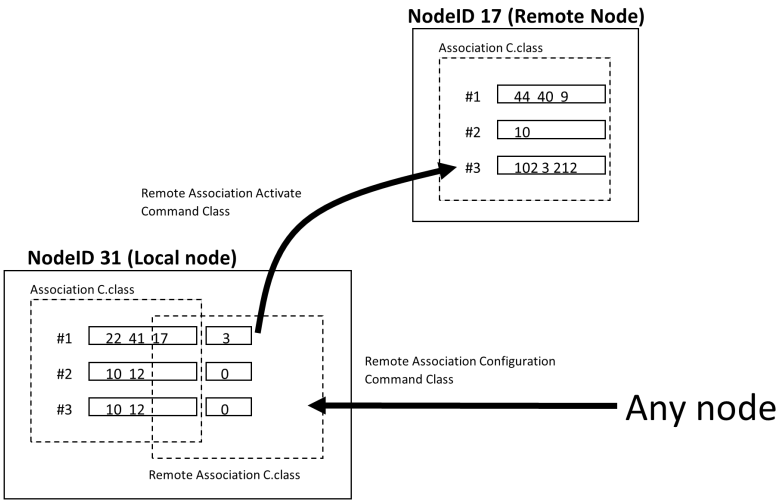


Figure 3.26: Remote Association Configuration Command Class

The Remote Association Configuration Command Class enables a 1st node (Any node) to configure a 2nd node (Local node) to issue a Remote Association Activate Command to a 3rd node (Remote node), which instructs the 3rd node to activate one of its locally stored association group identifiers as defined by its Association Command Class.

3.2.39.1 Remote Association Configuration Set Command

The Remote Association Configuration Set Command links two nodes' 'Association Command Class' defined grouping identifiers together. It allows one node (local node) to use its grouping identifiers to control a second node's (remote node) grouping identifiers, using the Remote Association Activation Command Class.

Table 3.146: Remote Association Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_REMOTE_ASSOCIATION (0x7D)							
Command = REMOTE_ASSOCIATION_CONFIGURATION_SET (0x01)							
Local Grouping identifier							
Remote NodeID							
Remote Grouping identifier							

Local Grouping identifier (8 bits)

Like in the Association Command Class, the Local Grouping identifier is used to specify the grouping identifier on the local node.

A Local grouping identifier = 0x0 will erase all links between local and remote grouping identifiers.

Remote NodeID (8 bits)

This NodeID used to specify the Node, which should receive the Remote Association Activate Command.

A NodeID = 0x0 will remove a link between the specified local grouping identifier and a remote grouping identifier.

Remote Grouping identifier (8 bits)

This group identifier used to specify the grouping identifier on the remote node.

3.2.39.2 Remote Association Configuration Get Command

The Remote Association Configuration Get Command is used to request the link between a Local Grouping identifier and a Remote Grouping identifier on a node.

The Remote Association Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.147: Remote Association Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_REMOTE_ASSOCIATION (0x7D)							
Command = REMOTE_ASSOCIATION_CONFIGURATION_GET (0x02)							
Local Grouping identifier							

Local Grouping identifier (8 bits)

Like in the Association Command Class, the Local Grouping identifier is used to specify the grouping identifier on the local node.

3.2.39.3 Remote Association Configuration Report Command

The Remote Association Configuration Report Command returns the remote NodeID and the grouping identifier.

Table 3.148: Remote Association Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_REMOTE_ASSOCIATION (0x7D)							
Command = REMOTE_ASSOCIATION_CONFIGURATION_REPORT (0x03)							
Local Grouping identifier							
Remote NodeID							
Remote Grouping identifier							

Local Grouping identifier (8 bits)

This group identifier used to specify the grouping identifier on the local node

Remote NodeID (8 bits)

This NodeID used to specify the remote node that the Remote Association Activate Command is sent to. If no link is established between the Local Grouping Identifier and a Remote Grouping Identifier, the Remote NodeID will return zero (0x0)

Remote Grouping identifier (8 bits)

This Remote grouping identifier used to specify the grouping identifier on the remote node that should be activated.

3.2.40 Time Command Class, version 1

The Time Command Class, version1 is used to read date and time from a supporting node in a Z-Wave network.

3.2.40.1 Compatibility considerations

Notice that the former Time Command Class version 1 (Revision 4 of this document) is discontinued and replaced by a new one.

3.2.40.1.1 Node Information Frame (NIF)

CC:008A.01.00.21.001 A supporting node MUST always advertise the Time Command Class in its NIF, regardless of the security bootstrapping outcome. This allows other nodes bootstrapped on any security level to request the current time from a supporting node.

3.2.40.2 Interoperability considerations

Nodes supporting this Command Class are time servers for other nodes in a Z-Wave network. Other nodes can learn the current date and time by querying nodes supporting this Command Class.

CC:008A.01.00.32.001 For nodes based on an end node Role Type, it is RECOMMENDED to support a dedicated Association Group which issues the Time Get Command and/or the Date Get Command.

CC:008A.01.00.32.002 Controlling nodes SHOULD automatically associate such association groups to a node supporting the Time Command Class.

3.2.40.3 Time Get Command

This command is used to request the current time from a supporting node.

CC:008A.01.01.11.001 The Time Report Command MUST be returned in response to this command.

CC:008A.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:008A.01.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.149: Time Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME (0x8A)							
Command = TIME_GET (0x01)							

3.2.40.4 Time Report Command

This command is used to report the current time.

Table 3.150: Time Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME (0x8A)							
Command = TIME_REPORT (0x02)							
RTC failure	Reserved			Hour Local Time			
Minute Local Time							
Second Local Time							

RTC failure (1 bit)

CC:008A.01.02.11.001 Many RTC chips have a stop bit indicating if the oscillator has been stopped. The RTC failure bit MUST be set to 1 in order to indicate to the receiving node that the RTC has been stopped and that the advertised time might be inaccurate.

CC:008A.01.02.11.002 If the sending node does not support this feature or if the oscillator has not been stopped, it MUST set this field to 0.

CC:008A.01.02.11.003 If the receiving node does not support this feature, it MUST ignore this field.

Reserved

CC:008A.01.02.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Hour Local Time (8 bits)

This field is used to indicate the number of complete hours that have passed since midnight in local time.

CC:008A.01.02.11.005 This field MUST be in the range 0..23.

Minute Local Time (8 bits)

This field is used to indicate the number of complete minutes that have passed since the start of the hour in local time.

CC:008A.01.02.11.006 This field MUST be in the range 0..59.

Second Local Time (8 bits)

This field is used to indicate the number of complete seconds that have passed since the start of the minute in local time.

CC:008A.01.02.11.007 This field MUST be in the range 0..59.

Note: the time synchronization between nodes may vary by a few seconds due to delays introduced by the wireless communication.

3.2.40.5 Date Get Command

This command is used to request the current date adjusted according to the local time zone and Daylight Saving Time from a supporting node.

CC:008A.01.03.11.001 The Date Report Command MUST be returned in response to this command.

CC:008A.01.03.11.002 This command MUST NOT be issued via multicast addressing.

CC:008A.01.03.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.151: Date Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME (0x8A)							
Command = DATE_GET (0x03)							

3.2.40.6 Date Report Command

This command is used to advertise the current date adjusted according to the local time zone and Daylight Saving Time.

Table 3.152: Date Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME (0x8A)							
Command = DATE_REPORT (0x04)							
Year 1							
Year 2							
Month							
Day							

Year (16 bits)

CC:008A.01.04.11.001 This field MUST specify the current year using the Gregorian calendar. The first byte (Year 1) MUST be the most significant byte.

For example, Year1 = 0x07 and Year2 = 0xD7 MUST indicate year 2007

Month (8 bits)

CC:008A.01.04.11.002 This field MUST specify the current month of the year.

CC:008A.01.04.11.003 This field MUST be in the range 1..12 (representing respectively January…December).

Day (8 bits)

CC:008A.01.04.11.004 This field MUST specify current the day of the month.

CC:008A.01.04.11.005 This field MUST be in the range 1..31.

3.2.41 Time Command Class, version 2

The Time Command Class, version 2 is used to read date and time from a supporting node in a Z-Wave network.

3.2.41.1 Compatibility considerations

The Time Command Class, version 2 is backwards compatible with the Time Command Class, version 1.

The Time Command Class, version 2 enables reading and setting Time Zone Offset and Daylight Saving Time parameters. The data formats are based on the International Standard ISO 8601.

CC:008A.02.00.21.001 Commands not described in this version MUST remain unchanged from version 1.

3.2.41.1.1 Node Information Frame (NIF)

CC:008A.02.00.21.002 A supporting node MUST always advertise the Time Command Class in its NIF, regardless of the security bootstrapping outcome.

This allows other nodes bootstrapped on any security level to request the current time from a supporting node.

3.2.41.2 Interoperability considerations

Nodes supporting this Command Class are time servers for other nodes in a Z-Wave network. Other nodes can learn the current date and time by querying nodes supporting this Command Class.

CC:008A.02.00.32.001 For nodes based on an end node Role Type, it is RECOMMENDED to support a dedicated Association Group which issues the Time Get Command, the Date Get Command and/or the Time Offset Get Command.

CC:008A.02.00.32.002 Controlling nodes SHOULD automatically associate such association groups to a node supporting the Time Command Class, version 2.

CC:008A.02.00.33.001 The purpose of this Command Class is to read the current date and time information from supporting nodes. Hence, supporting nodes MAY ignore the Time Offset Set Command introduced in version 2 if they rely on another source to retrieve the Time Zone Offset and Daylight Saving Time information.

3.2.41.3 Time Offset Get Command

This command is used to request the Time Zone Offset (TZO) and Daylight Savings Time (DST) parameters from a supporting node.

CC:008A.02.06.11.001 The Time Offset Report Command MUST be returned in response to this command.

CC:008A.02.06.11.002 This command MUST NOT be issued via multicast addressing.

CC:008A.02.06.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.153: Time Offset Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME (0x8A)							
Command = TIME_OFFSET_GET (0x06)							

3.2.41.4 Time Offset Set Command

This command is used to set Time Zone Offset (TZO) and Daylight Savings Time (DST) at the supporting node.

CC:008A.02.05.13.001 This command MAY be ignored by a supporting node if it relies on another source to retrieve the TZO/DST information.

CC:008A.02.05.12.001 This command SHOULD be ignored if it is not received at the highest granted security level.

Table 3.154: Time Offset Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME (0x8A)							
Command = TIME_OFFSET_SET (0x05)							
Sign TZO	Hour TZO						
Minute TZO							
Sign Offset DST	Minute Offset DST						
Month Start DST							
Day Start DST							
Hour Start DST							
Month End DST							
Day End DST							
Hour End DST							

Sign TZO (1 bit)

This field is used to indicate the sign (plus or minus) to apply to the Hour TZO and Minute TZO fields.

CC:008A.02.05.11.001 The value 0 MUST indicate the Plus sign (positive offset from UTC)

CC:008A.02.05.11.002 The value 1 MUST indicate the Minus sign (negative offset from UTC)

Hour TZO (7 bits)

This field is used to indicate the number of hours that the originating time zone deviates from UTC.

CC:008A.02.05.12.002 This field SHOULD be in the range 0..14.

Minute TZO (7 bits)

This field is used to indicate the number of minutes that the originating time zone deviates UTC.

CC:008A.02.05.11.003 This field MUST be in the range 0..59.

Sign Offset DST (1 bit)

This field is used to indicate the sign (plus or minus) for the Minute Offset DST field to apply to the current time while in the Daylight Saving Time.

CC:008A.02.05.11.004 The value 0 MUST indicate the Plus sign (positive offset from current time),

CC:008A.02.05.11.005 The value 1 MUST indicate the Minus sign (negative offset from current time).

Minute Offset DST (7 bits)

CC:008A.02.05.11.006 This field MUST indicate the number of minutes by which the current time is to be adjusted when Daylight Saving Time starts.

Month Start DST (8 bits)

CC:008A.02.05.11.007 This field MUST indicate the month of the year when Daylight Saving Time starts.

CC:008A.02.05.11.008 This field MUST be in the range 0..12. 1..12 representing respectively January…December or 0 if Daylight Saving Time is not used.

Day Start DST (8 bits)

CC:008A.02.05.11.009

This field MUST indicate the day of the month when Daylight Saving Time starts.

CC:008A.02.05.11.00A This field MUST be in the range 0..31. 1..31 representing the day of the month or 0 if Daylight Saving Time is not used.

Hour Start DST (8 bits)

CC:008A.02.05.11.00B This field MUST indicate the hour of the day when Daylight Saving Time starts.

CC:008A.02.05.11.00C This field MUST be in the range 0..23 representing minutes or 0 if Daylight Saving Time is not used.

Month End DST (8 bits)

CC:008A.02.05.11.00D This field MUST indicate the month of the year when Daylight Saving Time ends.

CC:008A.02.05.11.00E This field MUST be in the range 0..12. 1..12 representing respectively January…December or 0 if Daylight Saving Time is not used.

Day End DST (8 bits)

CC:008A.02.05.11.00F This field MUST indicate the day of the month when Daylight Saving Time ends.

CC:008A.02.05.11.010 This field MUST be in the range 0..31. 1..31 representing the day of the month or 0 if Daylight Saving Time is not used.

Hour End DST (8 bits)

CC:008A.02.05.11.011 This field MUST indicate the hour of the day when Daylight Saving Time ends.

CC:008A.02.05.11.012 This field MUST be in the range 0..23 representing minutes or 0 if Daylight Saving Time is not used.

3.2.41.5 Time Offset Report Command

This command is used to advertise the Time Zone Offset (TZO) and Daylight Savings Time (DST) parameters.

Table 3.155: Time Offset Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME (0x8A)							
Command = TIME_OFFSET_REPORT (0x07)							
Sign TZO	Hour TZO						
Minute TZO							
Sign Offset DST	Minute Offset DST						
Month Start DST							
Day Start DST							
Hour Start DST							
Month End DST							
Day End DST							
Hour End DST							

For fields' description, refer to Section 3.2.41.4 Time Offset Set Command.

CC:008A.02.02.11.001 A sending node MUST comply with fields' description from Section 3.2.41.4 Time Offset Set Command.

3.2.42 Time Parameters Command Class, version 1

The Time Parameters Command Class is used to set date and time in a device hosting this facility. In case the clock is updated via an external source such as SAT, internet, Rugby/Frankfurt source, omit this command class. Time zone offset and daylight savings may be set in the Time Command Class if necessary. The data formats are based on the International Standard ISO 8601.

3.2.42.1 Time Parameters Set Command

This command is used to set current date and time in Universal Time (UTC). Be aware that the communication overhead may be significant in case routing is necessary.

Table 3.156: Time Parameters Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME_PARAMETERS (0x8B)							
Command = TIME_PARAMETERS_SET (0x01)							
Year 1							
Year 2							
Month							
Day							
Hour UTC							
Minute UTC							
Second UTC							

Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Day (8 bits)

Specify the day of the month between 01 and 31.

Hour UTC (8 bits)

Specify the number of complete hours that have passed since midnight (00..23) in UTC.

Minute UTC (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00..59) in UTC. Minutes are measured in Universal Time (UTC).

Second UTC (8 bits)

Specify the number of complete seconds since the start of the minute (00..59) in UTC. Seconds are measured in Universal Time (UTC).

3.2.42.2 Time Parameters Get Command

This command is used to request date and time parameters.

The Time Parameters Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.157: Time Parameters Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME_PARAMETERS (0x8B)							
Command = TIME_PARAMETERS_GET (0x02)							

3.2.42.3 Time Parameters Report Command

This command is used to advertise date and time.

Table 3.158: Time Parameters Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_TIME_PARAMETERS (0x8B)							
Command = TIME_PARAMETERS_REPORT (0x03)							
Year 1							
Year 2							
Month							
Day							
Hour UTC							
Minute UTC							
Second UTC							

Refer to description under the Time Parameters Set Command.

3.2.43 Version Command Class, version 1 [OBSOLETED]

> **Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED**
>
> New implementations MUST NOT support this Command Class.
>
> New implemenetations MUST use the *Version Command Class, version 2*, or newer instead.

The Version Command Class may be used to obtain the Z-Wave library type, the Z-Wave protocol version used by the application, the individual Command Class versions used by the application and the vendor specific application version from a Z-Wave enabled device.

3.2.43.1 Compatibility Considerations

CC:0086.01.00.21.001 In a Multi Channel device, the Version Command Class MUST be supported by the Root Device,
CC:0086.01.00.22.001 while the Version Command Class SHOULD NOT be supported by individual End Points.

CC:0086.01.00.21.002 There may be cases where a given Command Class is not implemented by the Root Device of a Multi Channel device. However, the Root Device MUST respond to Version requests for any Command Class implemented by the Multi Channel device; also in cases where the actual Command Class is only provided by an End Point.

A node supporting a Command Class having a version higher than 1 must support the Version Command Class to be able to identify the supported version. If a node does not support the *Version Command Class, version 1 [OBSOLETED]* at its highest security level, it may be assumed that all command classes implement version 1.

3.2.43.2 Security Considerations

The Version Command Class provides information about the device implementation. This information can potentially be used by an attacker to find vulnerabilities.

CC:0086.01.00.41.002 If a node supports this Command Class and does not support the Security 2 Command Class, it MUST comply with Table 3.159.

Table 3.159: Version Command Class Support

	After Non-Secure Inclusion	After Secure Inclusion
Non-Secure or Less- Secure communication	The node MUST support the Version Command Class and advertise it in the NIF.	The node MUST NOT support the Version Command Class and MUST NOT advertise it in the NIF or Security Commands Supported Report commands.
Secure communication at the highest security class	N/A	The node MUST support the Version Command Class and advertise it in the Security Commands Supported Report commands.

3.2.43.3 Version Get Command

The Version Get Command is used to request the library type, protocol version and application version from a device that supports the Version Command Class.

CC:0086.01.11.11.001 The Version Report Command MUST be returned in response to this command.

CC:0086.01.11.11.002 This command MUST NOT be issued via multicast addressing.

CC:0086.01.11.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.160: Version Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_GET (0x11)							

3.2.43.4 Version Report Command

The Version Report Command is be used to advertise the library type, protocol version and application version from a device.

Table 3.161: Version Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_REPORT (0x12)							
Z-Wave Library Type							
Z-Wave Protocol Version							
Z-Wave Protocol Sub Version							
Application Version							
Application Sub Version							

Z-Wave Library Type (8 bits)

CC:0086.01.12.11.001 This field MUST carry the Z-Wave Protocol Library Type.

CC:0086.01.12.11.002 The value MUST comply with Table 3.162.

Table 3.162: Z-Wave Library Type

Library Type	Description	Version
0x00	N/A	-
0x01	Static Controller	1
0x02	Controller	1
0x03	Enhanced End Node	1
0x04	End Node	1
0x05	Installer	1
0x06	Routing End Node	1
0x07	Bridge Controller	1
0x08	Device Under Test (DUT)	1
0x09	N/A	1
0x0A	AV Remote	1
0x0B	AV Device	1

CC:0086.01.12.11.003 Values marked as not applicable (N/A) MUST be ignored by a receiving node.

CC:0086.01.12.11.004 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Z-Wave Protocol Version & Z-Wave Protocol Sub Version

These fields advertise information specific to Software Development Kits (SDK) provided by stack manufacturers.

Application Version (8 bits)

Returns the Application Version and can have values in the range 0 to 255. The manufacturer assigns the Application Version.

Application Sub Version (8 bits)

Returns the Application Sub Version and can have values in the range 0 to 255. The manufacturer assigns the Application Sub Version.

3.2.43.5 Version Command Class Get Command

The Version Command Class Get Command is used to request the individual command class versions from a device.

CC:0086.01.13.11.001 The Version Command Class Report Command MUST be returned in response to this command.

Only versions from the command classes shown in the NIF, in the Security Command Supported Report or in the Multi Channel Capability Report can be requested.

It is not possible to get a version number for the Generic and Specific Device Classes.

CC:0086.01.13.11.002 This command MUST NOT be issued via multicast addressing.

CC:0086.01.13.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

The Version Report Command is be used to advertise the library type, protocol version and application version from a device.

Table 3.163: Version Command Class Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_COMMAND_CLASS_GET (0x13)							
Requested Command Class							

Requested Command Class (8 bits)

This field is used to indicate the Command Class identifier that is being requested.

3.2.43.6 Version Command Class Report Command

The Version Command Class Report Command is used to report the individual command class versions from a device.

Table 3.164: Version Command Class Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_COMMAND_CLASS_REPORT (0x14)							
Requested Command Class							
Command Class Version							

Requested Command Class (8 bits)

The Requested Command Class field specifies what Command Class the returned version belongs to.

Command Class Version (8 bits)

If the requested Command Class is supported, this field is used to return the Command Class Version.
CC:0086.01.14.11.001 This field MUST be in the range 1..255. It starts with 1 and is incremented every time a new version of the Command Class is released.

CC:0086.01.14.11.002 If the requested Command Class is not supported or controlled, a responding node MUST set the Command Class Version field to 0x00.

3.2.44 Version Command Class, version 2

The Version Command Class, version 2 is extended to report the version of various firmware images such as a host processor firmware, etc. in addition to the firmware image running in the Z-Wave chip.

As an example, one may construct a product comprising a Z-Wave chip and a secondary host processor that maintains a security certificate. With Firmware Update Meta Data Command Class, version 3 the Z-Wave chip, the host processor and the security certificate may all be updated via individual firmware IDs. Version 2 of the Version Command Class (this Command Class) allows a controlling node to request the corresponding version information for each firmware ID.

Commands not mentioned here remain the same as specified for Version Command Class, version 1.

3.2.44.1 Version Report Command

This command is used to report the library type, protocol version and application version from a node.

Version 2 of this command renames the fields Application Version and Application Sub Version to Firmware 0 Version and Firmware 0 Sub Version. The use remains the same.

CC:0086.02.12.11.001 A node MUST advertise the version of all firmware images which can be updated via the Firmware Update Command Class.

CC:0086.02.12.11.002 A one-chip system MUST comply with the following:

CC:0086.02.12.11.003
- The Firmware 0 Version MUST reflect the complete firmware implementing the Z-Wave protocol stack as well as the Z-Wave application.

CC:0086.02.12.11.004 A multi-processor system MUST comply with the following:

CC:0086.02.12.11.005
- The Firmware 0 Version MUST reflect the firmware implementing the Z-Wave protocol stack and the inter-chip interface module that enables the Z-Wave application to run in the host processor.

CC:0086.02.12.11.006
CC:0086.02.12.13.001
  Another firmware number (e.g. Firmware 1) version MUST reflect the Z-Wave application that runs in the host processor. Any firmware number larger than 0 MAY be used for this purpose.

CC:0086.02.12.13.002 A node MAY advertise the version of additional images hosted by the node; even if such images cannot be updated via the Firmware Update Command Class.

An illustration is given in Figure 3.27 and Figure 3.28.

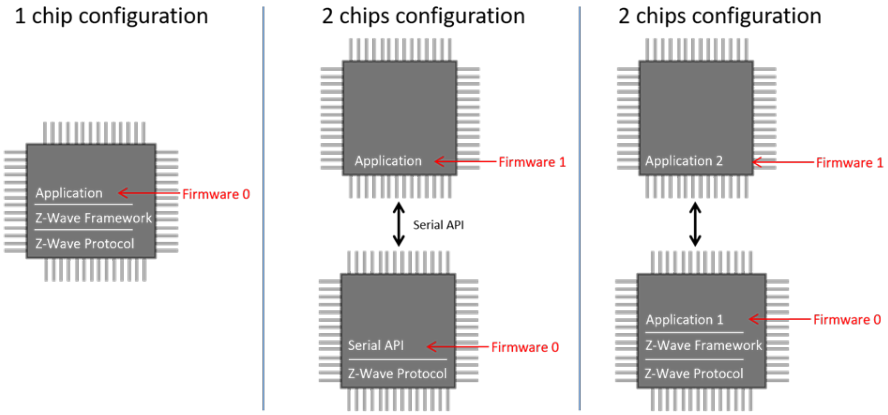


Figure 3.27: Version Report::Firmware Numbering

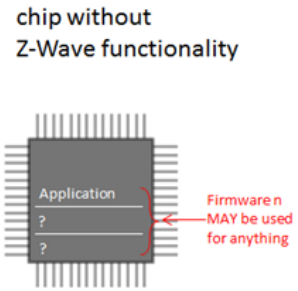


Figure 3.28: Version Report::Firmware Numbering (2)

Further, version 2 of the Version Report command introduces a Hardware Version field as well as new version fields for each additional firmware image implemented by the actual device.

Table 3.165: Version Report Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_REPORT (0x12)							
Z-Wave Protocol Library Type							
Z-Wave Protocol Version							
Z-Wave Protocol Sub Version							
Firmware 0 Version							
Firmware 0 Sub Version							
Hardware Version							
Number of firmware targets							
Firmware 1 Version							
Firmware 1 Sub Version							
...							
Firmware N Version							
Firmware N Sub Version							

Z-Wave Library Type (8 bits)

For a description, refer to Section 3.2.43.4.

Z-Wave Protocol Version and Z-Wave Protocol Sub Version

These fields advertise information specific to Software Development Kits (SDK) provided by stack manufacturers.

Firmware 0 Version (8 bits)

CC:0086.02.12.11.007 Returns the Firmware 0 Version. Firmware 0 is dedicated to the Z-Wave chip firmware. The manufacturer MUST assign a version number. Previously called Application Version.

Firmware 0 Sub Version (8 bits)

CC:0086.02.12.11.008 Returns the Firmware 0 Sub Version. Firmware 0 is dedicated to the Z-Wave chip firmware. The manufacturer MUST assign a sub version number. Previously called Application Sub Version.

Hardware Version (8 bits)

CC:0086.02.12.11.009 The Hardware Version field MUST report a value which is unique to this particular version of the
CC:0086.02.12.11.00A product. The value MUST be updated to a new value every time the hardware is modified for a given
CC:0086.02.12.12.001 product. The value 0x00 SHOULD NOT be used for the Hardware Version.

CC:0086.02.12.11.00B It MUST be possible to uniquely determine the hardware characteristics from the Hardware Version field in combination with the Manufacturer ID, Product Type ID and Product ID fields of Manufacturer Specific Info Report of the Manufacturer Specific Command Class.

This information allows a user to pick a firmware image version that is guaranteed to work with this particular version of the product.

Note that the Hardware Version field is intended for the hardware version of the entire product, not just the version of the Z-Wave radio chip.

Number of Firmware Targets (8 bits)

CC:0086.02.12.11.00C The Number of Firmware Targets field MUST report the number of firmware Version + Sub Version fields following this field. The Firmware 0 Version fields are not included.

CC:0086.02.12.11.00D The field MUST be zero if the device only implements a Firmware 0 target, the Z-Wave chip.

Firmware Version (N * 8 bits)

CC:0086.02.12.11.00E Returns the Firmware n Version. The manufacturer MUST assign a unique Firmware n Version.

Firmware Sub Version (N * 8 bits)

CC:0086.02.12.11.00F Returns the Firmware n Sub Version. The manufacturer MUST assign a unique Firmware n Sub Version.

3.2.45 Version Command Class, version 3

The Version Command Class, version 3 allows supporting nodes to advertise capabilities related to the Version Command Class and optionally provide a detailed list of information regarding implementation on the Z-Wave chip.

3.2.45.1 Compatibility considerations

The Version Command Class, version 3 is backwards compatible with the Version Command Class, version 2. A node supporting the Version Command Class, version 3 MUST also support the Version Command Class, version 2.

All commands and fields not described in this version remain unchanged from version 2.

The Version Command Class, version 3 introduces the possibility for a node to advertise its supported Version Command Class capabilities with the following commands:

- Version Capabilities Get Command
- Version Capabilities Report Command

Supporting nodes MUST advertise support for commands present in previous versions of this Command Class for backwards compatibility.

The following commands are optionally implemented by supporting nodes:

- Version Z-Wave Software Get Command
- Version Z-Wave Software Report Command

The Version Z-Wave Software Get/Report commands are used to provide detailed information on the actual software running on the Z-Wave radio SoC.

3.2.45.2 Version Capabilities Get Command

This command is used to request which version commands are supported by a node.

The Version Capabilities Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.166: Version Capabilities Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_CAPABILITIES_GET (0x15)							

3.2.45.3 Version Capabilities Report Command

This command is used to advertise the version commands supported by the sending node.

Table 3.167: Version Capabilities Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_CAPABILITIES_REPORT (0x16)							
Reserved					ZWS	CC	V

V (Version) (1 bit)

This field is used to advertise support for the version information queried with the Version Get Command.

This field MUST be set to 1.

CC (Command Class) (1 bit)

This field is used to advertise support for the Command Class version information queried with the Version Command Class Get Command.

This field MUST be set to 1.

ZWS (Z-Wave Software) (1 bit)

This field is used to advertise support for the detailed Z-Wave software version information queried with the Version Z-Wave Software Get Command.

The value 1 MUST indicate that the sending node supports the Version Z-Wave Software Get Command.

The value 0 MUST indicate that the sending node does not support the Version Z-Wave Software Get Command.

3.2.45.4 Version Z-Wave Software Get Command

This command is used to request the detailed Z-Wave chip software version information of a node.

A node advertising no support for detailed Z-Wave software version information in the *Version Capabilities Report Command* MUST ignore this command.

The *Version Z-Wave Software Report Command* MUST be returned in response to this command unless it is to be ignored.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.168: Version Z-Wave Software Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_ZWAVE_SOFTWARE_GET (0x17)							

3.2.45.5 Version Z-Wave Software Report Command

This command is used to advertise the detailed Z-Wave chip software version information of a node.

Table 3.169: Version Z-Wave Software Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_VERSION (0x86)							
Command = VERSION_ZWAVE_SOFTWARE_REPORT (0x18)							
SDK version 1 (MSB)							
SDK version 2							
SDK version 3 (LSB)							
Z-Wave Application Framework API Version 1 (MSB)							
Z-Wave Application Framework API Version 2							
Z-Wave Application Framework API Version 3 (LSB)							
Z-Wave Application Framework Build Number 1 (MSB)							
Z-Wave Application Framework Build Number 2 (LSB)							
Host Interface Version 1 (MSB)							
Host Interface Version 2							
Host Interface Version 3 (LSB)							
Host Interface Build Number 1 (MSB)							
Host Interface Build Number 2 (LSB)							
Z-Wave Protocol Version 1 (MSB)							
Z-Wave Protocol Version 2							
Z-Wave Protocol Version 3 (LSB)							
Z-Wave Protocol Build Number 1 (MSB)							
Z-Wave Protocol Build Number 2 (LSB)							
Application Version 1 (MSB)							
Application Version 2							
Application Version 3 (LSB)							
Application Build Number 1 (MSB)							
Application Build Number 2 (LSB)							

SDK version (24 bits)

This field is used to advertise the SDK version used for building the Z-Wave chip software components for the node.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the SDK version.

Z-Wave Application Framework API Version (24 bits)

This field is used to advertise the Z-Wave Application Framework API version used by the node.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the Z-Wave Application Framework API version.

Z-Wave Application Framework Build Number (16 bits)

This field is used to advertise the Z-Wave Application Framework build number running on the node.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the Z-Wave Application Framework build number.

This field MUST be set to 0 if the Application Framework API Version field is set to 0.

Host Interface Version (24 bits)

This field is used to advertise the version of the Serial API exposed to a host CPU or a second Chip.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the actual API version.

This field MUST be set to 0 by a node running on a single chip.

Host Interface Build Number (16 bits)

This field is used to advertise the build number of the Serial API software exposed to a host CPU or second Chip.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the actual API build number.

This field MUST be set to 0 if the Host Interface Version field is set to 0.

Z-Wave Protocol Version (24 bits)

This field is used to advertise the Z-Wave protocol version used by the node.

This field MUST be set to the same value as the Z-Wave Protocol Version and Z-Wave Protocol Sub Version fields present in the Version Report Command.

Byte 1 of this field MUST be set to the same value as the Z-Wave Protocol Version field present in the Version Report Command.

Byte 2 of this field MUST be set to the same value as the Z-Wave Protocol Sub Version field present in the Version Report Command.

Z-Wave Protocol Build Number (16 bits)

This field is used to advertise the actual build number of the Z-Wave protocol software used by the node.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the Z-Wave protocol build number.

Application Version (24 bits)

This field is used to advertise the version of application software used by the node on its Z-Wave chip.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the application version.

This field MUST be set to 0 by a node if its application is running on a host CPU.

If a supporting node's application is running on the Z-Wave chip:

- Byte 1 of this field MUST be set to the same value as the Application Version / Firmware 0 Version field present in the Version Report Command.
- Byte 2 of this field MUST be set to the same value as the Application Sub Version / Firmware 0 Sub Version field present in the Version Report Command.

Application Build Number (16 bits)

This field is used to advertise the actual build of the application software used by the node on its Z-Wave chip.

The value 0 MUST indicate that this field is unused. Other values MUST indicate the application build number.

3.2.45.5.1 Version fields

Fields indicating version numbers are composed of 3 bytes. Version fields MUST be used and interpreted in the following manner:

If used, Byte 1 of a version field MUST represent the major version digit.

If used, Byte 2 of a version field MUST represent the minor version digit.

If used, Byte 3 of a version field MUST represent the patch version digit.

For example, if the node used Z-Wave SDK version 6.51.09, it MUST set the SDK version field to 0x063309. Unused bytes MUST be set to 0.

3.2.46 Wake Up Command Class, version 1

The Wake Up Command Class allows a battery-powered device to notify another device (always listening), that it is awake and ready to receive any queued commands.

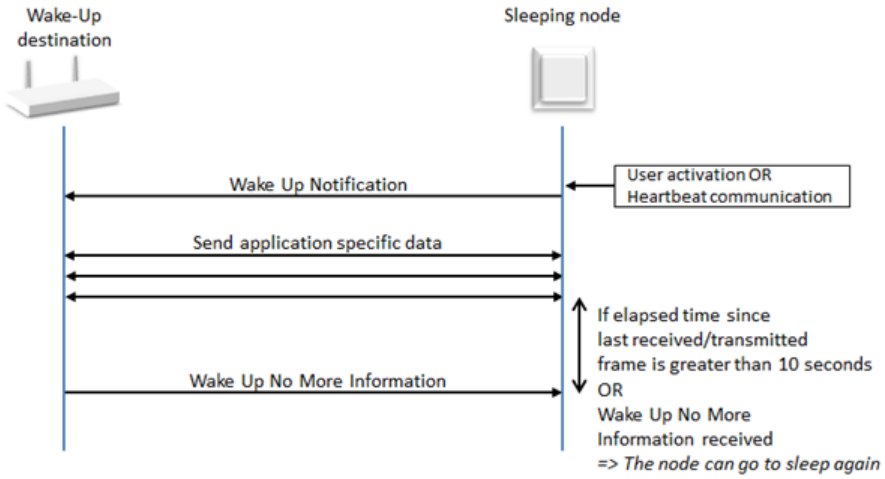


Figure 3.29: Wake Up Sequence

CC:0084.01.00.12.001 This Command Class SHOULD be implemented by battery powered devices. Wake Up Notification commands SHOULD be handled immediately by the destination node in order to minimize battery consumption.

3.2.46.1 Terminology

The Wake Up period is defined as the period during which a Wake Up node is supposed to be awake after issuing a Wake Up Notification Command to its Wake Up destination.

The Wake Up period starts when the supporting node issues a Wake Up Notification and it ends either 10 seconds after the last received/transmitted frame or at the reception of a Wake Up No More Information Command by the Wake Up destination.

3.2.46.2 Interoperability considerations

CC:0084.01.00.31.001 The Wake Up Notification Command Class MUST NOT be covered by any association group. This also means that the Wake Up Notification Command Class MUST NOT be covered by the Lifeline association group.

CC:0084.01.00.32.001 A node sending periodical or triggered unsolicited commands at the same time it wakes up SHOULD send the unsolicited commands before the Wake Up Notification Command.

CC:0084.01.00.32.002 A supporting node SHOULD NOT send unsolicited commands during the Wake Up period.

CC:0084.01.00.32.003 A supporting node having no Wake-Up destination set SHOULD fall back on the SIS NodeID (if any present in the network) to issue Wake-Up Notifications.

3.2.46.3 Wake Up Interval Set Command

This command is used to configure the Wake Up interval and destination of a node.

Table 3.170: Wake Up Interval Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_INTERVAL_SET (0x04)							
Seconds 1 (MSB)							
Seconds 2							
Seconds 3 (LSB)							
NodeID							

Seconds (24 bits)

This field is used to specify the time in seconds between Wake Up periods for the receiving node.

CC:0084.01.04.11.001 The first byte MUST be the most significant byte.

CC:0084.01.04.11.002 Values in the range 1..16777215 MUST indicate the time between Wake Up periods.

The value 0 MUST indicate that the receiving node MUST Wake Up when initiated by an event determined by the application e.g. a pushbutton activation.

NodeID (8 bits)

This field is used to specify the Wake Up destination NodeID.

CC:0084.01.04.11.003 A receiving node MUST assume the indicated NodeID as Wake Up destination. i.e. the Wake Up Notification Command MUST be sent to the NodeID specified in this field.

3.2.46.4 Wake Up Interval Get Command

This command is used to request the Wake Up Interval and destination of a node.

CC:0084.01.05.11.001 The *Wake Up Interval Report Command* MUST be returned in response to this command.

CC:0084.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:0084.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.171: Wake Up Interval Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_INTERVAL_GET (0x05)							

3.2.46.5 Wake Up Interval Report Command

This command is used to advertise the current Wake Up interval and destination.

Table 3.172: Wake Up Interval Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_INTERVAL_REPORT (0x06)							
Seconds 1 (MSB)							
Seconds 2							
Seconds 3 (LSB)							
NodeID							

Seconds (24 bits)

This field is used to advertise the time in seconds between Wake Up periods at the sending node.

CC:0084.01.06.11.001 The first byte MUST be the most significant byte.

CC:0084.01.06.11.002 Values in the range 1..16777215 MUST indicate the time between Wake Up periods.

The value 0 MUST indicate that the receiving node MUST Wake Up when initiated by an event determined by the application e.g. a pushbutton activation.

NodeID (8 bits)

This field is used to advertise the Wake Up destination NodeID configured at the sending node.

CC:0084.01.06.11.003 The sending node MUST send the Wake Up Notification Commands to the advertised NodeID in this field.

3.2.46.6 Wake Up Notification Command

This command allows a node to notify its Wake Up destination that it is awake.

CC:0084.01.07.12.001 The sending node SHOULD start a timer allowing it power down again in case no Wake Up No
CC:0084.01.07.11.001 More Information Command is received. If implemented, the timer MUST comply with the timing requirements defined in the Z-Wave Plus Device Types Specification (Section 8, Section 7).

CC:0084.01.07.13.001 A non-securely included node MAY send this command as broadcast if no Wake Up destination NodeID is configured.

CC:0084.01.07.11.002 A receiving node MUST NOT return a *Wake Up No More Information Command* in response to
CC:0084.01.07.12.002 this command issued via broadcast. Upon receiving this command via broadcast, a receiving node SHOULD configure a relevant Wake Up destination issuing a Wake Up Interval Set Command to the node that issued this command via broadcast.

Table 3.173: Wake Up Notification Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_NOTIFICATION (0x07)							

3.2.46.7 Wake Up No More Information Command

CC:0084.01.08.13.001 This command is used to notify a supporting node that it MAY return to sleep to minimize power consumption.

CC:0084.01.08.12.001 A node receiving the Wake Up No More Information Command SHOULD return a MAC layer Ack frame for the Wake Up No More Information Command before returning to sleep. Not acknowledging the command will cause a number of retransmissions.

CC:0084.01.08.12.002 It is RECOMMENDED that the handling of the Wake Up No More Information Command and the associated time window for MAC layer acknowledgement is left to the protocol layer in order to simplify application design.

Table 3.174: Wake Up No More Information Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_NO_MORE_INFORMATION (0x08)							

3.2.47 Wake Up Command Class, version 2

The Wake Up Command Class version 2 enables read back of the Wake Up Interval capabilities in a node.

3.2.47.1 Compatibility considerations

CC:0084.02.00.21.001 A node supporting Wake Up Command Class, version 2 MUST also support Wake Up Command Class, version 1.

All commands not mentioned in this version remain unchanged from the Wake Up Command Class, version 1.

CC:0084.02.00.22.001 A version 2 supporting node may receive a Wake Up Interval Set outside the range of supported intervals from a version 1 controlling node. A version 2 supporting node SHOULD NOT ignore a Wake Up Interval Set with an invalid time period and set the Wake Up Interval to a supported value.

3.2.47.2 Wake Up Interval Set Command

The Wake Up Interval Set Command is used to configure the wake up interval of a device and the NodeID of the device receiving the Wake Up Notification Command.

Table 3.175: Wake Up Interval Set Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_INTERVAL_SET (0x04)							
Seconds 1 (MSB)							
Seconds 2							
Seconds 3 (LSB)							
NodeID							

All fields not described below remain unchanged from version 1

Seconds (24 bits)

This field is used to specify the time in seconds between Wake Up periods for the receiving node.

CC:0084.02.04.11.001 The first byte MUST be the most significant byte.

CC:0084.02.04.11.002 Values in the range 1..16777215 MUST indicate the time between Wake Up periods.

The value 0 MUST indicate that the receiving node MUST Wake Up when initiated by an event determined by the application e.g. a pushbutton activation.

CC:0084.02.04.11.003 A receiving node MUST accept any interval value within the interval limits advertised by the Wake Up Interval Capabilities Report Command.

CC:0084.02.04.12.001 A receiving node SHOULD NOT ignore this command if this field contains a non-valid value and it SHOULD apply the following:

- CC:0084.02.04.12.002 If the received value is higher than the minimum valid value (advertised in the Wake Up Interval Capabilities Report Command), the receiving node SHOULD round down the value to the next valid value but SHOULD NOT set it to 0 if 0 is the minimum value.
- CC:0084.02.04.12.003 If the value specified in this field is lower than the minimum valid value, the receiving node SHOULD apply the minimum valid value.

CC:0084.02.04.12.004 A receiving node SHOULD accept the value 0, even if the Minimum Wake Up Interval advertised by the Wake Up Interval Capabilities Report Command is larger than 0. If accepted, the value 0 MUST disable the timer-based transmission of Wake Up Notification Commands.

CC:0084.02.04.12.005 The node SHOULD be able to issue a Wake Up Notification in response to some local activation, e.g. a button press.

3.2.47.3 Wake Up Interval Capabilities Get Command

This command is used to request the Wake Up Interval capabilities of a node.

CC:0084.02.09.11.001 The Wake Up Interval Capabilities Report Command MUST be returned in response to this command.

CC:0084.02.09.11.002 This command MUST NOT be issued via multicast addressing.

CC:0084.02.09.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.176: Wake Up Interval Capabilities Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_INTERVAL_CAPABILITIES_GET (0x09)							

3.2.47.4 Wake Up Interval Capabilities Report Command

This command is used to advertise the Wake Up Interval capabilities of a node.

Table 3.177: Wake Up Interval Capabilities Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_INTERVAL_CAPABILITIES_REPORT (0x0A)							
Minimum Wake Up Interval Seconds 1 (MSB)							
Minimum Wake Up Interval Seconds 2							
Minimum Wake Up Interval Seconds 3 (LSB)							
Maximum Wake Up Interval Seconds 1 (MSB)							
Maximum Wake Up Interval Seconds 2							
Maximum Wake Up Interval Seconds 3 (LSB)							
Default Wake Up Interval Seconds 1 (MSB)							
Default Wake Up Interval Seconds 2							
Default Wake Up Interval Seconds 3 (LSB)							
Wake Up Interval Step Seconds 1 (MSB)							
Wake Up Interval Step Seconds 2							
Wake Up Interval Step Seconds 3 (LSB)							

CC:0084.02.0A.11.001 Byte 1 MUST be the most significant byte for all fields in this command.

Minimum Wake Up Interval Seconds (24 bits)

CC:0084.02.0A.11.002 This field is used to advertise the minimum Wake Up Interval supported by the sending node. The field MUST be in the range 0..16777215

CC:0084.02.0A.11.00B The value 0 MUST indicate that the receiving node MUST supports Wake Up when initiated by an event determined by the application e.g. a pushbutton activation.

CC:0084.02.0A.11.00C Values in the range 1..16777215 MUST indicate Minimum supported Wake Up interval in seconds

Maximum Wake Up Interval Seconds (24 bits)

This field is used to advertise the maximum Wake Up Interval supported by the sending node. The
CC:0084.02.0A.11.003 field MUST comply with Table 3.178.

Table 3.178: Wake Up Interval Capabilities Report::Maximum Wake Up Interval Seconds Encoding

Decimal	Description
0	This value is used to indicate that there is no minimum / maximum / default wake up interval. In this case, the sending node is activated by e.g. user interaction in form of a button press. CC:0084.02.0A.11.004 If this field is set to 0, the Minimum and Default Wake Up Interval fields MUST also be 0.
<min interval> ..16777215	Values in this range indicate the maximum wake up interval in seconds supported by the sending node. CC:0084.02.0A.11.005 This field MUST NOT be set to a lower value than the Minimum Wake Up Interval Seconds value. CC:0084.02.0A.13.001 This field MAY be set to the same value as the Minimum Wake Up Interval Seconds value, which means the sending node only supports one value.

Default Wake Up Interval Seconds (24 bits)

This field is used to advertise the default Wake Up Interval value for the sending node.

CC:0084.02.0A.12.006 This field MUST be set to a value included in the range defined by the Minimum Wake Up Interval and the Maximum Wake Up Interval

CC:0084.02.0A.12.001 The Default Wake Up Interval SHOULD be 0 if the Minimum Wake Up Interval is 0.

Wake Up Interval Step Seconds (24 bits)

This field is used to advertise the resolution of valid Wake Up Intervals values for the sending node.
CC:0084.02.0A.11.007 The field MUST comply with Table 3.179.

Table 3.179: Wake Up Interval Capabilities Report::Wake Up Interval Step Seconds Encoding

Decimal	Description
0	This value is used to indicate that no interval steps are possible. The battery-operated device only supports the minimum and maximum Wake Up Interval values. CC:0084.02.0A.11.008 This field MUST be set to 0 if the maximum and the minimum interval are equal.
1..16777215	Values in this range indicate the Wake Up Interval step in seconds supported by the sending node. CC:0084.02.0A.11.009 This field's value MUST NOT exceed the difference between the Minimum and Maximum Wake Up Intervals. CC:0084.02.0A.12.002 This field SHOULD have a value so that the difference between the maximum and minimum Wake Up Interval is a multiple of this field's value. *Examples*: CC:0084.02.0A.11.00A If a device has minimum wake up interval of 5 minutes (300 seconds) and a maximum wake up interval of 10 minutes (600 seconds), the wake up interval step MUST NOT exceed 5 minutes (300 seconds) as this would be larger than the difference of the minimum and maximum interval. A Wake Up Interval Step of 100 seconds indicates that the nodes supports the following Wake Up Intervals : 300 seconds 400 seconds 500 seconds 600 seconds

3.2.48 Wake Up Command Class, version 3

The Wake Up Command Class, version 3 introduces the capability to request a wake up on demand to supporting nodes during unsolicited communication.

CC:0084.03.00.21.001 The Wake Up Command Class, version 3 is backwards compatible with the Wake Up Command Class, version 2. Fields and commands not described in this version MUST remain unchanged from version 2.

3.2.48.1 Wake Up Interval Capabilities Report Command

This command is used to advertise the Wake Up capabilities of a supporting node.

Table 3.180: Wake Up Interval Capabilities Report Command, version 3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP (0x84)							
Command = WAKE_UP_INTERVAL_CAPABILITIES_REPORT (0x0A)							
Minimum Wake Up Interval Seconds 1 (MSB)							
Minimum Wake Up Interval Seconds 2							
Minimum Wake Up Interval Seconds 3 (LSB)							
Maximum Wake Up Interval Seconds 1 (MSB)							
Maximum Wake Up Interval Seconds 2							
Maximum Wake Up Interval Seconds 3 (LSB)							
Default Wake Up Interval Seconds 1 (MSB)							
Default Wake Up Interval Seconds 2							
Default Wake Up Interval Seconds 3 (LSB)							
Wake Up Interval Step Seconds 1 (MSB)							
Wake Up Interval Step Seconds 2							
Wake Up Interval Step Seconds 3 (LSB)							
Reserved							Wake Up on Demand Support

Wake Up On Demand support (1 bit)

This field is used to advertise if the supporting node supports the Wake Up On Demand functionality.

CC:0084.03.0A.11.001 If this field is set to 0, the supporting node MUST NOT support the Wake Up On Demand functionality.

If this field is set to 1, the supporting node MUST support the Wake Up On Demand functionality.

CC:0084.03.0A.11.002 A node supporting the Wake Up On Demand functionality MUST support the Supervision Command Class, version 2.

CC:0084.03.0A.11.003 A node supporting the Wake Up On Demand functionality MUST return a Wake Up Notification Command if the Wake Up destination node issued a Supervision Report Command with the Wake Up Request bit set to 1.

CC:0084.03.0A.11.004 A node supporting the Wake Up On Demand functionality MUST ignore the *Wake Up Request* field if the Supervision Report is not issued by the Wake Up Destination.

CC:0084.03.0A.11.005 A node issuing a Wake Up Notification Command after a Wake Up Request MUST restart its Wake Up timer for the next Wake Up period.

3.2.48.1.1 Wake Up On Demand functionality

If the Wake Up destination has important messages to be transmitted to the Wake Up node, it can use the Wake Up Request bit in the Supervision report when the sleeping node uses Supervision encapsulation. An example frame flows is shown in Figure 3.30.

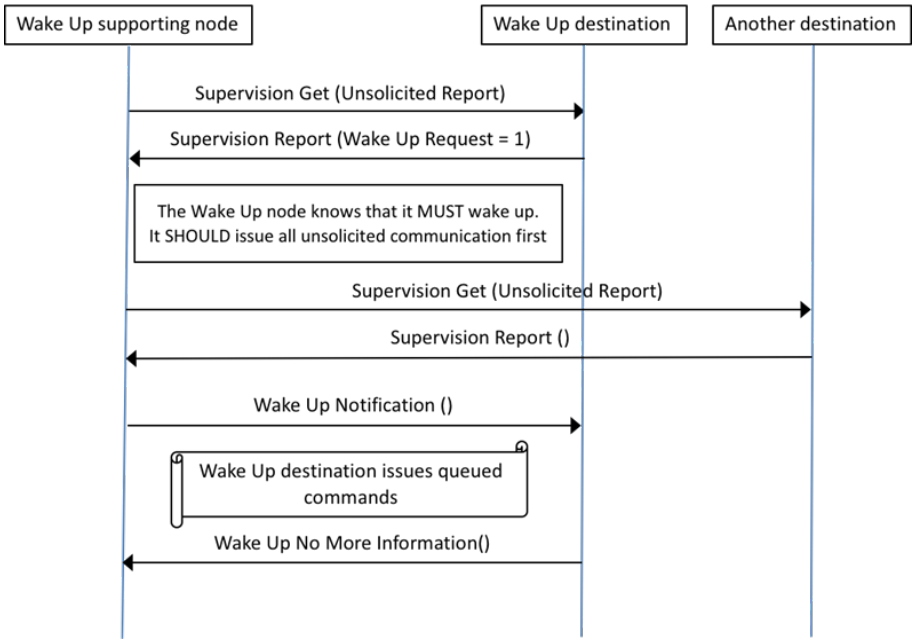


Figure 3.30: Wake Up Destination Initiates an On Demand Wake Up

3.2.49 Z/IP Naming and Location Command Class, version 1

The Z/IP Naming and Location Command Class is used to manage the Name and a Location text strings of a Z/IP resource.

3.2.49.1 Compatibility Considerations

A Z/IP resource is identified by an IPv6 address and an End Point ID. Capable Z/IP nodes SHOULD store the naming and location information locally in non-volatile storage. In addition, a Z/IP Resource Directory (RD) MAY store the naming and location of all Z/IP Resources; also sleeping resources. The Z/IP RD manages node properties retrieved from actual nodes as well as extended information which is not always available in classic Z-Wave nodes.

Information managed by the Z/IP RD may be announced to Z/IP Clients via service discovery mechanisms such as mDNS.

Commands defined in this command class MUST be encapsulated in Z/IP Packets.

A client modifying the name and/or location information MUST ensure that the combined length of the Z/IP Name and Z/IP Location is 62 bytes or less in order to comply with DNS-SD limitations.

3.2.49.2 Z/IP Name Set Command

This command is used to set the name of a Z/IP Resource.

Table 3.181: Z/IP Name Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_NAMING (0x68)							
Command = ZIP_NAMING_NAME_SET (0x01)							
Name 1							
...							
Name N							

Name (N bytes)

This field is used to specify the Name portion of the resource name.

The combined Name and Location strings MUST NOT be longer than 62 bytes.

The Name String MUST be UTF-8 encoded. Since a UTF-8 character may require two or more bytes, the maximum length of a name string depends on the composition of the string.

The Name String MUST NOT contain any appended termination characters. The number of bytes MUST be determined from the command length.

The name MUST NOT contain the dot (also known as period) character ".".

The escape sequence "\." (backslash followed by dot) MAY be used for encoding the dot character.

A user interface SHOULD allow users to enter names that contain dot characters.

If a dot character is entered, the user interface MUST replace the dot character with the escape sequence "\."before issuing this command.

The name MUST NOT contain the underscore character "_".

The name MUST NOT end with the dash character "-".

Node names MUST be case insensitive.

3.2.49.3 Z/IP Name Get Command

This command is used to request the name from a Z/IP Resource

The Z/IP Name Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.182: Z/IP Name Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_NAMING (0x68)							
Command = ZIP_NAMING_NAME_GET (0x02)							

3.2.49.4 Z/IP Name Report Command

This command is used to advertise the name of a Z/IP Resource.

Table 3.183: Z/IP Name Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_NAMING (0x68)							
Command = ZIP_NAMING_NAME_REPORT (0x03)							
Name 1							
...							
Name N							

Name (N bytes) Refer to Section 3.2.49.2 Z/IP Name Set Command.

It may be attractive to use dots in names. The escape sequence "\." (backslash followed by dot) MAY be used for encoding the dot character in names.

A receiving device SHOULD display the escape sequence "\." as "." in user interfaces facing end users.

3.2.49.5 Z/IP Location Set Command

This command is used to set the location of a Z/IP Resource.

Table 3.184: Z/IP Location Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_NAMING (0x68)							
Command = ZIP_NAMING_LOCATION_SET (0x04)							
Location 1							
...							
Location N							

Location (N bytes)

This field is used to specify the Location portion of the resource name.

The combined Name and Location strings MUST NOT be longer than 62 bytes.

The Location string MUST be UTF-8 encoded. Since a UTF-8 character may require two or more bytes, the maximum length of a location string depends on the composition of the string.

The Location string MUST NOT contain any appended termination characters. The number of bytes MUST be determined from the message length.

The location string MAY contain the dot (also known as the period) character ".".

A user interface MUST NOT replace the dot character with the escape sequence "\."before issuing this Command.

The location string MUST NOT contain the underscore character "_".

Each location sub-string (separated by the dot character ".") MUST NOT end with the dash character "-".

Node locations MUST be case insensitive.

3.2.49.6 Z/IP Location Get Command

This command is used to request the location from a Z/IP Resource

The Z/IP Location Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.185: Z/IP Location Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_NAMING (0x68)							
Command = ZIP_NAMING_LOCATION_GET (0x05)							

3.2.49.7 Z/IP Location Report Command

This command is used to advertise the location of a Z/IP Resource.

Table 3.186: Z/IP Location Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_NAMING (0x68)							
Command = ZIP_NAMING_LOCATION_REPORT (0x06)							
Location 1							
...							
Location N							

Location (N bytes)

Refer to Section 3.2.49.5 Z/IP Location Set Command.

The escape sequence "\." (backslash followed by dot) MAY be used for encoding the dot character in locations.

A receiving node SHOULD display the escape sequence "\." as "." in user interfaces facing end users.

3.2.50 Z-Wave Plus Info Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS VERSION HAS BEEN OBSOLETED

CC:005E.01.00.11.001 New implementations MUST use the *Z-Wave Plus Info Command Class, version 2*.

3.2.51 Z-Wave Plus Info Command Class, version 2

The Z-Wave Plus Info Command Class is used to differentiate between Z-Wave Plus, Z-Wave for IP and Z-Wave devices. Furthermore this command class provides additional information about the Z-Wave Plus device in question.

The Z-Wave Plus Info Command Class defines two icon types. Icon types allow for a meaningful, homogeneous representation in user and installer Graphical User Interfaces (GUI), respectively. A
CC:005E.02.00.11.001 Z-Wave Plus product MUST specify valid icon types. Icon types do not affect the operation of a product or how it is certified. The actual graphical appearance of icons is out of scope of this specification. Any app may define its own icon library mapping to the Icon Type identifiers.

3.2.51.1 Compatibility Considerations

3.2.51.1.1 Node Information Frame (NIF)

CC:005E.02.00.21.001 A supporting node MUST always advertise the Z-Wave Plus Info Command Class in its NIF, regardless of the inclusion status and security bootstrapping outcome.

CC:005E.02.00.21.002 The Z-Wave Plus Info Command Class MUST be advertised as the first supported command class in Node Information Frame (NIF).

3.2.51.2 Multi Channel considerations

A Multi Channel device implements a Root Device and a number of Multi Channel End Points.

CC:005E.02.00.11.002 The Z-Wave Plus Info Command Class MUST be supported for each End Point in order to advertise individual icons for each End Point.

This means that the Z-Wave Plus Version, Role Type and Node Type information is advertised in a redundant fashion. The advertised Z-Wave Plus Version, Role Type and Node Type information
CC:005E.02.00.11.003 values MUST be identical for the Root Device and all Multi Channel End Points.

3.2.51.3 Z-Wave Plus Info Get Command

The Z-Wave Plus Info Get Command is used to get additional information of the Z-Wave Plus device in question.

CC:005E.02.01.11.001 The Z-Wave Plus Info Report Command MUST be returned in response to this command.

CC:005E.02.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:005E.02.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 3.187: Z-Wave Plus Info Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZWAVEPLUS_INFO (0x5E)							
Command = ZWAVEPLUS_INFO_GET (0x01)							

3.2.51.4 Z-Wave Plus Info Report Command

The Z-Wave Plus Info Report Command is used to report version of Z-Wave Plus framework used and additional information of the Z-Wave Plus device in question.

Table 3.188: Z-Wave Plus Info Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZWAVEPLUS_INFO (0x5E)							
Command = ZWAVEPLUS_INFO_GET (0x01)							
Z-Wave Plus Version							
Role Type							
Node Type							
Installer Icon Type MSB							
Installer Icon Type LSB							
User Icon Type MSB							
User Icon Type LSB							

Z-Wave Plus Version (8 bits)

The Z-Wave Plus Version field enables a future revision of the Z-Wave Plus framework where it is necessary to distinguish it from the previous frameworks.

CC:005E.02.02.11.006 This field MUST be set to 1 if the sending node is compliant with Z-Wave Plus [34].

This field MUST be set to 2 if the sending node is compliant with *Device Type v2 Specification*.

Role Type (8 bits)

The Role Type field indicates the role the Z-Wave Plus device in question possess in the network and functionalities supported. The full functionality is determined in conjunction with the Device Type.

Node Type (8 bits)

The Node Type field indicates the type of node the Z-Wave Plus device in question possess in the network.

The table below shows the current list of Node Types:

Table 3.189: Node Type Identifiers

Value	Identifier	Node Types
0x00	NODE_TYPE_ZWAVEPLUS_NODE	Z-Wave Plus node
0x02	NODE_TYPE_ZWAVE-PLUS_FOR_IP_GATEWAY	Z-Wave Plus for IP gateway

CC:005E.02.02.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Z-Wave Plus for IP gateway

The device provides access to and from IP networks and allows IP hosts to discover resources in the Z-Wave network via Z/IP Discovery.

CC:005E.02.02.11.003 Two IP services MUST be available to represent Z-Wave resources in the IP domain: ICMP Echo (Ping) and UDP port 4123 (Z-Wave control protocol).

The device is typically a stand-alone device based on a residential IPv6 router. The Z-Wave interface is presented as an IP network interface to external clients but all IP communication to classic Z-Wave nodes is terminated by the gateway and forwarded in classic Z-Wave frames.

Installer Icon Type (16 bits)

The Installer Icon Type field indicates the icon to use in Graphical User Interfaces for network management, e.g. in a floor plan. Installer Icons provide a finer granularity than user icons, e.g. a light

on/off resource may be a built-in wall outlet, a plug-in module or an entire power strip. A sensor may be a single sensor icon or multiple sensors in one casing.

For further details, refer to the User Icon Type section below.

User Icon Type (16 bits)

The User Icon Type field indicates the icon to use in Graphical User Interfaces for end users. User Icons provide a basic granularity, e.g. a light resource is always shown as a light bulb whether the physical device is a built-in wall outlet, a plug-in module or the output of a power strip.

CC:005E.02.02.11.004 In case of multi-endpoint devices, e.g. power strips or multi-sensors, the info command class MUST
CC:005E.02.02.13.001 be supported and Icon Types MUST be specified for each endpoint.

The User Icon Type MAY differ for individual endpoints.

Icon types are defined by the Z-Wave Alliance [33]. The 16 bit identifier values defined by [33] allows the GUI designer to know how a given product would like to be represented using resources from the GUI designer's own favorite icon library. The icon graphics found in [33] are only suggested graphics.
CC:005E.02.02.12.001 Output device icons SHOULD be tailored to the actual geographical region.

CC:005E.02.02.12.002 If a GUI does not know the specific icon type, it SHOULD use the generic icon type as a fallback
CC:005E.02.02.11.005 alternative. The generic icon type MUST be derived by setting the least significant 8 bits of the icon type identifier to zero.

4 Transport-Encapsulation Command Classes

4.1 Command Class Overview

General Command Class overview and rules are described in Section 2 and are valid for the Command Classes presented in this document.

The following subsections present the additional considerations relating to the Encapsulation Command Classes.

4.1.1 Encapsulation Command Classes Support/Control

CC:0000.00.00.11.01A A node MUST advertise support for Multi Channel Command Class only if it implements End Points.

CC:0000.00.00.11.01B A node able to communicate using the Multi Channel encapsulation but implementing no End Point MUST NOT advertise support for the Multi Channel Command Class.

CC:0000.00.00.11.01C A node able to receive commands encapsulated with the Transport Service, Security (S0/S2) or CRC-16 Command Class MUST advertise the respective Command Class as supported and MUST also be able to send commands encapsulated with the respective Command Class.

The Command Class support/control definition presented in Section 2 applies to the Supervision Command Class.

4.1.2 Node Information Frame

For the NIF definition, refer to Section 2.

The NIF represents the Root Device's Command Class capabilities when using no Security encapsulation.

Multi Channel Root Devices MUST advertise their non-secure capabilities via the NIF.

Multi Channel End Points MUST advertise their non-secure capabilities via the Multi Channel Capability Report Command.

Security bootstrapped nodes MUST advertise their capabilities using security encapsulation (for both Root Devices and End Points) via the Security Commands Supported Report Command or the Security 2 Commands Supported Report Command.

4.1.3 Multi Channel Overview

The section presents the concept of Multi Channel End Points. The concept ties closely to command classes such as Multi Channel Command Class, Multi Channel Association Command Class, Association Group Information Command Class as well as the Z-Wave Plus Icon Type. Multi Channel functionality may be used for controlling as well as for supporting devices.

4.1.3.1 Terminology

A Z-Wave node is conceptually an **application resource** in a plastic box communicating via a Z-Wave radio. Composite devices pack multiple application resources in the same plastic box, thus sharing the same Z-Wave radio. Application resources can always be addressed individually.

Within a network, a Z-Wave node is identified by its NodeID. The NodeID represents the plastic box and the radio.

Multi-resource devices are organized as Multi Channel **End Points**. Each application resource is identified by its own unique End Point. The plastic box itself is referred to as the **Root Device**.

Multi Channel Encapsulation allows a sending node to specify a **source End Point** and a **destination End Point**. Further, the destination End Point may be structured as a **multicast**

mask, targeting up to 7 End Points by one command. Multi Channel Encapsulation is used for transmission from one End Point to another, from an End Point to a Root Device as well as from a Root Device to an End Point.

Multi Channel Encapsulation is not used between Root Devices.

An **Aggregated End Point** implements a function which relates to multiple individual End Points. An Aggregated End Point is addressed just as an individual End Point.

One example of an Aggregated End Point is a common power meter of a power strip which measures the total power consumption of all End Points.

The aggregation of End Points should not be confused with multicast addressing. Sending a Meter Reset command via multi-End Point addressing to all individual End Points causes all individual End Points to reset their individual meters. Sending a Meter Reset command to the Aggregated End Point causes the common power meter to be reset.

Bridging devices may provide connectivity to other technologies than Z-Wave via dynamic End Points. Dynamic End Points may be **created**, **changed** or **removed**.

A controlling device MAY use Multi Channel encapsulation to communicate with Multi Channel End Points in other devices. If such a controlling device does not implement any End Points, the device MUST NOT advertise the Multi Channel Command Class in the Node Information Frame.

One may create a **Multi Channel Association** to allow an End Point to control another End Point. The End Point may also control a Root Device. Likewise, a Multi Channel Association may be created to allow a Root Device to control an End Point.

The **Association Group Information** advertises the association capabilities of each Association Group in each End Point as well as in the Root Device.

4.1.3.2 Backwards compatibility

The Multi Channel concept provides a toolbox for sub-addressing. Any controlling device should implement the functionality required to interact with supporting Multi Channel devices.

Legacy devices only understand the concept of the Root Device. Therefore, a Multi Channel device providing multiple application resources also provides a meaningful subset of the application functionality via the Root Device on behalf of one or more End Points.

One example is a composite temperature and humidity sensor, which exposes the temperature sensor functionality via the Root Device as if the device was a stand-alone temperature sensor.

Another example is a 5-output power strip implemented as 5 Multi Channel End Points. The Root Device exposes one Binary Switch resource but a command to the Root Device spawns internal control commands to all outputs, so the Root Device acts as a main switch.

It is seen that the mapping of application functionality to the Root Device depends heavily on the actual product feature set. However, a few principles apply:

1. The Root Device provides access to the primary functionality of the actual product.
2. The Root Device of a Multi Channel device does not implement any application functionality that cannot be reached via End Points.
3. The Root Device of a Multi Channel always presents functionality, which can also be reached via End Point 1. If it makes sense in the actual product, a Root Device command may affect more End Points than End Point 1.

As it is optional how to forward commands from the Root Device to multiple End Points, one cannot be sure that a command to the Root Device will target all End Points. The Multi Channel multicast feature may be used to send a Set command to the first 7 End Points. Alternatively, one may communicate to each individual End Point. This works for Set as well as Get commands.

4.1.3.3 GUI presentation

The Root Device always advertises application functionality, even when the application functionality is actually provided by forwarding commands to one or more End Points of the device.

Management tools may have a need to present expanded views of Multi Channel devices, e.g. in the floor plan view of a smart home deployment. By ignoring application-style Command Classes supported by one or more End Points, the Root Device can be presented the way it really works.

4.1.3.4 Applicability examples

- A gateway may target a destination End Point to control the individual output of a power strip
- A gateway may create a Multi Channel Association from the Root Device Lifeline association group of a two-channel indoor/outdoor temperature sensor to receive sensor reports. The Multi Channel encapsulation source End Point allows the gateway to distinguish indoor readings from outdoor readings.
- One End Point of a dual End Point indoor/outdoor temperature sensor may have a (non-Multi Channel) association created to control the Root Device of an air conditioner on basis of the measured indoor temperature.

4.1.3.5 Encapsulation order overview

Command Class encapsulation MUST be applied in the following order:

1. Encapsulated Command Class (payload), e.g Basic Set
2. Multi Command
3. Supervision
4. Multi Channel
5. Any one of the following combinations:
   a. Security (S0 or S2) followed by transport service
   b. Transport Service
   c. Security (S0 or S2)
   d. CRC-16

The encapsulation order is also shown in Figure 4.1.

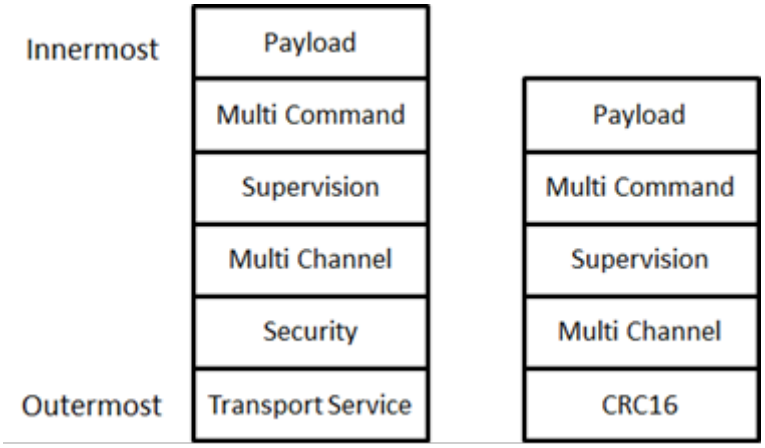


Figure 4.1: Encapsulation overview

An exception to Figure 4.1 is made when querying Multi Channel End Points about their secure capabilities. In this case, the S0/S2 Security Commands Supported Report Command is carried in a Multi Channel encapsulation Command, thus, the encapsulation order is:

Security (Multi Channel (Security Commands Supported Get Command))

CC:0000.00.00.11.021 Responses to a given frame MUST be carried out using the same encapsulation or lack of encapsulation as it was received, unless specified otherwise in the Command Class specification.

The Transport Service Command Class and Multi Command Command Class are exceptions to the above rule.

CC:0000.00.00.11.022 The transport service MUST be used only if the payload does not fit in the Z-Wave MAC frame size.

CC:0000.00.00.12.006 The Multi Command Command Class is optional to use and SHOULD be used only if several commands are queued for transmission.

4.2 Command Class Definitions

The following subchapters contain definitions of Transport-Encapsulation Command Classes.

4.2.1 CRC-16 Encapsulation Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this Command Class, but it is RECOMMENDED that new implementations use the Security 2 Command Class only.

Note: some Device Types are still REQUIRED to support this Command Class

The CRC-16 Encapsulation Command Class is used to encapsulate a command with an additional CRC-16 checksum to ensure integrity of the payload. The purpose for this command class is to ensure a higher integrity level of payloads carrying important data using 9.6/40kbps communication, in case the LRC checksum (8 bits) provided on protocol level is not sufficient to ensure integrity.

4.2.1.1 Compatibility Considerations

4.2.1.1.1 Node Information Frame (NIF)

CC:0056.01.00.21.001 A supporting node MUST always advertise the CRC-16 Command Class in its NIF, regardless of the inclusion status and security bootstrapping outcome.

CC:0056.01.00.21.002 A supporting node MUST NOT advertise the CRC-16 Command Class in its S0/S2 Commands Supported Report.

4.2.1.1.2 Control and support

CC:0056.01.00.21.003 The CRC-16 Encapsulation Command Class MUST NOT be encapsulated by any other Command Class.

Alternatives to using CRC-16 Encapsulation Command Class are:

- The Security (S0) or Security 2 (S2) Command Class to ensure privacy and integrity of data.
- The 100kbps communication speed already provides a CRC-16 checksum at the protocol level.

CC:0056.01.00.21.004 A node supporting the CRC-16 Encapsulation Command Class may receive a combination of encapsulated and normal non-encapsulated requests and the response MUST be as follows:

CC:0056.01.00.21.005 a. If the request is sent encapsulated, the response MUST be returned encapsulated.

CC:0056.01.00.21.006 b. If the request is sent non-encapsulated, the response MUST be sent non-encapsulated.

CC:0056.01.00.21.007 A node supporting the CRC-16 Encapsulation Command Class MUST be able to receive and interpret the encapsulated version of all the command classes that it lists in the NIF.

CC:0056.01.00.21.008 Before sending an encapsulated command, the controlling node MUST ensure that the destination supports the CRC-16 Encapsulation Command Class.

4.2.1.2 CRC-16 Encapsulated Command

The CRC-16 Encapsulation Command is used to encapsulate a command with an additional checksum to ensure integrity of the payload. Be aware of the payload limitations with respect to a routed single cast frame.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CRC_16_ENCAP (0x56)							
Command = CRC_16_ENCAP (0x01)							
Command Class (1 or 2 bytes)							
Command							
Data 1							
...							
Data N							
Checksum 1							
Checksum 2							

Command Class (8 bits or 16 bits)

CC:0056.01.01.11.001 This field MUST specify the Command Class identifier of the encapsulated Command. This field MUST carry a normal Command Class (8 bits) or an Extended Command Class (16 bits).

Command (8 bits)

This field MUST specify the Command identifier of the encapsulated command.

Data (N bytes)

CC:0056.01.01.11.003 This field MUST carry the payload of the encapsulated command.

Checksum (16 bits)

This field is used to advertise the checksum of the data contained in the actual command.

CC:0056.01.01.11.004 The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). Refer to *CRC-CCITT Source Code* for the CRC_CCITT source code.

CC:0056.01.01.11.005 The checksum data MUST be built by taking all bytes starting from the CRC16 Command Class identifier (COMMAND_CLASS_CRC_16_ENCAP) until the last byte of the Data field.

CC:0056.01.01.11.006 The first byte of this field MUST be the most significant byte. For example, a node sending a Basic Get Command encapsulated with CRC-16 MUST be according to Table 4.1.

Table 4.1: Basic Set Command with CRC-16 Encapsulation

	CRC-16 Command fields	Value	Description
1	COMMAND_CLASS_CRC_16_ENCAP	0x56	CRC-16 Command Class identifier
2	CRC_16_ENCAP	0x01	CRC-16 Encapsulation Command id
3	COMMAND_CLASS_BASIC	0x20	Basic Command Class identifier
4	BASIC_GET	0x02	Basic Get Command
5	Checksum 1	0x4D	MSB for CRC-16 checksum
6	Checksum 2	0x26	LSB for CRC-16 checksum

4.2.2 Multi Channel Command Class, version 3 [OBSOLETED]

> **Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED**
>
> New implementations MUST NOT support this Command Class.
>
> New implemenetations MUST use the *Multi Channel Command Class, version 4*, or newer instead.

The Multi Channel command class is used to address one or more End Points in a Multi Channel device.

Refer to Section 4.1.3 for an introduction to the Multi Channel concept.

4.2.2.1 Compatibility Considerations

CC:0060.03.00.23.001 A Multi Channel device MAY implement from 1 to 127 End Points.

CC:0060.03.00.21.002 A Multi Channel device MUST implement all application functionality in End Points.

CC:0060.03.00.21.003 End Point 1 MUST implement the primary application functionality of the actual Multi Channel device.

CC:0060.03.00.23.002 Additional End Points MAY implement an identical functionality; as an example, a power strip may implement five End Points (one for each outlet) with identical functionality.

CC:0060.03.00.21.004 For backwards compatibility, the Root Device MUST mirror the application functionality of End Point 1.

CC:0060.03.00.23.003 Further, the Root Device MAY mirror the application functionalities of additional End Points. As an example, Basic Off and On commands for the Root Device may control all outlets of a power strip with five outlets.

CC:0060.03.00.21.005 With the exception of dynamic End Points, the Root Device MUST advertise all End Point functionality which is mirrored by the Root Device.

CC:0060.03.00.22.001 The Root Device SHOULD NOT advertise the application functionality of any dynamic End Point.

CC:0060.03.00.21.006 The Root Device of a Multi Channel device MUST only advertise application functionality that can be reached via one or more End Points. However, if the node supports the following Command Classes,
CC:0060.03.00.22.003 they SHOULD only be supported and advertised by the Root Device:

- Central Scene Command Class
- Configuration Command Class
- Anti-Theft Command Class
- Anti-Theft Unlock Command Class
- Clock Command Classes
- Geographic Location Command Class

CC:0060.03.00.21.007 It MUST NOT be possible to limit the functionality, or enable non-compliant behavior of any End Point in the device by sending a command to the Root Device.

4.2.2.1.1 Node information Frame (NIF)

CC:0060.03.00.21.001 A node supporting this Command Class MUST set the Optional Functionality bit in its NIF.

CC:0060.03.00.21.00C Multi Channel Root Devices MUST advertise their non-secure capabilities via the NIF.

CC:0060.03.00.21.00D Multi Channel End Points MUST advertise their non-secure capabilities via the Multi Channel Capability Report Command.

CC:0060.03.00.21.00E Security bootstrapped nodes MUST advertise their capabilities using security encapsulation (for both Root Devices and End Points) via the S0 Security Commands Supported Report Command or the S2 Security 2 Commands Supported Report Command.

Secure End Point capabilities are therefore requested using the following encapsulation:

- S0/S2 Encapsulation
- Multi Channel Encapsulation
- S0/S2 Commands Supported Get/Report Commands

4.2.2.1.2 Dynamic End Point considerations

DYNAMIC END POINTS HAVE BEEN OBSOLETED

Dynamic End Points have been obsoleted. New implementation MUST use dynamic capabilities.

A node may add and/or remove End Points based on a user action, such as changing configuration parameters or the physical addition/removal of a module. When this happen, the node SHOULD treat such behavior as dynamic capabilities [For details, refer to *Dynamic Capabilities and Node Discovery*] and notify the End Point modification capabilities to the lifeline destinations.

CC:0060.03.00.23.004 An End Point MAY be dynamic. Dynamic End points are intended End Points able to change their capabilities or that can be added and removed from a Multi Channel device.

CC:0060.03.00.22.002 When creating a new dynamic End Point, it SHOULD be assigned an End Point identifier which has not been used recently to allow applications to discover the removal of a dynamic End Point.

CC:0060.03.00.21.009 When a dynamic End Point is removed, all other End Points MUST maintain their current End Point identifiers.

CC:0060.03.00.23.005 A supporting device implementing dynamic End Points MAY advertise the creation, change or removal of a dynamic End Point via the Root Device Lifeline association group by issuing a Multi Channel Capability Report.

CC:0060.03.00.21.00A A node MUST NOT advertise changes to dynamic End Points via broadcast transmission. A receiving node MUST ignore such broadcasted advertisements.

CC:0060.03.00.21.00B After advertising the removal, a removed dynamic End Point MUST ignore all commands.

4.2.2.2 Interoperability considerations

CC:0060.03.00.53.001 A controlling node MAY use Multi Channel Encapsulation Command to communicate with Multi Channel End Points in other nodes. If such a controlling node does not implement any End Points,
CC:0060.03.00.51.001 it MUST NOT advertise the Multi Channel Command Class in its Node Information Frame (NIF) or S0/S2 Commands Supported Report Command.

An example of such device would be a controller or gateway which can control End Points in other supporting nodes via commands from the Root Device of the gateway. Likewise, the Root Device of the gateway may receive unsolicited commands from other nodes End Points.

4.2.2.3 Multi Channel End Point Get Command

This command is used to query the number of End Points implemented by the receiving node.

CC:0060.03.07.11.001 The Multi Channel End Point Report Command MUST be returned in response to this command.

CC:0060.03.07.51.001 This command MUST NOT be issued via multicast addressing.

CC:0060.03.07.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_END_POINT_GET (0x07)							

4.2.2.4 Multi Channel End Point Report Command

This command is used to advertise the number of End Points implemented by the sending node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_END_POINT_REPORT (0x08)							
Dynamic	Identical	Res					
Res	End Points						

Dynamic (1 bit)

This field is used to advertise if the node implements a dynamic number of End Points.

CC:0060.03.08.11.001 The value 1 MUST be used to indicate that the number of End Points is dynamic.

The value 0 MUST be used to indicate that the number of End Points is static.

Identical (1 bit)

This field is used to advertise if all end points have identical capabilities

CC:0060.03.08.11.002 This bit MUST be set to 1 if all End Points advertise the same Generic and Specific Device Class and support the same Command Classes.

CC:0060.03.08.11.003 This bit MUST be set to 0 if End Points do not advertise the same Device Class or Command Class information.

Res

CC:0060.03.08.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Points (7 bits)

This field is used to advertise the number of End Points implemented by the sending node.

CC:0060.03.08.11.005 This field MUST be in the range 1..127.

CC:0060.03.08.11.006 If the sending node implements dynamic End Points, this field MUST advertise the number of End
CC:0060.03.08.13.001 Points currently instantiated by the node. A dynamic End Point MAY be assigned any End Point identifier in the range 2..127.

4.2.2.5 Multi Channel Capability Get Command

This command is used to query the non-secure Command Class capabilities of an End Point.

CC:0060.03.09.11.001 The Multi Channel Capability Report Command MUST be returned in response to this command unless it is to be ignored.

CC:0060.03.09.51.001 This command MUST NOT be issued via multicast addressing.

CC:0060.03.09.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_CAPABILITY_GET (0x09)							
Res	End Point						

Res

CC:0060.03.09.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Point (7 bits)

CC:0060.03.09.11.004 This field MUST specify the End Point for which the capabilities MUST be returned.

CC:0060.03.09.11.005 If the specified End Point does not exist, this command MUST be ignored.

If the specified End Point represents a removed dynamic End Point, this command MUST be ignored.

4.2.2.6 Multi Channel Capability Report Command

This command is used to advertise the Generic and Specific Device Class and the supported command classes of an End Point.

CC:0060.03.0A.11.002 When advertising the removal of a dynamic End Point, this command MUST carry the following values:

- Dynamic MUST be set to 1
- End Point MUST be set to the actual End Point identifier
- Generic Device Class MUST be set to 0xFF (GENERIC_TYPE_NON_INTEROPERABLE)
- Specific Device Class MUST be set to 0x00 (SPECIFIC_TYPE_NOT_USED)
- CC:0060.03.0A.11.003 The Command Class field MUST be omitted.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_CAPABILITY_REPORT (0x0A)							
Dynamic	End Point						
Generic Device Class							
Specific Device Class							
Command Class 1							
...							
Command Class N							

Dynamic (1 bit)

This field is used to advertise if the advertised End Point is dynamic.

CC:0060.03.0A.11.001 This field MUST be set to 1 if this is a dynamic End Point.

This field MUST be set to 0 to indicate that this is a static End Point.

End Point (7 bits)

CC:0060.03.0A.11.004 This field MUST advertise the actual End Point for which the capabilities are advertised.

Generic Device Class (8 bits)

CC:0060.03.0A.11.005 This field MUST carry the Generic Device Class of the advertised End Point. For a detailed description of all available Generic Device Classes, refer to [34] and Section 7 for Z-Wave Plus nodes.

Specific Device Class (8 bits)

CC:0060.03.0A.11.006 This field MUST carry the Specific Device Class of the advertised End Point. For a detailed description of all available Specific Device Classes, refer to [34] and Section 7 for Z-Wave Plus nodes.

Command Class (N bytes)

This field is used to advertise the non-secure supported Command Classes by the actual End Point.

CC:0060.03.0A.11.007 This field MUST be omitted if the advertised End Point does not exist or have been removed. The number of Command Class bytes MUST be determined from the length of the frame.

CC:0060.03.0A.11.009 This field MUST represent the capabilities of an End Point with no security encapsulation.

CC:0060.03.0A.11.008 The Multi Channel Command Class MUST NOT be advertised in this list.

CC:0060.03.0A.11.00A Non-secure End Point capabilities MUST also be supported securely and MUST also be advertised in the S0/S2 Commands Supported Report Commands unless they are encapsulated outside Security or Security themselves.

CC:0060.03.0A.11.00B Nodes supporting S0 MUST advertise S0 as supported for each End Point that can be addressed with S0 encapsulation

CC:0060.03.0A.11.00C Nodes supporting S2 MUST support addressing every End Point with S2 encapsulation and MAY explicitly list S2 in the non-secure End Point capabilities.

4.2.2.7 Multi Channel End Point Find Command

This command is used to request End Points having a specific Generic or Specific Device Class in End Points.

CC:0060.03.0B.11.001 The Multi Channel End Point Find Report Command MUST be returned in response to this command.

CC:0060.03.0B.51.001 This command MUST NOT be issued via multicast addressing.

CC:0060.03.0B.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_END_POINT_FIND (0x0B)							
Generic Device Class							
Specific Device Class							

Generic Device Class (8 bits)

CC:0060.03.0B.11.003 This field MUST indicate the receiving node to return the list of End Points having the specified Generic Device Class.

CC:0060.03.0B.11.004 The value 0xFF MUST indicate that all existing End Points MUST be returned. If this field is set to 0xFF, the Specific Device Class field MUST also be set to 0xFF.

Specific Device Class (8 bits)

CC:0060.03.0B.11.005 This field MUST indicate the receiving node to return the list of End Point having the specified Specific Device Class.

CC:0060.03.0B.11.006

The value 0xFF MUST indicate that the list of all End Points having the specified Generic Device Class MUST be returned.

4.2.2.8 Multi Channel End Point Find Report Command

This command is used to advertise End Points that implement a given combination of Generic and Specific Device Classes.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_END_POINT_FIND_REPORT (0x0C)							
Reports to Follow							
Generic Device Class							
Specific Device Class							
Res	End Point 1						
...							
Res	End Point N						

Reports to Follow (8 bits)

This field is used if multiple Report Commands are necessary for returning all the requested End Points.

CC:0060.03.0C.11.001 This field MUST advertise the number of Multi Channel End Point Find Report Command following the actual frame.

Generic Device Class (8 bits)

This field is used to advertise the Generic Device Class of all advertised End Points in this command.

CC:0060.03.0C.11.002 The value 0xFF MUST be advertised if this value was specified in the Multi Channel End Point Find Command.

If 0xFF is advertised, the *Specific Device Class* field MUST also advertise the value 0xFF.

CC:0060.03.0C.13.001 If the value 0xFF is advertised, the advertised End Points MAY implement different Generic and Specific Device Classes.

Specific Device Class (8 bits)

This field is used to advertise the Specific Device Class of all advertised End Points in this command.

CC:0060.03.0C.13.002 If the value 0xFF is advertised, the advertised End Points MAY implement different specific device classes.

CC:0060.03.0C.11.004 This field MUST be set to 0xFF if the *Generic Device Class* field is set to 0xFF.

Res

CC:0060.03.0C.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Point (N * 7 bits)

This field is used to advertise the list of End Point identifier(s) that matches the advertised Generic and Specific Device Class values.

CC:0060.03.0C.11.006 If no End Point matches the advertised Generic Device Class and/or Specific Device Class, the sending node MUST set this field to 0x00 and this field's size MUST be 7 bits (only 1 list entry).

4.2.2.9 Multi Channel Command Encapsulation

This command is used to encapsulate commands to or from a Multi Channel End Point.

CC:0060.03.0D.11.001 The Multi Channel Command Encapsulation Command MUST NOT carry Source End Point and Destination End Point fields that are both zero.

CC:0060.03.0D.13.001 A receiving node MAY respond to a Multi Channel encapsulated command if the Destination End
CC:0060.03.0D.11.002 Point field specifies a single End Point. In that case, the response MUST be Multi Channel encapsulated.

CC:0060.03.0D.11.003 A receiving node MUST NOT respond to a Multi Channel encapsulated command if the Destination End Point field specifies multiple End Points via bit mask addressing.

CC:0060.03.0D.11.004 A node MUST NOT return a Multi Channel Encapsulated command in response to a non-encapsulated command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_CMD_ENCAP (0x0D)							
Res	Source End Point						
Bit address	Destination End Point						
Command Class (1 or 2 bytes)							
Command							
Parameter 1							
...							
Parameter N							

Res

CC:0060.03.0D.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Source End Point (7 bits)

CC:0060.03.0D.11.006 This field is used to advertise the originating End Point. The Source End Point MUST be in the range 0..127.

CC:0060.03.0D.11.007 The value 0 MUST indicate that the encapsulated command is issued by the Root Device.

Values in the range 1…127 MUST indicate the actual End Point identifier which issues the encapsulated command.

CC:0060.03.0D.11.008 This field MUST be set to a different value than 0 if the *Destination End Point* field is set to 0.

CC:0060.03.0D.11.009 A node returning a response to a Multi Channel encapsulated command MUST swap the Source and Destination End Point identifiers in this command.

CC:0060.03.0D.11.00A This field MUST be set to 0 if a sending node does not implement Multi Channel End Points or if the Root Device of the Multi Channel device is issuing a command.

Bit address (1 bit)

This bit is used to advertise if the destination End Point is specified as a bit mask.

CC:0060.03.0D.11.00B The value 1 MUST indicate that the Destination End Point field is specified as a bit mask.

The value 0 MUST indicate that the Destination End Point field is specified as an End Point identifier.

Destination End Point (7 bits)

This field is used to advertise the destination End Point.

CC:0060.03.0D.11.00C This field MUST be interpreted based on the “Bit address” value.

CC:0060.03.0D.11.00D If the Bit Address field is set to 0, the Destination End Point field MUST carry a single End Point identifier value in the range 0..127.

CC:0060.03.0D.11.00E If the Bit Address field is set to 1, the Destination End Point MUST use the following encoding:

- Bit 0 in the Destination End Point indicates if End Point 1 is a destination
- Bit 1 in the Destination End Point indicates if End Point 2 is a destination
- …

CC:0060.03.0D.11.00F The bit value 0 MUST be used to advertise that the corresponding End Point is not a destination.

The bit value 1 MUST be used to advertise that the corresponding End Point is a destination.

Command Class (8 bits or 16 bits)

CC:0060.03.0D.11.011 This field MUST specify the Command Class identifier of the encapsulated Command. This field MUST carry a normal Command Class (8 bits) or an Extended Command Class (16 bits)

Command (8 bits)

CC:0060.03.0D.11.012 This field MUST specify the Command identifier of the encapsulated command.

Parameter (N bytes)

CC:0060.03.0D.11.010 This field MUST carry the payload of the encapsulated command. The length of this field MUST be determined from the Z-Wave frame length.

4.2.3 Multi Channel Command Class, version 4

The Multi Channel Command Class is used to address one or more End Points in a Multi Channel device.

Refer to Section 4.1.3 for an introduction to the Multi Channel concept.

4.2.3.1 Compatibility Considerations

CC:0060.04.00.21.001 Compatibility considerations requirements from version 3 MUST also be observed by a version 4 supporting node. Refer to Section 4.2.2.1 Compatibility considerations.

CC:0060.04.00.21.002 Multi Channel Command Class, version 4 is backwards compatible with Multi Channel Command Class, version 3. Fields and commands not described in this version MUST remain unchanged from version 3.

> **AGGREGATED END POINTS HAVE BEEN DEPRECATED**
>
> Aggregated End Points have been deprecated. It is RECOMMENDED to issue multiple report back-to-back if data from several End Points at a precise time needs to be reported.
>
> Additionally, it is RECOMMENDED to aggregate all sensors functionalities at the Root device. i.e. accumulated readings for all End Points SHOULD be advertised via the Root Device.

The Multi Channel Command Class, version 4 introduces Aggregated End Points. Aggregated End Points are assigned End Point identifiers following immediately after the identifiers allocated to individual End Points. Thus:

- Aggregated End Points are invisible to devices supporting Multi Channel Command Class, version 3 or older.
- Individual End Points are identical in version 3 and version 4.
- A version 3 controlling node can discover and control individual End Points.
- A version 4 controlling node can discover and control individual End Points and Aggregated End Points.

4.2.3.2 Interoperability Considerations

CC:0060.04.00.31.001 Interoperability considerations from version 3 MUST also apply in this version.

Refer to Section 4.2.2.2 Interoperability considerations

4.2.3.2.1 Aggregated End Point design principles

CC:0060.04.00.31.002 An **Aggregated End Point** MUST implement a function which relates to multiple individual End Points.

CC:0060.04.00.31.003 An Aggregated End Point MUST NOT forward commands to individual End Points. In other words, communication to a number of individual End Points MUST be done via multiple singlecast commands or via bit mask addressing.

CC:0060.04.00.31.004 A command issued to an Aggregated End Point MUST NOT cause any individual End Point to return a command in response.

CC:0060.04.00.31.005 Aggregated End Point MUST be assigned End Point identifiers from a continuous range starting immediately after the last individual End Point.

CC:0060.04.00.31.006 An Aggregated End Point MUST NOT support other Command Classes and types than the ones explicitly listed in Table 4.2.

Table 4.2: Aggregated End Point Command Class support

Command Class	Type	Measurement Mode
Meter	Electricity	Instant, Accumulated
Meter	Gas	Instant, Accumulated
Meter	Water	Instant, Accumulated
Multilevel Sensor	Power	Instant
Multilevel Sensor	Current	Instant
Multilevel Sensor	Air flow	Instant
Multilevel Sensor	Tank Capacity	Instant

4.2.3.2.2 Dynamic End Point considerations

In the case a node implements both Dynamic and Aggregated End Points, the Aggregated End Points identifiers will vary accordingly to the last active dynamic End Point. An illustration is given in Figure 4.2.

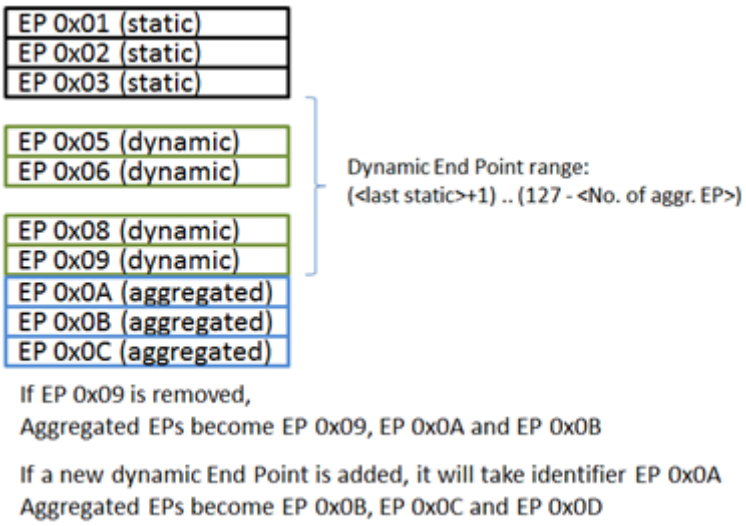


Figure 4.2: Static, dynamic and aggregated End Point layout example

4.2.3.3 Multi Channel End Point Report Command

This command is used to advertise the number of Multi Channel End Points and other relevant Multi Channel attributes.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_END_POINT_REPORT (0x08)							
Dynamic	Identical	Res					
Res	Individual End Points						
Res	Aggregated End Points						

CC:0060.04.08.11.001 Fields not described below MUST remain unchanged from version 3. Refer to Section 4.2.2.4 *Multi Channel End Point Report Command*.

Res

CC:0060.04.08.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Individual End Points (7 bits)

This field is used to advertise the number of individual End Points implemented by the sending node.

CC:0060.04.08.11.003 This field MUST be in the range 1..127.

Aggregated End Points (7 bits)

CC:0060.04.08.11.004 CC:0060.04.08.13.001 If the sending node implements dynamic End Points, this field MUST advertise the number of End Points currently instantiated by the node. A dynamic End Point MAY be assigned any End Point identifier in the range 2..127.

CC:0060.04.08.11.005 The sum of the values advertised by this field and the *Aggregated End Points* field MUST be in the range 1..127.

4.2.3.4 Multi Channel Capability Get/Report Commands

These commands are unchanged in version 4.

Aggregated End Points SHOULD reply to this command and to the S2/S0 Supported Get Commands.

Aggregated End Points SHOULD set the Generic and Specific Device Class field to an identical value to one of the Individual End Point it aggregates.

The list of supported Application Command Classes at the highest security level MUST only comprise these Command Classes:

- Meter Command Class
- Multilevel Sensor Command Class.

4.2.3.5 Multi Channel Capability Find Report Commands

This command is unchanged in version 4.

Aggregated End Points MUST NOT be advertised in the list of End Points returned for any Generic/Specific Device Type.

4.2.3.6 Multi Channel Aggregated Members Get Command

This command is used to query the members of an Aggregated End Point.

CC:0060.04.0E.11.001 The Multi Channel Aggregated Members Report MUST be returned in response to this command.

CC:0060.04.0E.51.001 This command MUST NOT be issued via multicast addressing.

CC:0060.04.0E.11.002 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_AGGREGATED_MEMBERS_GET (0x0E)							
Res	Aggregated End Point						

Res

CC:0060.04.0E.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Aggregated End Point (7 bits)

CC:0060.04.0E.11.004 This field MUST specify the Aggregated End Point identifier for which the aggregated members MUST be returned.

CC:0060.04.0E.11.005 The value MUST be in the range of advertised Aggregated End Points. If the value does not indicate valid aggregated End Point identifier, a receiving node MUST return a response with the *Number of Bit Masks* field set to zero.

4.2.3.7 Multi Channel Aggregated Members Report Command

This command is used to advertise the members of an Aggregated End Point.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL (0x60)							
Command = MULTI_CHANNEL_AGGREGATED_MEMBERS_REPORT (0x0F)							
Res	Aggregated End Point						
Number of Bit Masks							
Aggregated Members Bit Mask 1							
...							
Aggregated Members Bit Mask N							

Res

CC:0060.04.0F.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Aggregated End Point (7 bits)

This field is used to advertise the Aggregated End Point identifier for which the members are advertised in this command.

Number of Bit Masks (8 bits)

This field is used to advertise the length in bytes of the *Aggregated Member Bit Mask* field.

CC:0060.04.0F.11.002 The value 0 MUST indicate that the Aggregated Members Bit Mask field is MUST be omitted.

Values in the range 1..255 MUST indicate the length of the *Aggregated Members Bit Mask* field in bytes.

Aggregated Members Bit Mask (N bytes)

This field is used to advertise the End Point members of the actual Aggregated End Point.

CC:0060.04.0F.11.003 The length of this field in bytes MUST be according to the *Number of Bit Masks* field value.

This field MUST be treated as a bit mask and MUST use the following encoding for advertising members:

- Bit 0 in Bit Mask 1 indicates if End Point 1 is a member
- Bit 1 in Bit Mask 1 indicates if End Point 2 is a member
- ...

CC:0060.04.0F.11.004 The bit value 0 MUST be used to advertise that the corresponding End Point is not a member.

The bit value 1 MUST be used to advertise that the corresponding End Point is a member.

CC:0060.04.0F.11.005 The first byte of this field MUST represent End Points 1..8.

4.2.4 Multi Command Command Class, version 1

The Multi Command Command Class is used to bundle multiple commands in one encapsulation Command. This command class may be used to limit the number of transmissions and to extend battery lifetime.

4.2.4.1 Interoperability considerations

> **The "Answer-as-asked" requirement for Multi Command Encapsulation commands carrying Get type commands has been OBSOLETED.**
>
> This allows for a simple parser design in end devices, enables battery power savings and supports the deployment of gateways with a part of the application logic placed in the cloud.
>
> Refer to section Section 4.2.4.2 for updated requirements text.

Older implementations may expect Get type commands to be answered with the same encapsulation.

CC:008F.01.00.31.001 However, responding nodes MUST NOT return answers with the same encapsulation if the destination does not advertise the Multi Command Command Class as supported.

CC:008F.01.00.32.001 Supporting nodes SHOULD NOT return an answer Multi Command encapsulated if returning a single command and SHOULD NOT return individually Multi Command encapsulated response commands.

4.2.4.2 Compatibility considerations

4.2.4.2.1 Multi Command Support

CC:008F.01.00.21.001 A node supporting this Command Class MUST be able to receive Multi Command Encapsulated commands.

CC:008F.01.00.21.002 A supporting node MUST support the Multi Command Encapsulation of all command classes advertised as supported (for the received Security Class) except for command classes that are encapsulated outside Multi Command. Refer to the encapsulation order defined in section Section 4.1.3.5.

CC:008F.01.00.21.003 A supporting node MUST respond to an encapsulated command requiring an answer to be returned, e.g. a Get type Command.

CC:008F.01.00.21.004 A responding node MUST NOT return Multi Command encapsulated commands in response to encapsulated requests if the sender does not support the Multi Command Command Class.

4.2.4.2.2 Multi Command Control

A node controls the Multi Command Command Class if it sends Multi Command Encapsulated commands to supporting nodes.

CC:008F.01.00.23.001 It means that a node MAY issue unsolicited Multi Command Encapsulated commands without advertising support for the Multi Command Command Class.

CC:008F.01.00.51.001 A controlling node MUST verify that the destination supports Multi Command in its NIF before using Multi Command Encapsulation or be explicitly enabled by another controller node to use Multi Command encapsulation.

4.2.4.2.3 Node Information Frame (NIF)

CC:008F.01.00.21.005 A supporting node MUST always advertise the Multi Command Command Class in its NIF, regardless of the security bootstrapping outcome.

This allows other nodes bootstrapped on any security level to know that they can use the Multi Command encapsulation with the supporting node.

4.2.4.3 Multi Command Encapsulated Command

The Multi Command Encapsulated Command used to contain multiple Commands.

CC:008F.01.01.11.001 The encapsulated Commands MUST be executed in the order they are received. In case Get type Commands in a Multi Command Encapsulated Command are received by a device, the Report type Commands MUST be returned in the same order as the Get type Commands were received.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CMD (0x8F)							
Command = MULTI_CMD_ENCAP (0x01)							
Number of Commands							
Command Length 1							
Command Class 1 (1 or 2 bytes)							
Command 1							
Data 1,1							
...							
Data 1,N							
...							
Command Length X							
Command Class X (1 or 2 bytes)							
Command X							
Data X,1							
...							
Data X,N							

Number of Commands (8 bits)

CC:008F.01.01.11.002 This field MUST specify the number of encapsulated commands.

CC:008F.01.01.12.001 This field SHOULD be set to a value greater than 1.

CC:008F.01.01.11.003 Each block carrying an encapsulated command MUST comprise the following fields:

- Command length
- Command Class
- Command
- Data

CC:008F.01.01.11.004 A supporting node MUST accept and execute all encapsulated commands contained in this command.

A supporting node MUST NOT discard any encapsulated command based on the number of commands encapsulated in the command.

Command Length (8 bits)

CC:008F.01.01.11.005 This field MUST specify the number of bytes occupied by the Command Class, Command and the Data fields in the actual command block.

Command Class (8 bits or 16 bits)

CC:008F.01.01.11.006 This field MUST specify the Command Class identifier of the encapsulated command. This field MUST carry a normal Command Class (8 bits) or an Extended Command Class (16 bits).

Command (8 bits)

CC:008F.01.01.11.007 This field MUST specify the Command identifier of the encapsulated Command.

Data (N bytes)

CC:008F.01.01.11.008 This field MUST carry the payload of the encapsulated command.

4.2.5 Security 0 (S0) Command Class, version 1

The Security Command Class create the foundation for secure application communication between nodes in a Z-Wave network. The security layer provides confidentiality, authentication and replay attack robustness through AES-128.

The Security Command Class defines a number of commands used to facilitate handling of encrypted frames in a Z-Wave Network. The commands deal with three main areas:

- Message Encapsulation. The task of taking a plain text frame and encapsulating the frame into an encrypted Security Message.
- Command Class Handling. The task of handling what command classes are supported when communicating with a Security enabled device
- Network Key Management. The task of initial key distribution.

4.2.5.1 Compatibility considerations

A node supporting the S0 Command Class MAY use the S2 CTR_DRBG as a PNRG.

4.2.5.1.1 Node Information Frame (NIF)

A supporting node MUST advertise the Security 0 Command Class in its NIF before inclusion.

A supporting node MUST advertise the Security 0 Command Class in its NIF after successful S0 security bootstrapping.

A supporting node MAY advertise the Security 0 Command Class in its NIF after inclusion without Security bootstrapping.

A supporting node MUST NOT advertise the Security 0 Command Class in its S0/S2 Commands Supported Report list.

4.2.5.2 Message Encapsulation and Command Class Handling

For encapsulating messages, Z-Wave requires four commands. Before sending an encrypted frame, the sender MUST request a nonce (number used once) from the recipient. The sender subsequently uses this number along with the locally generated nonce and the network key to generate the Security Message Encapsulation Command as illustrated in Figure 4.3.

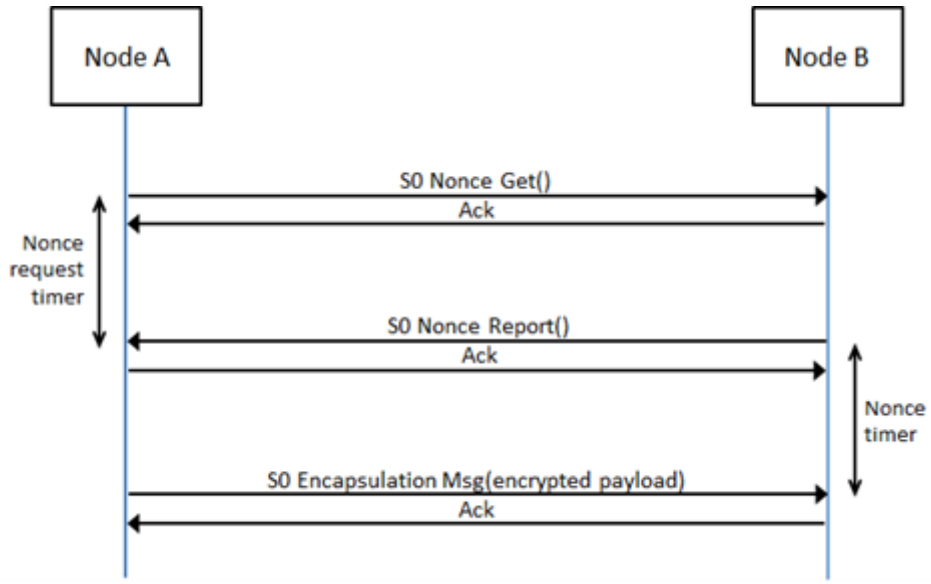


Figure 4.3: Sending secure messages

This mechanism generates an overhead of three commands for each single frame that is sent encrypted (plus acknowledge frames).

A number of timers have to be implemented in order to mitigate attacks.

A timer denoted *Nonce request timer* in Figure 4.3 and Figure 4.4 SHOULD be started by a node sending a Nonce Get Command. If the *Nonce request timer* is started, the Nonce Report MUST be received before the timer runs out. The duration of this timer will depend on the application it is trying to protect.

A timer denoted *Nonce timer* in Figure 4.3 and Figure 4.4 MUST be started by a node after sending a Nonce Report Command. The S0 Encapsulated Message MUST be received within the specified timeout in order to be accepted.

The *Nonce timer* MUST implement a timeout in the range 3..20 seconds.

Note that the *Nonce timer* and the *Nonce request timer* MUST be started when the command has been sent and not when the transmission has been acknowledged, since an attacker could delay the acknowledgement frame.

Both timers MUST be used in all communication that uses the mentioned commands.

In order to optimize the performance the device MUST use streaming when transmitting multiple frames. The overhead using this option will converge towards two (instead of three) transmissions as the number of frames increases.

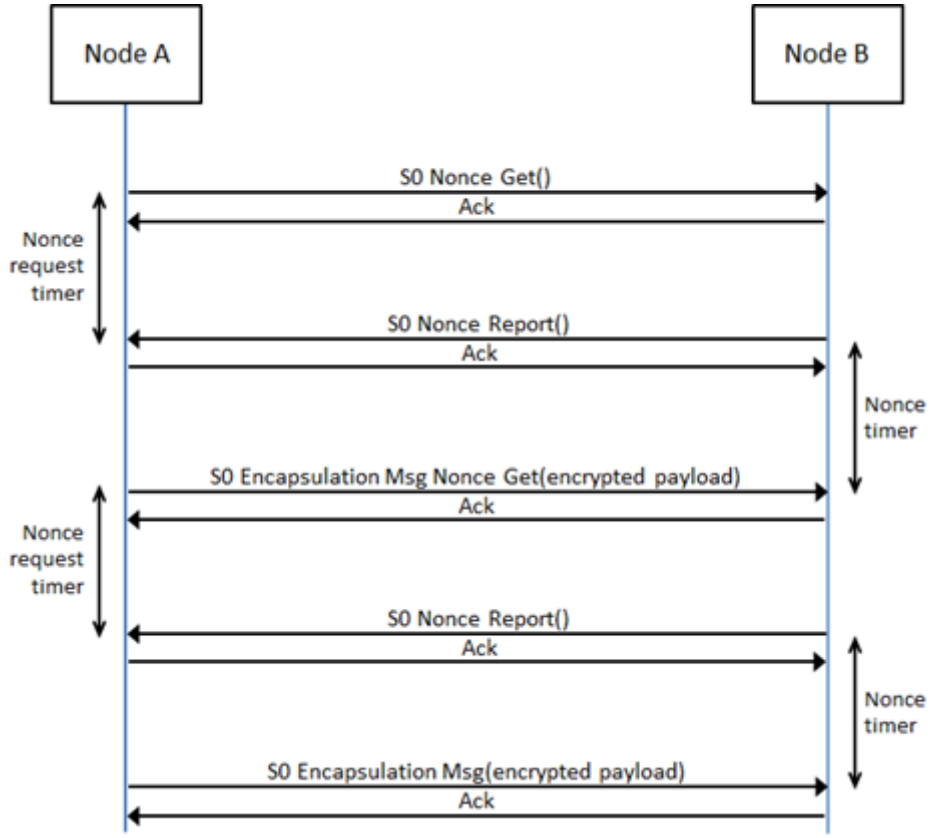


Figure 4.4: Streaming secure messages

Notice: The maximum command size is reduced by 20 bytes due to the security encapsulation command overhead. Larger commands can use sequencing as described in Section 4.2.5.2.2.

4.2.5.2.1 Nonce Get Command

This command is used to request an external nonce from the receiving node.

Note that a nonce will only be valid for one encrypted command attempt. The nonce is discarded when the receiver has used it for decrypting the next received command. A new nonce MUST be exchanged for each new command.

The Nonce Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Security Header = SECURITY_NONCE_GET (0x40)							

4.2.5.2.2 Nonce Report Command

This command is used to return the next nonce to the requesting node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Security Header = SECURITY_NONCE_REPORT (0x80)							
Nonce Byte 1							
Nonce Byte 2							
Nonce Byte 3							
Nonce Byte 4							
Nonce Byte 5							
Nonce Byte 6							
Nonce Byte 7							
Nonce Byte 8							

Nonce byte (8 bytes)

This field contains the 8 bytes external nonce used for encryption, generated with the PNRG by the sending node.

4.2.5.2.3 Security Message Encapsulation Command

The device uses the Security Message Encapsulation command to encapsulate Z-Wave commands using AES-128.

A sending node is also requesting a new nonce from the receiving node when transmitting the Security Message Encapsulation Nonce Get Command. The sending node uses the new nonce when streaming multiple secure messages without having to send a separate Nonce Get Command after sending each command as shown in Figure 4.4, Streaming secure messages.

A device MUST ignore the received Security Message Encapsulation Command if the generated Nonce has timed out.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Security Header = SECURITY_MESSAGE_ENCAPSULATION (_NONCE_GET) (0x81/0xC1)							
Initialization Vector Byte 1							
Initialization Vector Byte 2							
Initialization Vector Byte 3							
Initialization Vector Byte 4							
Initialization Vector Byte 5							
Initialization Vector Byte 6							
Initialization Vector Byte 7							
Initialization Vector Byte 8							
Reserved		Second Frame	Sequenced	Sequence Counter			
(Command Class identifier)							
(Command identifier)							
Command byte 1							
...							
Command byte N							
Receiver's nonce Identifier							
Message Authentication Code byte 1							
Message Authentication Code byte 2							
Message Authentication Code byte 3							
Message Authentication Code byte 4							
Message Authentication Code byte 5							
Message Authentication Code byte 6							
Message Authentication Code byte 7							
Message Authentication Code byte 8							

Initialization Vector byte (8 byte)

The initialization vector is the internal nonce generated by the sender. The payload is encrypted with the external and internal nonce concatenated together.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

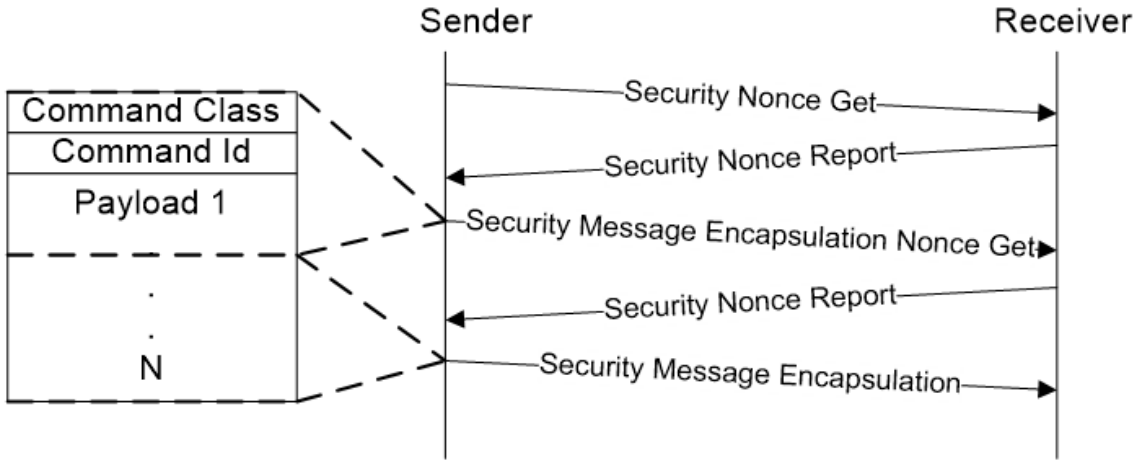


Figure 4.5: Frame flow for sequenced frames

Sequenced (1 bit)

This flag MUST be set if the command is transmitted using multiple frames. This flag MUST not set if the command is contained entirely in a single (this) frame. As shown in figure, the first frame in

a sequence MUST be sent using Security Message Encapsulation Nonce Get. To minimize overhead, following frames SHOULD be sent using the Security Message Encapsulation Nonce Get command. The last frame MAY be sent using Security Message Encapsulation.

Second Frame (1 bit)

If this flag and the Sequenced flag are set, the frame is the second out of two. If the flag is not set, and Sequenced flag is set, it is the first frame out of two. Valid combinations are:

Table 4.3: Security message encapsulation::Second Frame combinations

	Sequenced 1	Sequenced 0
Second Frame 1	Second frame of two	-
Second Frame 0	First frame of two	Single Frame

Sequence Counter (4 bits)

If Sequenced flag is set, the frame is one out of two. In order to tell multiple sequences apart, they MUST be uniquely identified based on the sender NodeID and the Sequence Counter. For each sequenced set of frames a node sends it MUST increment the Sequence Counter by one.

Command Class Identifier (8 bits) (Part of Encrypted Payload)

This field contains the identifier of the Command class, which the device sends to the NodeID.

Command identifier (8 bits) (Part of Encrypted Payload)

This field contains the identifier of the Command, which the device sends to the NodeID.

Command byte (N bytes) (Part of Encrypted Payload)

These fields contain the parameters, which the device sends to the NodeID.

Receiver's nonce Identifier (8 bits)

Identifies nonce being used.

Message Authentication Code byte (8 bytes)

Data used for authenticating the received message to prevent tampering.

4.2.5.3 Network Key Management

The same network key is used by all secure nodes in the network. Distribution of network keys uses a temporary key to protect the key exchange. Exchange of network key happens immediately after successful inclusion of the node. It requires a secure primary/inclusion controller to include a secure node into the secure network as secure.

4.2.5.3.1 Network inclusion

The first step of including a node to a secure network is using the standard Z-Wave inclusion process. If both the new node and the inclusion controller support Security command class, the controller will subsequently send the network key to the newly included node.

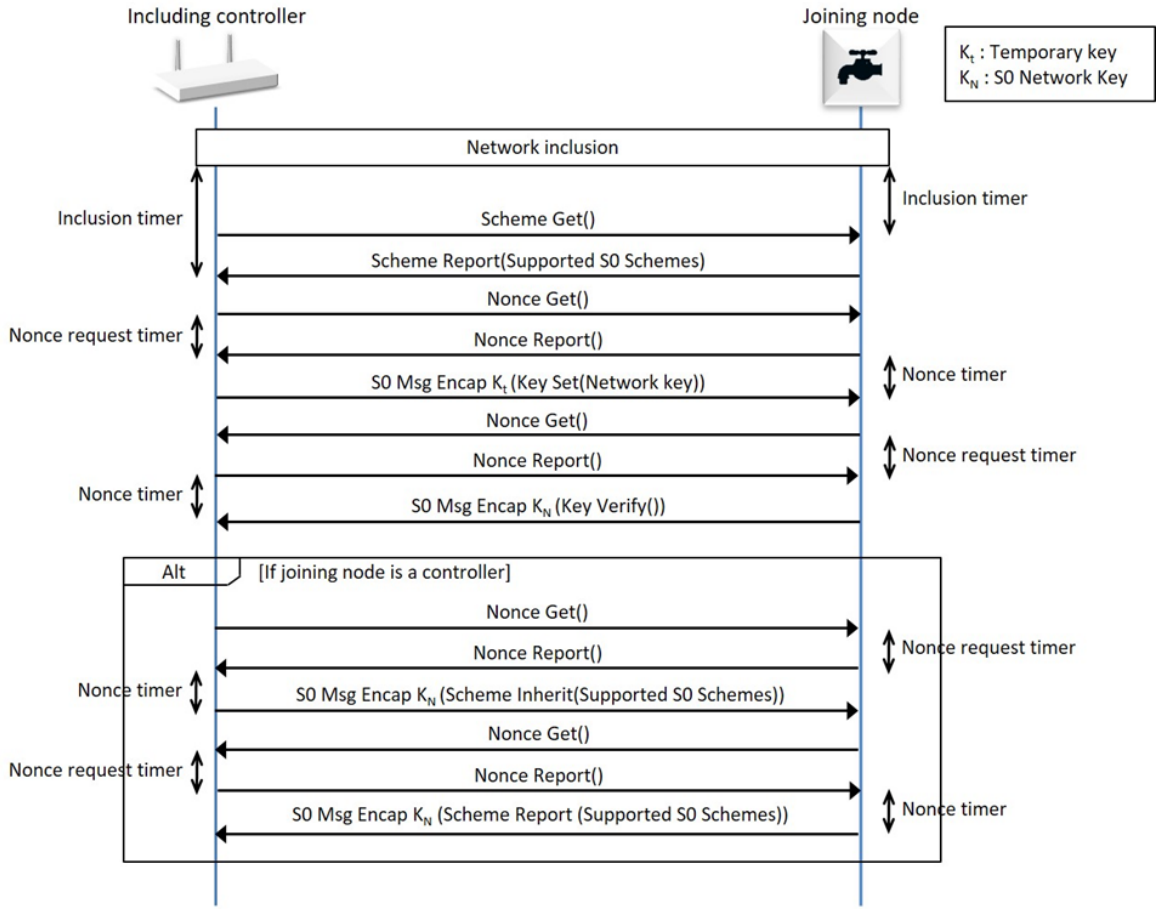


Figure 4.6: Inclusion into a secure network

To protect the security of a secure network, all controllers SHOULD require a PIN to unlock the security inclusion process and end nodes SHOULD require a PIN to accept being included and excluded.

Following the inclusion of the node into the network, the controller will request the security scheme supported by the included node. Battery operated devices SHOULD stay awake for the duration of the setup of the Security Command class.

Currently one security scheme exist which is extendable at a later stage:

1. **Security 0/N:** 0x00 repeated 16 times as temporary key for encrypting the network key when it is transferred using normal power.

The validity of the key is verified in both the added node and the including controller. The node verifies the key based on the Message Authentication Code and then transmits an encrypted Network Key Verify command as response to the controller. When a device supporting the Security Command class does not manage to enter the secure network, it will function as a non-secure device. The node requires exclusion from the network before another attempt comprising of inclusion and network key exchange is possible.

For the currently available Security 0/N scheme, the same network key is used by all nodes in the network.

For the including controller to allow S0 bootstrapping into the secure network, a common security scheme needs to be supported by both nodes. When supporting multiple common schemes, the highest possible scheme MUST be used. If no common schemes are supported the node MUST NOT be S0 bootstrapped.

When controller nodes in the secure network wish to establish a connection to a node that supports the Security 0 Command class, they MUST send the Security 0 Command Supported Get Command to the node. Receiving no Security Command Supported Report (since the receiving node does not have the key to decrypt the request), means that it will not be able to talk to this node securely.

The same applies for the situation where a secure node does not become part of the secure network because it was included by a non-secure controller.

A node based on a end node Role Type MUST NOT consider a secure inclusion successful until the Network Key Set has been received.

A node based on a controller Role Type MUST NOT consider the secure inclusion successful until the Security Scheme Inherit Command has been received.

4.2.5.3.2 Inclusion through Non-Secure Inclusion controller

A Security-enabled SIS MAY perform secure setup after inclusion from a non-secure inclusion controller. As soon as the Security enabled SIS (hereafter SIS), receives information from the non-secure inclusion controller that a node with support for the Security command class has been included, the SIS MAY start the secure setup process of sending the network key to the newly included node as illustrated in Figure 4.7. At this stage the SIS acts as if it, itself had performed the inclusion and MAY carry out all the steps REQUIRED for secure setup, included making sure the timeouts are not exceeded.

Before starting the Secure inclusion process, the SIS MUST be put into a state that allows it to carry out the secure setup for 1 node for the next 3 minutes and no longer. The SIS MUST be put in this state through a password-protected menu to avoid unintentional reveal of the network key by a fake controller.

It should be noted that performing the secure setup on behalf of a non-secure inclusion controller might add to the complexity of the actions required by the user, and thus make it easier for a hacker to perform social engineering to circumvent the security so care must be taken to inform the user accordingly.

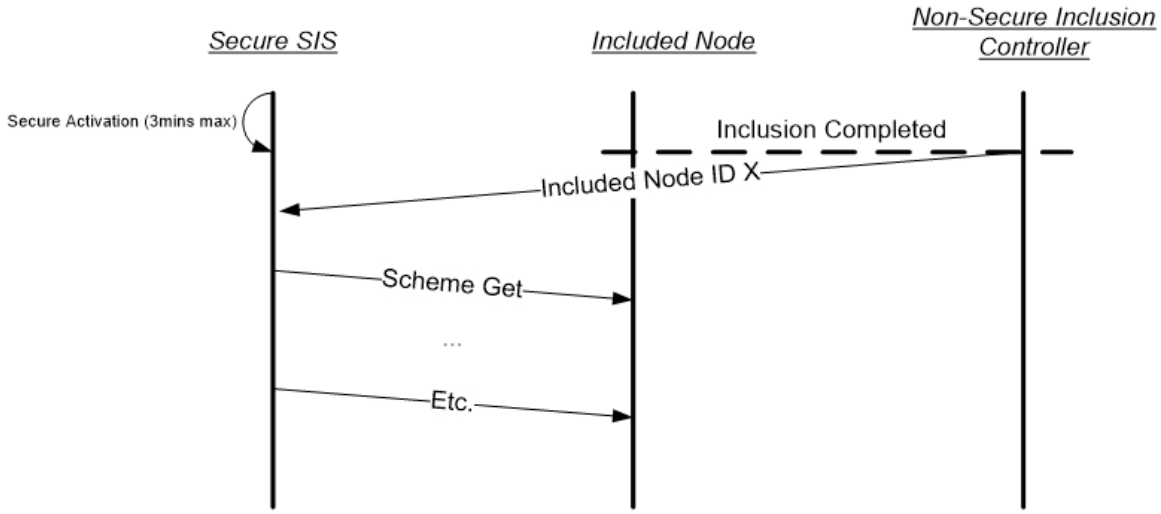


Figure 4.7: Secure Inclusion through Non-Secure Inclusion Controller

4.2.5.3.3 Inclusion Timers

As shown in Figure 4.6, a number of timeout MUST be complied with. For the including controller see Figure 4.8.

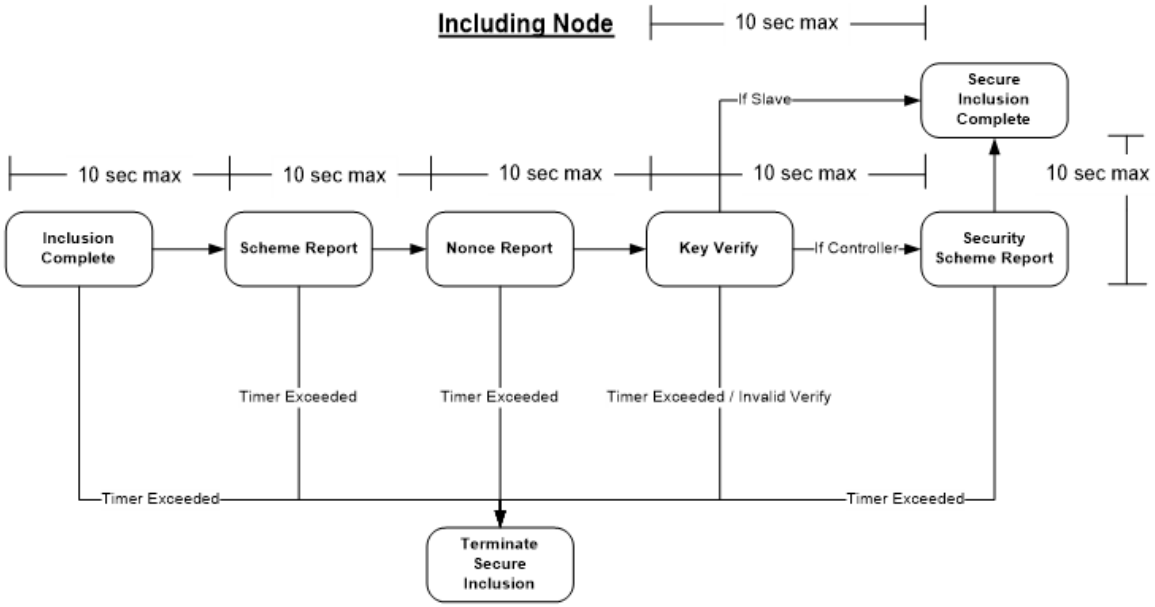


Figure 4.8: Timers on Including Controller

For the new included node, the timers in Figure 4.9 MUST be complied with.

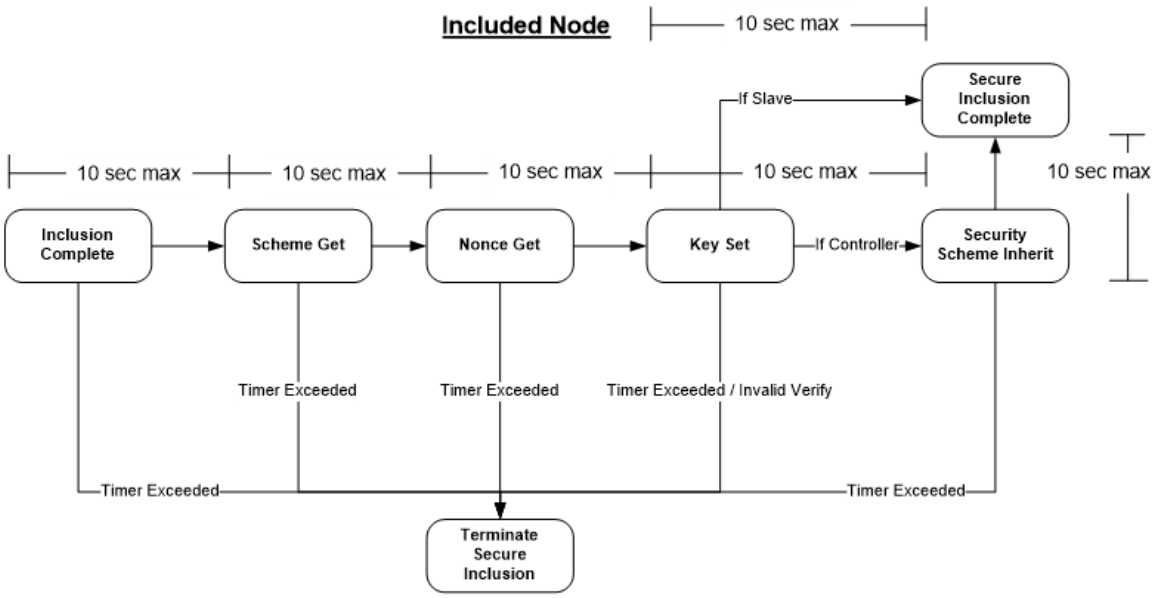


Figure 4.9: Timers on newly Included Node

The Network Key MUST NOT be sent to the new node if a Security Scheme Report Command is received by the including controller later than 10 seconds after successful inclusion of the node. The controller SHOULD notify the user of an error condition in case of timeout because the device functions only as non-secure. In addition, the included node MUST NOT accept and respond to a Scheme Get it is received later than 10 seconds after network inclusion. When a valid frame is received before the timeout, the timeout is extended to allow the next part of the inclusion process. The S0 bootstrapping process MUST be terminated if any message times out.

4.2.5.3.4 Security Scheme Get Command

A controlling device MUST send Security Scheme Get Command immediately after the successful inclusion of a node that supports the Security Command class.

A node is considered newly included if it has been included for less than 10 seconds.

A newly included node MUST return the Security Scheme Report Command in response to this command.

Whether a node has been included securely or non-securely, the node MUST NOT respond to the Security Scheme Get command if it is not newly included.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = SECURITY_SCHEME_GET (0x04)							
Supported Security Schemes							

Supported Security Schemes (8 bits)

The Security Schemes which are supported by the primary/inclusion controller. At least one security scheme MUST be supported. Values MUST comply with Table 4.4.

Table 4.4: Security Scheme Get::Supported Security Schemes encoding

Bit	Supports
0	Security 0 using normal power = 0

Bit 0 MUST always be set to 0, indicating support for Security 0. All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

4.2.5.3.5 Security Scheme Report Command

This command is used to advertise security scheme 0 support by the node being included. Upon reception, the including controller MUST send the network key immediately without waiting for input, by using 16 times 0x00 as the temporary key. The including controller MUST NOT perform any validation of the Supported Security Schemes byte.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = SECURITY_SCHEME_REPORT (0x05)							
Supported Security Schemes							

Supported Security Schemes (8 bits)

Refer to Security Scheme Get Command.

4.2.5.3.6 Network Key Set Command

The Device can use the Network Key Set Command to set the network key in a Z-Wave node. Transmission of the Network Key Set command requires existence of a common agreed security scheme. The device uses the agreed temporary key to encapsulate the Network Key Set command. The included node MUST handle the Network Key Set command according to the guidelines in section Section 4.2.5.3.

This command MUST be sent encapsulated by the Security Message Encapsulation command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = NETWORK_KEY_SET (0x06)							
Network Key byte 1							
...							
Network Key byte N							

Network Key byte (N bytes)

The Network key to exchange application data secure in the network.

4.2.5.3.7 Network Key Verify Command

When the included node has received a Network Key Set that is has successfully decrypted, verified by the MAC, it MUST send a Network Key Verify Command to the including controller. If the controller is capable of decrypting the Network Key Verify command it would indicate that the included node has successfully entered the secure network. Since there is no timeout for the Network Key Verify, the controller can send a Security Commands Supported Get command, and if no response is received, it SHOULD be concluded that the node has not been included properly.

This command MUST be sent encapsulated by the Security Message Encapsulation command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = NETWORK_KEY_VERIFY (0x07)							

4.2.5.3.8 Security Scheme Inherit Command

When a controller is included to the network, it MUST inherit the same security scheme as the including controller allowing it to become an inclusion controller. This is achieved through the Security Scheme Inherit Command, which is sent when the network key has successfully been setup, as shown in Figure 4.6.

When including a controller into the secure network, the new controller MUST inherit any common supported security schemes. For example, if the new controller supports security scheme bit 1 and bit 4 but the including controller only supports security scheme bit 1, the new controller MUST after inclusion also only support security scheme bit 1.

This command MUST be sent encapsulated by the Security Message Encapsulation command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = SECURITY_SCHEME_INHERIT (0x08)							
Supported Security Scheme							

Supported Security Schemes (8 bits)

See Security Scheme Get command, for a definition.

To ensure that the included controller has inherited the correct security scheme, it MUST respond with a Security Scheme Report command as illustrated in Figure 4.6. If the reported security scheme does not match, the installer MUST be notified that the included controller is violating the security scheme, and the node SHOULD be excluded again as an error situation has occurred.

4.2.5.4 Encapsulated Command Class Handling

The Node Info Frame is only used to advertise all the command classes that are supported non-securely. Command classes supported securely MUST be advertised by using the Security Commands Supported Get/Report.

- All non-securely supported command classes MUST also be supported securely.
- All non-securely controlled command classes MUST also be controlled securely.
- All non-securely supported command classes MUST NOT be explicitly advertised in the Security Commands Supported Report.

To make a security enabled device compatible with non-secure applications a secure node MAY choose to report support for some command classes non-secure in the Node Info Frame, as well as in the Security Command Supported Report.

Initially, the Node Info Frame MUST advertise all non-securely supported command classes, while the Node Info Frame MAY advertise non-securely controlled command classes.

If the node is included into a secure network, it MAY choose to remove all or some command classes from the Node Info Frame, and thus only support them securely – removing support for the command classes for all non-secure nodes.

If an S0 node is included into a non-secure network, it MAY choose to support command classes it would not support non-securely if it had been included into a secure network.

An example of this could be a relay as shown in Table 4.5.

Table 4.5: S0 Node Command Class support depending on inclusion (example)

	Before Inclusion	Included Non-Secure	Included Secure
Security Command Supported Report Frame	-N/A	-N/A	Binary Switch Version
Node Info Frame	Security Binary Switch Version	Binary Switch Version	Security

It is up to the implementation of each application to decide which commands should be supported using security encapsulation and non-secure.

If a command class is only supported securely it MUST NOT be listed in the node info frame, while it MUST be advertised in the security commands supported report frame.

In a secure network, initially only the including controller will have any knowledge about what nodes in the network have been setup securely. If a node wishes to talk to another node it MAY send a Security Command Supported Get command encapsulated to the other node. If a Security Commands Supported Report is returned the node is in possession of a valid network key, and is part of the secure network. This mechanism may also be used by the including controller to ensure that the node has been included properly.

4.2.5.4.1 Multi Channel Handling

Any device that supports the Security and Multi Channel Command Classes MAY choose to support a different set of Command Classes securely for each Multi Channel End Point. An End Point with support for Security MUST report the Security Command Class as supported for that End Point. The command classes supported for each endpoint securely is determined by using the Security Commands Supported Get command sent to each individual endpoint Security Encapsulated. Hence, the encapsulation order is: Security Encapsulation - Multi Channel Encapsulation - Security Commands Supported Get Command

When communicating with a device that supports multiple Multi Channel End Points, the Security Encapsulation MUST be added outside of the Multi Channel Command Class. Thus, a receiving node MUST first remove the Security Encapsulation and then forward it to the actual destination Multi Channel End Point.

- A Multi Channel End Point MUST be considered as a separate device, with separate NIF - given by Multi Channel Capability Report and Security Commands Supported Report.
- Multi Channel End Points are logical abstractions. Only the Root Device is included in the network.

This means:

- Inclusion always deals with the Root Device.
- A Security Command Support Get must reply as a Root Device. If the Multi Channel Command Class is not supported non-securely, it will only be listed in the Security Command Supported Report.
- The Multi Channel Capability Report MUST advertise the Security 0 Command Class as supported for all End Points that implement command classes that are supported securely.

  It has been found that legacy nodes do not always advertise the S0 Command Class in their Multi Channel Capability Report and still accept all their Command Class using S0 encapsulation. A controlling node SHOULD try to control End Points with S0 encapsulation even if S0 is not listed in the Multi Channel Capability Report.
- The implicit rule that all non-secure command classes for an End Point must be controllable securely is still in effect, if the endpoint is reported secure.
- An End Point only inherits the security capabilities of the End Point itself. I.e. each End Point is considered a device itself.

4.2.5.4.2 Security Commands Supported Get Command

This command is used to query the commands supported by the device when using secure communication.

The Security Commands Supported Report Command MUST be returned in response to this command.

A node MAY choose only to advertise a Command Class as 'supported' and/or 'controlled', when secure communication is used. In that case the Command Class MUST NOT be advertised in the NIF, while it MUST be advertised in the Security Commands Supported Report Command.

Secure communication MUST be used when transmitting this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = SECURITY_COMMANDS_SUPPORTED_GET (0x02)							

4.2.5.4.3 Security Commands Supported Report Command

This command advertises which command classes are supported using security encapsulation..

- All non-securely supported command classes MUST NOT be advertised in the Security Commands Supported Report.
- All securely supported command classes MUST be advertised in the Security Commands Supported Report if they are only supported securely.

Secure communication MUST be used when transmitting this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = SECURITY_COMMANDS_SUPPORTED_GET (0x02)							

To support extended command classes use the following format. Note that these MAY be mixed.

This command MUST only be send encapsulated by the Security Message Encapsulation command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY (0x98)							
Command = SECURITY_COMMANDS_SUPPORTED_REPORT (0x03)							
Reports to follow							
Commands Class MSB (0xF1-0xFF) 1							
Commands Class LSB (0x00-0xFF) 1							
...							
Commands Class MSB (0xF1-0xFF) N							
Commands Class LSB (0x00-0xFF) N							
COMMAND_CLASS_MARK							
Commands Class MSB (0xF1-0xFF) 1							
Commands Class LSB (0x00-0xFF) 1							
...							
Commands Class MSB (0xF1-0xFF) K							
Commands Class LSB (0x00-0xFF) K							

Reports to follow (8 bits)

This value indicates how many report frames left before transferring the entire list of command classes.

Command Class (N * 8 or 16 bits)

The Command Class identifier.

Command Class Mark (8 bits)

The COMMAND_CLASS_MARK is used to indicate that all preceding command classes are supported, and all following command classes are controlled.

4.2.6 Security 2 (S2) Command Class, version 1

The Security 2 Command Class is a framework for allowing nodes to communicate securely in a Z-Wave network.

The Security 2 Command Class provides backwards compatibility to nodes implementing the Security 0 Command Class. Security 2 Command Class also defines a new encapsulation format, new Security Classes and a new KEX Scheme 1, which together offers a number of advantages over the Security 0 Command Class. Security 2 Command Class is scalable and allows more KEX Schemes, Security Classes and encapsulation formats to be introduced in the future if necessary.

Communication may be protected for a number of purposes. Known as CIA, the three main areas addressed by communications security are Confidentiality, Integrity and Authenticity. **Confidentiality** ensures that only the recipient can decode the communication. **Integrity** ensures that the recipient can determine if the communication has been modified or replayed. **Authentication** ensures that the communication really comes from the advertised sender.

Security 2 provides Confidentiality, Integrity and Authentication through:

- Key Exchange, that allows distribution of Network Keys in a Secure Network while preventing interception through:
  - Out-of-Band verification
  - Narrow time windows
  - Physical Activation
- Secure Message Encapsulation between nodes that have a shared network key.
  - A Pre-Agreed Nonce (PAN) is used to prevent Replay Attacks where communication is recorded and played back later, e.g. to unlock a door. In S2 a Nonce is exchanged once and then used to compute subsequent Singlecast PANs (SPAN) and Multicast PANs (MPAN), respectively.
  - Secure Singlecast communication, supports one-frame secure messages; allowing for lower latency and faster response times.
  - Secure Multicast communication, allows a sending node to simultaneously send secure messages to multiple nodes.
  - Cryptographic algorithms are intimately dependent on a perfect **Pseudo Random Number Generator** (PRNG). The PRNG is seeded with a unique noise pattern to ensure a unique starting point in the long sequence of random numbers generated by the PRNG. Many radio systems feature a special mode where the radio is capable of generating white noise which is not affected by the RF signal received from the antenna.

4.2.6.1 Compatibility Considerations

4.2.6.1.1 Command Class dependencies

CC:009F.01.00.21.001 Nodes supporting the Security 2 Command Class MUST also support the following command classes:

- *Transport Service Command Class, version 2*
- *Supervision Command Class, version 1*

CC:009F.01.00.21.002 In addition, nodes based on a controller Role Type MUST support the following Command Class:

- *Inclusion Controller Command Class, version 1*

4.2.6.1.2 Node Information Frame (NIF)

CC:009F.01.00.21.008 A supporting node MUST always advertise the Security 2 Command Class in its NIF, regardless of the inclusion status and security bootstrapping outcome.

CC:009F.01.00.21.00C A supporting node MUST NOT advertise the Security 2 Command Class in its S0/S2 Commands Supported Report list.

4.2.6.1.3 Mixed Security Classes

With the advent of the Security 2 Command Class, three new security classes have been added to the existing non-secure and Security 0. This makes for a total of five different security levels of which neither can communicate directly with each other.

In Security 0, this could be alleviated to some extent by allowing a device to choose to support certain of its command classes as non-secure. However, the impact of this is that a device is not entirely
CC:009F.01.00.22.001 secure. For this reason, command classes SHOULD NOT be supported non-securely by S0 enabled nodes if they leak information about the state of the device.

CC:009F.01.00.21.00B In Security 2, a node MUST support its command classes only when communication is using its highest Security Class granted during security bootstrapping.

The above rule does not apply to certain command classes, such as Transport Service or Z-Wave Plus Info, which must always be supported non-securely and present in the NIF if they are supported by a node. In this case, non-secure support requirements are specified in each individual Command Class definition. Command Classes present in the NIF (supported non-securely) MUST be supported at any granted Security Class level unless they are encapsulated outside security encapsulation or stated otherwise by another requirement (e.g. CC:0074.01.01.11.005,: DT:00.22.0006.1).

CC:009F.01.00.23.002 To allow inter-device communication for different security classes, the SIS (or Primary Controller) MAY perform Security Class elevation, by working as a middle-man and thus elevating the Security Class of a sending device to reach a different Security Class on a receiving device.

CC:009F.01.00.21.005 In case a forwarding rule is created from a lower Class to a higher Class, the UI MUST issue a warning to the user.

CC:009F.01.00.23.003 A node supporting S2 MAY control Command Classes at any of the granted Security Classes.

CC:009F.01.00.22.002 A controlling node attempting to communicate with a supporting node SHOULD try using its highest Security Class. If the communication is not successful and the controlling node has been granted
CC:009F.01.00.23.004 several Security Classes, the controlling node MAY try using any lower Security Classes.

4.2.6.1.4 Migration of existing devices to the Security 2 Command Class

An included node may be upgraded to support S2 via an OTA firmware update. Since non-secure operation as well as the Security 0 Command Class provide a lower level of trust, it is not possible to
CC:009F.01.00.21.006 automatically switch to S2 protection. Instead the Device Specific Key (DSK) MUST be generated by the running firmware on the device after being updated. Having the DSK generated internally, means it is not possible to input it directly on the S2 bootstrapping controller (for Access Control and Au-
CC:009F.01.00.21.007 thenticated Security Classes), instead it MUST be possible to input the DSK of the S2 bootstrapping controller on the joining node.

This process is described in further details in Section 4.2.6.2.6.

4.2.6.2 Security Considerations

4.2.6.2.1 Application enabled delivery confirmation

The use of SPAN and MPAN enables secure communication without preparations for each message. It is powerful, yet it requires application awareness. Safety and security related applications like door locks may require immediate command confirmations via the Supervision Command Class. Further, the risk of a delay attack can be mitigated through the use of the Supervision Command Class. This applies to S2 Singlecast as well as S2 Multicast transmissions.

A firmware update process may prefer to transfer firmware fragments as fast as possible while accepting the minor risk that the process stops for a moment in the unlikely event that the SPAN needs to be updated by the transmitter.

4.2.6.2.2 Potential Singlecast Delay Attack via interception and jamming

Since the SPAN is not limited by a timeout or synchronized clock, it is possible to perform a delay attack by intercepting an encrypted message while at the same time jamming the intended receiver. This way, the receiver SPAN is not incremented and the receiver will accept the encrypted message when an attacker decides to transmit the delayed message. This attack further requires that the attacker returns a MAC layer acknowledgement to the sender to avoid that the user gets error messages.

CC:009F.01.00.42.001 The Supervision Command Class SHOULD be used for S2 delivery acknowledgement. Only the intended receiver can respond correctly to a Supervision Get command.

4.2.6.2.3 Potential Multicast Delay Attack

Since the MPAN is not limited by a timeout or synchronized clock, it is possible to perform a delay attack by intercepting an encrypted message while at the same time jamming the intended receivers. This way the receiver MPAN is not incremented and the receivers will accept the encrypted message when an attacker decides to transmit the delayed message.

There is no way to distinguish this attack from simple radio interference phenomena as S2 Multicast messages are not acknowledged on the Z-Wave MAC layer.

While the singlecast follow-up message is primarily intended to ensure command execution in all multicast group members, the message also makes the receiver increment its MPAN inner state to invalidate previous multicast messages. This effectively eliminates an attacker's options for mounting a multicast delay attack.

S2 Multicast and singlecast follow-up messages are described in Section 4.2.6.5.3.

4.2.6.2.4 Circumventing DSK authentication

4.2.6.2.5 Controller-side authentication

The first 2 bytes of the public key of the joining node are obfuscated when requesting access to the "S2 Access Control" or "S2 Authenticated" class. The purpose of the DSK validation procedure is to force the user to enter information that can only be gathered from the joining device.

In other words, a user entering a part of the DSK proves to the including controller that the joining node is indeed the node that the user wants to add to the network. With this information, the including controller can construct the full public key of the joining node.

It is theoretically possible for the including controller to guess the complete public key of the joining node in less than 65536 calculations without user interaction. The including controller needs to try decrypting the received frame until a valid KEX Set (Echo) command is found.

The DSK verification is a device authentication step that ensures that a joining node can be trusted. The security of the system does not depend on the public key being secret but an including controller that skips authentication runs the risk of handing out network keys to joining nodes that cannot be trusted.

A man-in-the-middle attacker may establish a shared secret with the joining node by using the joining node's public key but the user's including controller will detect a mismatch between the DSK of the joining node and the first 2 or 16 bytes of the attacker's public key. Therefore the including controller rejects to complete the security bootstrapping before the network key is exposed to the attacker.

This applies to the S2 Access Control Class as well as the S2 Authenticated Class.

4.2.6.2.6 Client-side authentication

When upgrading existing devices to support Security 2 through an over-the-air (OTA) firmware update, there is no DSK printed on the node, which is required for S2 bootstrapping of the S2 Authenticated and S2 Access Control Classes.

In this case, a reverse verification procedure can be carried out, where the controller obfuscates the first 4 bytes of its public key and the joining node is input the controller's DSK in order to perform the authentication.

4.2.6.2.7 Protecting keys from physical extraction

A device may be mounted in an outdoor or public location. In such locations, there is a risk that the device is removed physically.

CC:009F.01.00.42.002 The hardware of the device SHOULD be designed to prevent the read-back of ECDH keys and network keys via debug connectors.

CC:009F.01.00.42.003 The hardware of the device SHOULD be designed to prevent the read-back of ECDH keys and network keys via a malicious firmware image.

CC:009F.01.00.42.004 The non-volatile memory used for storing security keys SHOULD be automatically cleared in case a firmware image is programmed in the NVM via the debug interface; only allowing keys to survive if firmware update is handled entirely via internal software APIs.

4.2.6.3 Interoperability Considerations

4.2.6.3.1 Pragmatic Decryption calculations in constrained environments

This specification recommends that a receiving node accepts incoming S2 Multicast frames which are up to 4 iterations into the future relative to the current MPAN inner state for the actual Group ID.

This specification also recommends that a sending node issues S2 Singlecast Follow-up frames after sending S2 Multicast frames. Decryption calculations may put a significant load on constrained processors. Thus, the receiving node may still be trying to decrypt a received S2 Multicast frame when an S2 Singlecast Follow-up is received.

CC:009F.01.00.31.001 A receiving node MUST respond correctly to an S2 Singlecast frame received immediately after an S2 Multicast frame.

CC:009F.01.00.32.001 A receiving node SHOULD monitor the arrival of S2 Multicast frames in order to avoid repeated MPAN synchronization in conditions where the S2 Multicast frame is only rarely received correctly.

4.2.6.4 Building Blocks

S2 functionality is relying on a number of building blocks for different parts of the protocol.

4.2.6.4.1 ECDH Key pair generation

Each S2 node has one or more ECDH key pairs used to setup a temporary secure channel for the Network Key exchange. Key pair generation is described in [1].

CC:009F.01.00.11.09D The ECDH private key MUST be created from 32 random bytes, which are generated using the PRNG function (Section 4.2.6.4.8).

The public key is calculated from the private key using Curve25519 [1].

4.2.6.4.2 Key exchange overview

In the following, the term Node A refers to the including node (typically a primary controller or SIS) while the term Node B refers to the joining node.

- **Inclusion Step 1: Create a shared secret between Node A and Node B**

  Both nodes calculate a shared secret based on an Authenticated Elliptic Curve Diffie Hellman key exchange (*AuthECDH*). Node A takes as input the Public Key of B, `KeyPub_B` and its own Private Key, `KeyPriv_A`. Node B takes as input the Public Key of A, `KeyPub_A` and its own Private Key, `KeyPriv_B`. Both returning the same `ECDH Shared Secret`.

  – *AuthECDH* is based on ECDH using Curve25519 [1]. Authentication is achieved through user verification as specified in Section 4.2.6.6.2.

- **Inclusion Step 2: Derive shared symmetric key for key exchange**

  To establish a temporary Network Key for AES128-CCM and CTR_DRBG, two steps are needed:

  – To convert the `ECDH Shared Secret` into a 16-byte Pseudo Random Key (`PRK`). *CKDF TempExtract* takes as input the `ECDH Shared Secret` along with `KeyPub_A` and `KeyPub_B`.

  – Temporary symmetric keys are derived based on CKDF-TempExpand, by giving the `PRK`, `KeyPub_A` and `KeyPub_B` as input. This returns the following keys:

  * Temporary CCM Key, combined Encryption and Authentication Key, denoted `TempKeyCCM`

  * Temporary Personalization String, denoted `TempPersonalizationString`.

- **Inclusion Step 3: Exchange permanent Network Keys**

  To exchange one or several Permanent Network Key (`PNK`), Singlecast Message Encapsulation is used with temporary symmetric derived keys (`TempKeyCCM` and `TempPersonalizationString`).

  – All Permanent Network Key Exchanges are carried out using the temporary symmetric key.

  – All Permanent CCM Keys, `KeyCCM`, `KeyMPAN` and `PersonalizationString`, are derived from the corresponding `PNK` using *CKDF-NetworkKeyExpand*

  – All `CKDF` functions are based on `AES128-CMAC`.

- **Singlecast Message Encapsulation:**

  The algorithm uses an authenticated encryption scheme conforming to AES128-CCM [13]. It is used to encrypt and authenticate secure payloads. AES128-CCM takes the `KeyCCM` and a `Nonce` as input. The algorithm returns a CCM Authenticated Ciphertext.

  - A Nonce is a "Number used Once". Nonces are initially exchanged between the two nodes and then mixed into a Mixed Entropy Input, `MEI`, using *CKDF-MEI-Extract* and *CKDF-MEI-Expand*. With both nodes holding the same `MEI`, they use the `MEI` and `PersonalizationString` as input into CTR_DRBG to generate a new Nonce.

- **Multicast Message Encapsulation:**

  Both singlecast and multicast frames use AES-128 CCM for encryption and authentication but multicast frames use a different algorithm to generate the IV. For multicast, the CCM IV is called the Multicast Pre-Agreed Nonce (`MPAN`).

CC:009F.01.00.11.001 MPANs are pushed from Node A, to multiple receiving nodes, B1, B2, etc. MPANs MUST be generated by Node A (see section Section 4.2.6.4.22) and MUST be distributed securely to each node B1, B2 etc. using singlecast messages.

CC:009F.01.00.11.002 Implementations of the Security 2 Command Class MUST provide AES-128 cryptographic services in the following modes of operation:

- **AES-128 "RAW"**

  This is also known as the AES-Electronic Code Book (ECB) and defines the basic operation of encrypting a 16 Byte Plaintext into a 16 Byte Ciphertext using an encryption key. AES-128 is used as a foundation for the following modes.

- **AES-128 CCM**

  The Counter with CBC-MAC (CCM) [13] is an authenticated encryption scheme.

- **AES-128 CMAC**

  The Cipher based Message Authentication Code (CMAC) is an essential part of the Key Derivation functions and used to mix Nonce contributions for SPAN synchronization.

- **AES-128 CTR_DRBG**

  The Counter mode Deterministic Random Byte Generator (CTR_DRBG) [14] is a block cipher based Pseudo Random Number Generator (PRNG) that is used to create Initialization Vectors and Network Keys.

CC:009F.01.00.11.003 In addition to the AES modes, the following building blocks MUST also be provided:

- **AuthECDH**

  Authenticated Elliptic Curve Diffie Hellman key exchange. Provides the temporary shared key material for protecting the network key exchange during inclusion.

4.2.6.4.3 Core AES (AES)

CC:009F.01.00.11.004 The security layer MUST implement the AES-128 encryption algorithm. This algorithm encrypts a single 128-bit block of plaintext using the Advanced Encryption Standard, AES. It receives a 128-bit key and a 128-bit plaintext block as input and produces a 128-bit cipher text block.

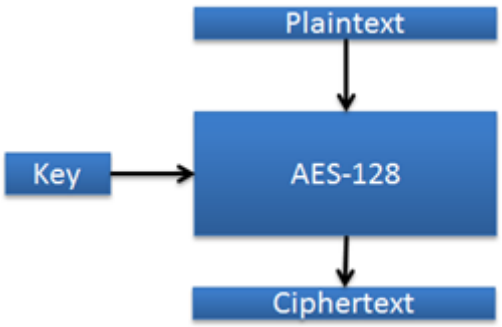


Figure 4.10: AES-ECB (Electronic Code Book)

4.2.6.4.4 AES-128 CCM Encryption and Authentication

CC:009F.01.00.11.005 The payload in the S2 Message Encapsulation Command MUST be encrypted and authenticated using AES-128 CCM. For details about AES-128 CCM and message decryption and validation, refer to [13] and RFC 3610.

CCM takes the following input:

- A combined encryption and authentication key denoted `KeyCCM`
- A Nonce
- A variable-length additional authenticated data structure (AAD)
- A variable-length payload to encrypt and authenticate

CCM yields as output:

- An encrypted version of the payload combined with an authentication tag covering the payload and the AAD

4.2.6.4.5 CCM profile

CC:009F.01.00.11.006 RFC 3610 defines a number of parameters that together determine the CCM profile used. S2 nodes MUST use the following parameter values for the CCM profile:

- Additional Authenticated Data (AAD) MUST be used (Adata = 1)

  The actual AAD is defined in Section 4.2.6.4.6.
- The Length field MUST be 2 bytes long (L = 2 bytes)
- The Authentication tag length MUST be 8 bytes (M = 8 bytes)
- The Nonce length MUST be 13 bytes (N = 13 bytes)

CC:009F.01.00.11.007 The following length requirements MUST be observed:

- AAD structures of up to 30 bytes MUST be supported. Larger AAD structures MAY be supported.

CC:009F.01.00.11.008 The 13 byte Nonce MUST be generated by taking the 13 most significant bytes of the *NextNonce* or Nonce0 as described in Section 4.2.6.4.14.

4.2.6.4.6 Additional Authenticated Data (AAD)

CC:009F.01.00.11.009 A structure MUST be constructed and used as AAD input for each CCM operation.

If both the Sender NodeID and Destination Tag are less or equal to 255, the following structure MUST be used as ADD:

7	6	5	4	3	2	1	0
Sender NodeID							
Destination Tag							
HomeID Byte 1							
...							
HomeID Byte 4							
Message Length Byte 1 (MSB)							
Message Length Byte 2 (LSB)							
Sequence Number							
Reserved						Enc Ext	Ext
Extension Data 1							
...							
Extension Data M							

If either the Sender NodeID or Destination Tag are greater than 255, the following structure MUST be used as ADD:

7	6	5	4	3	2	1	0
Sender NodeID (MSB)							
Sender NodeID (LSB)							
Destination Tag (MSB)							
Destination Tag (LSB)							
HomeID Byte 1							
...							
HomeID Byte 4							
Message Length Byte 1 (MSB)							
Message Length Byte 2 (LSB)							
Sequence Number							
Reserved						Enc Ext	Ext
Extension Data 1							
...							
Extension Data M							

Sender NodeID (1 or 2 bytes)

NodeID of the sending node.

Destination Tag (1 or 2 bytes)

The use of this field depends on the actual frame.

CC:009F.01.00.11.00A If the field is used for a Singlecast frame, this field MUST carry the Receiver NodeID.

If the field is used for an S2 Multicast frame, this field MUST carry the S2 Multicast Group ID.

HomeID (4 bytes)

HomeID of the sending node.

Message length (2 bytes)

This field indicates the total length in bytes of the Security 2 Message Encapsulation Command.

Sequence Number (1 byte)

Refer to the Security 2 Message Encapsulation Command (Section 4.2.6.5.11).

Ext (1 bit)

Refer to the Security 2 Message Encapsulation Command (Section 4.2.6.5.11).

Enc Ext (1 bit)

Refer to the Security 2 Message Encapsulation Command (Section 4.2.6.5.11).

Reserved

Refer to the Security 2 Message Encapsulation Command (Section 4.2.6.5.11).

Extension Data (M bytes)

CC:009F.01.00.11.00B This field MUST contain all non-encrypted extension objects.

CC:009F.01.00.11.00C This field MUST include the Length and Type fields prepending the actual data of each extension.

4.2.6.4.7 Message Authentication Code - AES-128 CMAC

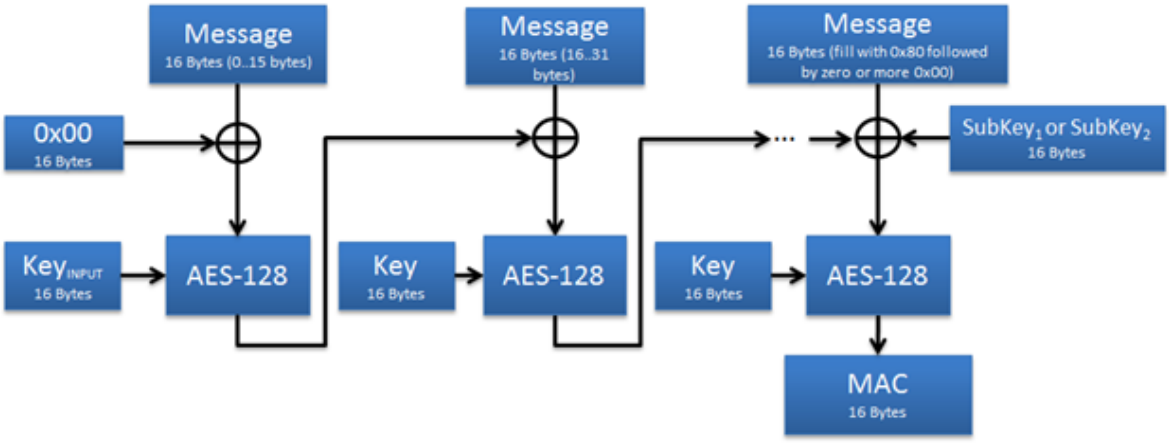


Figure 4.11: AES-128 CMAC building blocks

AES-128 CMAC is used for key derivation and Nonce mixing. The CMAC operation is specified in [15].

4.2.6.4.8 Pseudo Random Number Generator (PRNG)

CC:009F.01.00.11.015 The PRNG MUST be used for:

- Generating new network keys when provisioning a new network.
- Generating Nonce contributions for synchronizing the SPAN with peer nodes.

CC:009F.01.00.11.016 The PRNG MUST be implemented as an AES-128 CTR_DRBG as specified in [14]. The following profile MUST be used:

- No derivation function
- No reseeding counter
- Personalization string of 0x00 repeated 32 times
- Output length = 16 bytes
- security_strength is not used

CC:009F.01.00.11.017 The entropy_input [14] for instantiating the PRNG MUST be generated by a truly random source, e.g. white radio noise. The PRNG MUST be hardware seeded.

CC:009F.01.00.11.018 The inner state of the PRNG MUST be separated from the SPAN table.

4.2.6.4.9 Key extraction and derivation

The functions described in this section are used during key exchange.

4.2.6.4.10 CKDF-TempExtract

The *CKDF-TempExtract* function is used to extract the key entropy from the non-uniformly distributed ECDH Shared Secret.

```
CKDF-TempExtract(ConstantPRK, ECDH Shared Secret, KeyPub_A, KeyPub_B) ->
PRK
```

- The function's input is defined by:
  - `ConstantPRK = 0x33 repeated 16 times`
  - `ECDH Shared Secret is the output of the ECDH key exchange`
  - `Public Keys of Nodes A and B`
  - `PRK = CMAC(ConstantPRK, ECDH Shared Secret | KeyPub_A | KeyPub_B )`

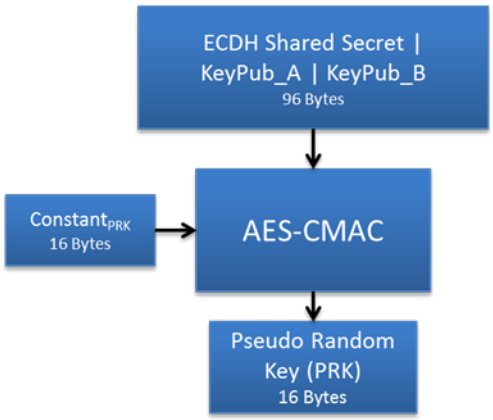


Figure 4.12: CKDF-TempExtract function block diagram

4.2.6.4.11 CKDF-TempExpand

CC:009F.01.00.11.08D Once the PRK has been computed, the temporary Authentication, Encryption and Nonce Keys MUST be derived using the *CKDF-TempExpand* function [10].

```
CKDF-TempExpand(PRK, ConstantTE) -> {TempKeyCCM, TempPersonalizationString}
```

- The function's input is defined by:
  - `PRK is calculated in the previous` Section 4.2.6.4.10
  - `ConstantTE = 0x88 repeated 15 times`
- Calculations are performed as follows:
  - `T1 = CMAC(PRK, ConstantTE | 0x01)`
  - `T2 = CMAC(PRK, T1 | ConstantTE | 0x02)`
  - `T3 = CMAC(PRK, T2 | ConstantTE | 0x03)`
- Output is defined as follows:
  - `TempKeyCCM = T1. Temporary CCM Key, combined Encryption and Authentication Key.`
  - `TempPersonalizationString = T2 | T3`

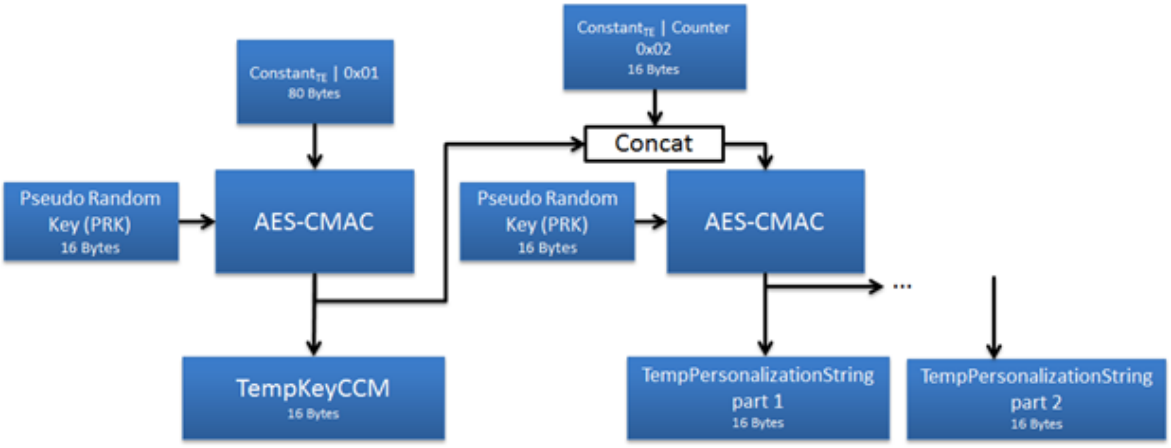


Figure 4.13: CKDF-TempExpand function block diagram

4.2.6.4.12 Permanent Key Exchange

A node that has been security bootstrapped into the network is characterized by being in possession of one or more Network Key(s). This key(s) is used throughout the lifetime of the network, typically measured in years to decades. Different nodes MAY have different Class Keys. A central controller
CC:009F.01.00.13.014 node MUST manage all the keys and select at inclusion time which class key(s) to share with a given
CC:009F.01.00.11.08E node.

CC:009F.01.00.11.08F Network Keys for all Security 2 Classes MUST be 16 random bytes, generated by using the PRNG function described in Section 4.2.6.7.

4.2.6.4.13 Key Derivation

Once the Network Key(s) has been exchanged, the `KeyCCM`, `PersonalizationString` and `KeyMPAN`
CC:009F.01.00.11.090 values MUST be derived using the *CKDF-NetworkKeyExpand* function [10].

```
CKDF-NetworkKeyExpand (PNK, ConstantNK) -> {KeyCCM, PersonalizationString,
KeyMPAN}
```

- The function's input is defined by:
  - `PNK is the permanent network key.`
  - `ConstantNK = 0x55 repeated 15 times`
- Calculations are performed as follow:
  - `T1 = CMAC(PNK, ConstantNK | 0x01)`
  - `T2 = CMAC(PNK, T1 | ConstantNK | 0x02)`
  - `T3 = CMAC(PNK, T2 | ConstantNK | 0x03)`
  - `T4 = CMAC(PNK, T3 | ConstantNK | 0x04)`
- Output is obtained by:
  - `KeyCCM = T1; CCM Key, combined Encryption and Authentication`
  - `PersonalizationString = T2 | T3`
  - `KeyMPAN = T4`

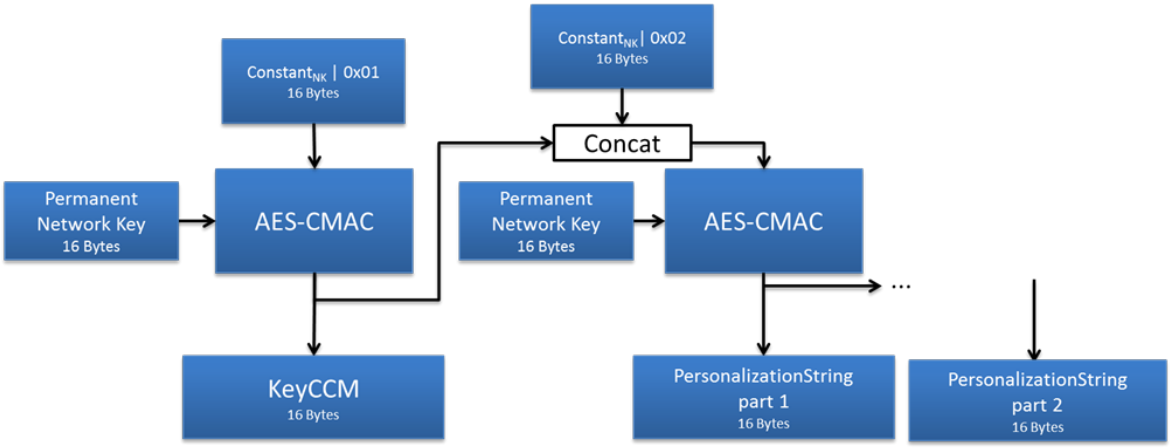


Figure 4.14: CKDF-NetworkKeyExpand function block diagram

4.2.6.4.14 Nonces for CCM

The Nonce is used for the CCM encryption in a Security 2 Message Encapsulation Command. Nonces provide these security benefits:

- Replay protection. Replaying an old message will result in the replay being discarded because the Nonce has already been used.
- The security proofs for CCM rely on the assumption that the same plaintext is never encrypted under the same key twice. Having unique Nonces upholds this assumption.

Singlecast and Multicast transport use different mechanisms for generating Nonces as described in the following sections.

Singlecast Pre-Agreed Nonces are referred to as `SPAN`.

Multicast Pre-Agreed Nonces are referred to as `MPAN`.

4.2.6.4.15 SPAN NextNonce Generator

CC:009F.01.00.11.00D Singlecast Nonces MUST be generated in the following way:

Skip steps 1 through 3 if a SPAN is already established

1. Exchange 16 bytes of Entropy between the Sender and the Receiver
2. Mix the entropy contributions
3. Instantiate CTR_DRBG and store the working state in the SPAN table
4. Generate 13 bytes of `Nonce` from the established `SPAN` using the *NextNonce* function.
5. Save the updated working state in the SPAN table

4.2.6.4.16 SPAN Instantiation

CC:009F.01.00.11.00E The Sender and Receiver MUST instantiate the CTR_DRBG as follows:

1. The Sender and Receiver MUST exchange 16 bytes of Entropy Input (EI), resulting in 32 bytes of shared entropy
   a. Both contributions MUST be generated using the PRNG (see Section 4.2.6.4.8)
2. Mix the 32 bytes EI into `MEI`, using *CKDF-MEI-Extract* and *CKDF-MEI-Expand* functions

CC:009F.01.00.11.00F The CTR_DRBG MUST be instantiated using the following profile:

- a. Entropy Input = `MEI` (obtained with *CKDF-MEI_Expand*)
  b. Personalization_String = `PersonalizationString`
  c. Output length = 16
  d. No derivation function
  e. No reseeding counter
  f. No security_strength
3. The CTR_DRBG now has its inner state set, referred to as the `InnerSPAN`

4.2.6.4.17 Mixing

Mixing of the two EIs is done by first calling *CKDF-MEI-Extract* with the EIs as input and using the result as input for *CKDF-MEI-Expand*

4.2.6.4.18 CKDF-MEI-Extract

```
CKDF-MEI-Extract(ConstNonce, SenderEI | ReceiverEI) -> NoncePRK
```

- The Input is defined by:
  - `ConstNonce = 0x26 repeated 16 times`
- The Output is obtained by:
  - `NoncePRK = CMAC(ConstNonce, SenderEI | ReceiverEI)`

4.2.6.4.19 CKDF-MEI-Expand

```
CKDF-MEI-Expand(NoncePRK, ConstEntropyInput) -> MEI
```

- The Input is defined by:
  - `NoncePRK is the pseudo random value obtained in the Extract step/`
  - `ConstEntropyInput = 0x88 repeated 15 times`
- The Output is obtained by:
  - `T0 = ConstEntropyInput | 0x00`
  - `T1 = CMAC(NoncePRK, T0 | ConstEntropyInput | 0x01)`
  - `T2 = CMAC(NoncePRK, T1 | ConstEntropyInput | 0x02)`
  - `MEI = T1 | T2`

4.2.6.4.20 Generation

CC:009F.01.00.11.010 With the CTR_DRBG having its inner state set, new `SPANs` MUST now be generated by subsequent calls to the *NextNonce* function.

4.2.6.4.21 NextNonce

The NextNonce function is defined as the CTR_DRBG_Generate_algorithm [14] with the following parameters:

- working_state = `InnerSPAN`
- requested_number_of_bits = 128
- additional_input = "" (empty string, i.e. zero bytes)

CC:009F.01.00.11.011 The *NextNonce* function MUST return a new `SPAN` for each call. The `InnerSPAN` MUST be stored in the SPAN table as described in Section 4.2.6.5.1.

CC:009F.01.00.11.012 The 16 bytes of Nonce MUST be truncated to the 13 most significant bytes before passing to the CCM module.

4.2.6.4.22 MPAN NextNonce Generator

Multicast Pre-Agreed Nonces (MPAN) are generated by encrypting successive values of a counter.
CC:009F.01.00.11.013 The initial value MUST be a 16 byte random number generated by the PRNG (Section 4.2.6.4.8). Whenever an MPAN is generated and consumed by the CCM module, the inner MPAN state is incremented.

MPAN instantiation and synchronization is described in Section 4.2.6.5.3.

4.2.6.4.23 MPAN Generation

MPAN generation is needed when a node sends or receives a secure multicast message. The generation
CC:009F.01.00.11.014 procedure, called *NextMPAN*, MUST comply with the following description:

1. The node reads the 16-byte inner MPAN state N from the MPAN table.
2. The node performs an *AES-128-ECB* encryption of the inner MPAN state N using `KeyMPAN` (obtained during *Key Derivation*) as key. The result is MPAN N.
3. The node increments the inner MPAN state N. The result is the inner MPAN state N+1 which is stored in the MPAN table. The increment operation MUST treat the inner MPAN state as an unsigned 16 byte integer. Thus, incrementing all ones MUST yield all zeros.

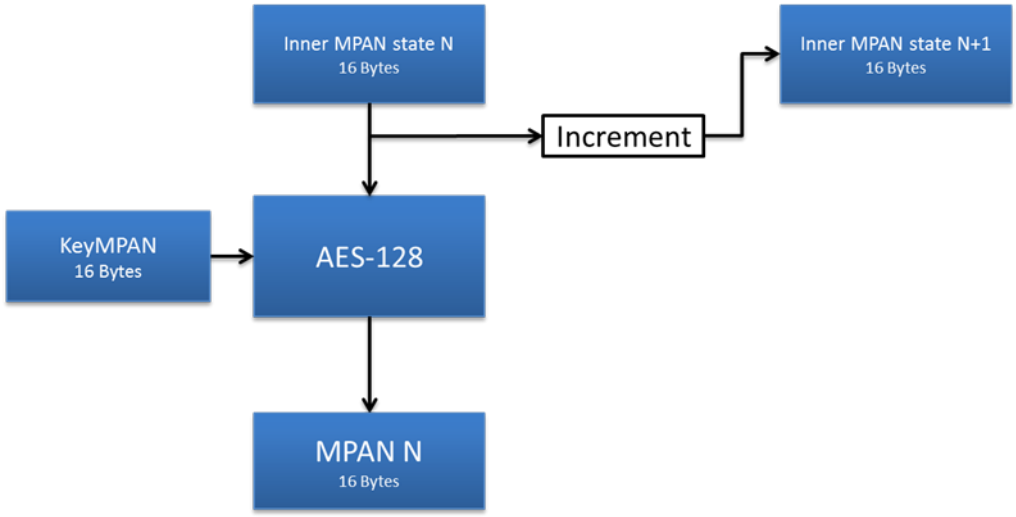


Figure 4.15: MPAN Generation

4.2.6.5 Message Encapsulation

The Security 2 Command Class supports Singlecast as well as Multicast communication

CC:009F.01.00.11.019 The S2 Transport Layer MUST NOT provide retransmission if the security layer discards a message due to SPAN synchronization failure or failed authentication.

CC:009F.01.00.12.001 An application SHOULD use the Supervision Command Class for delivery acknowledgement of Security 2 Encapsulated commands.

CC:009F.01.00.12.002 The Supervision Report command returns high-level status information on the execution status of the transmitted command which SHOULD be used by a controlling application instead of polling the destination node repeatedly.

4.2.6.5.1 Singlecast messages and SPAN Management

Compared to the Security 0 Command Class, the Security 2 Command Class provides a more efficient way to communicate securely. The Security 2 Command Class eliminates the frame overhead of the Security 0 Command Class challenge-response handshake.

CC:009F.01.00.11.01A Singlecast transport MUST comply with the steps described below:

CC:009F.01.00.11.01B 1. Before sending an encrypted frame, the **Sender** MUST establish a Singlecast Pre-Agreed Nonce (SPAN) with the **Receiver** if a SPAN is not already established.
    a. If a SPAN exists between the two nodes:
        i. Skip to step 3
    b. If the Sender is in possession of a matching Receiver's Entropy Input:
        i. The **Sender** uses its PRNG to generate a random 16-byte Nonce (SEI).
        ii. The **Sender** uses the combined 32-byte Sender's and Receiver's Entropy Input to instantiate a *NextNonce* Generator, hence creating the inner SPAN state. (described in Section 4.2.6.4.15)
        iii. Skip to step 3
    c. If no SPAN or matching Receiver's Entropy Input exists between the two parties:
        i. The **Sender** holds back the frame for subsequent transmission.
        ii. The **Sender** sends a Nonce Get to the **Receiver**

2. The **Receiver** uses its PRNG to generate a random 16-byte Nonce, store it in the SPAN table and send it in a Nonce Report to the **Sender**, with the Singlecast Out of Sync (SOS) flag set, to indicate that the frame contains a Receiver's Entropy Input (REI).
    a. The **Sender** stores the Receiver's Entropy Input in the SPAN table in the entry matching the receiver's NodeID.
    b. If the **Sender** has a pending frame for transmission then proceed to step 3. Otherwise no further action is taken.
3. The **Sender** constructs a Security 2 Message Encapsulation Command.
    a. If the inner SPAN state was just created in step 2.a, the SPAN extension containing the SEI is added to the S2 Message encapsulation Command to allow the **Receiver** to compute the same inner SPAN state.
    b. The **Sender** creates the next SPAN with the *NextNonce* function.
    c. The **Sender** uses the SPAN as IV for the AES-128 CCM encryption and authentication (Section 4.2.6.4.4) of the plaintext payload using `KeyCCM`.
    d. The Security 2 Message Encapsulation Command is transmitted to the **Receiver**.
4. The **Receiver** inspects the received Security 2 Message Encapsulation Command
    a. If the **Receiver** is unable to authenticate the singlecast message with the current SPAN, the **Receiver** SHOULD try decrypting the message with one or more of the following SPAN values, stopping when decryption is successful or the maximum number of iterations is reached. The maximum number of iterations performed by a receiving node MUST be in the range 1..5.

    CC:009F.01.00.12.003

    CC:009F.01.00.11.01C

    If the maximum number of iterations is reached without successful decryption, a Nonce Report MUST be sent to the **Sender** with the SOS flag set and containing a new REI. At the same time, the **Receiver** MUST invalidate the SPAN table entry for the Sender NodeID.

    CC:009F.01.00.11.01D

    CC:009F.01.00.11.01E

    b. If a SPAN Extension is present, the **Receiver** MUST:

    CC:009F.01.00.11.01F

        i. Instantiate a new SPAN Generator using the Receiver's Entropy Input stored locally and the Sender's Entropy Input just received.
        ii. Store the inner SPAN state in a SPAN table entry with the **Sender** as Peer NodeID.
        iii. Generate a SPAN by running the *NextNonce* function on the newly instantiated inner SPAN state.
        iv. Attempt authentication with the SPAN.
        v. If the authentication succeeds, skip to step 5.
        vi. If the authentication fails, the **Receiver** MUST go to step 2.

    CC:009F.01.00.11.020

    c. If the SPAN is already established and a SPAN Extension is not present, the **Receiver** MUST calculate a new inner SPAN state by passing the old inner SPAN state through the *NextNonce* function.

    CC:009F.01.00.11.021
5. After the **Receiver** has successfully authenticated the encapsulation command, the decrypted payload can be presented to the application layer.

The process is also illustrated in Figure 4.16.

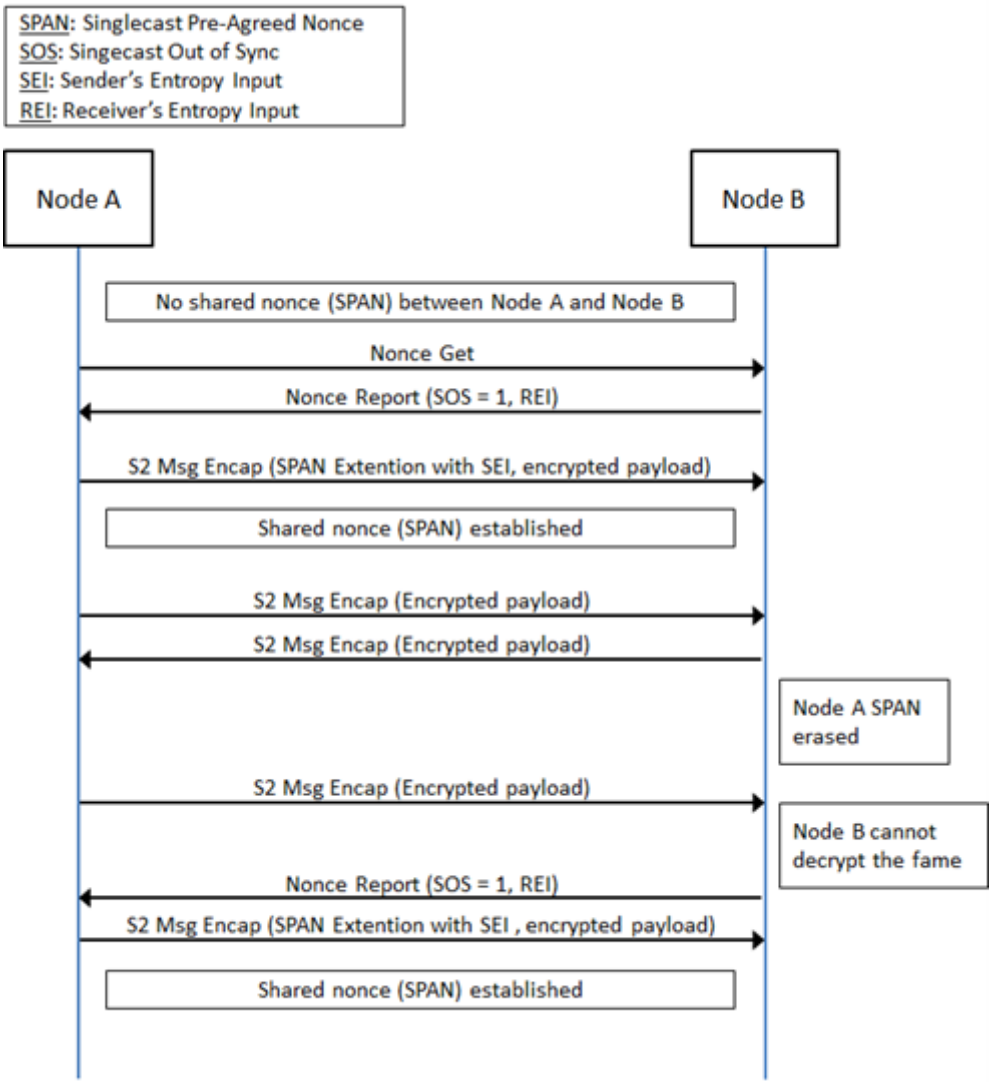


Figure 4.16: Singlecast communication frame flow: Span establishment

4.2.6.5.2 SPAN Table

CC:009F.01.00.11.022 A receiving S2 node MUST accept a SPAN table update even if the SPAN table is full.

CC:009F.01.00.12.004 It is RECOMMENDED that the node discards the least recently used entry from the SPAN table to make room for the new entry.

CC:009F.01.00.11.0A0 A supporting node MUST support at least 1 singlecast session (1 SPAN table entry).

Supporting always listening nodes MUST support at least 5 concurrent singlecast sessions (5 SPAN table entries).

CC:009F.01.00.12.005 The following format is RECOMMENDED:

Table 4.6: SPAN table format

7	6	5	4	3	2	1	0
Reserved		Security key				Nonce Type	
Sequence Number							
SPAN State Byte 1							
...							
SPAN State Byte 32							
Peer NodeID							

Security key (4 bits)

This field is used to remember which to Security key the SPAN is associated to. An example is given in Table 4.7.

Table 4.7: SPAN table::Security key

Value	Description
0x00	ECDH Temporary Key
0x01	S2.2: Access Control
0x02	S2.1: Authenticated
0x03	S2.0: Unauthenticated
0x04	S0

Nonce Type (2 bits)

This field is used to advertise the format of the SPAN State field.

CC:009F.01.00.12.006 The field SHOULD be encoded according to Table 8.

Table 4.8: SPAN table::Nonce Type

Nonce Type	Type	Description
0x00	Free	Table entry is not used
0x01	Receiver's Entropy Input (Locally Generated)	Bytes 1-16 are set to zero Bytes 17-32 contain the Receiver's Entropy Input
0x02	SPAN state	Bytes 1-32 contain the SPAN state

Sequence Number (1 byte)

This field is used to store the sequence number. Refer to description in *Security 2 Message Encapsulation Command*.

CC:009F.01.00.11.024 A receiving node MUST validate the Sequence Number and the actual sender's NodeID before accepting a SPAN table update.

SPAN State (32 bytes)

When the Nonce Type is "SPAN state", this contains the internal state of the *NextNonce* Generator. When the Nonce Type is Receiver's Entropy Input this field contains a locally generated Nonce awaiting the matching Sender's Entropy Input.

Peer NodeID (1 byte)

This field is used to store the NodeID of the corresponding peer.

4.2.6.5.3 Multicast messages and MPAN Management

The S2 Multicast protocol allows a sending node to reach a group of always listening nodes by using pre-agreed information for authentication and encryption.

The following terminology is used in this section:

- S2 SC = S2 Singlecast (carried in Z-Wave singlecast frame)
- S2 MC = S2 Multicast (carried in Z-Wave broadcast or multicast frame, with the MGRP extension)
- S2 SC-F = S2 Singlecast Follow-up (carried in Z-Wave singlecast frame, with the MGRP extension)

CC:009F.01.00.11.025 An S2 Multicast group MUST be represented by a unique (sender NodeID, Group ID) combination.

CC:009F.01.00.13.001 A multicast sender or group owner MAY maintain multiple S2 Multicast groups.

CC:009F.01.00.11.026

Sending and receiving nodes MUST maintain a unique Multicast Pre-Agreed Nonce (MPAN) value for each Group ID.

CC:009F.01.00.11.027 The MPAN is sender specific, and MUST NOT be used by the receiver to send S2 Multicast frames. S2 Multicast can only be performed within a group of nodes in the same S2 Security Class.

All nodes in the network will receive the S2 Multicast frame and will not try to decrypt it if the MPAN of the corresponding (sender NodeID, Group ID) is not known. The S2 Multicast protocol uses S2 SC-F frames for MPAN synchronization.

CC:009F.01.00.12.007 A sending node SHOULD send an S2 SC-F frame to each S2 Multicast group member after sending an S2 MC frame. This ensures that all group members receive the frame and at the same time eliminates the risk of the S2 MC frame being replayed in a delay attack.

CC:009F.01.00.12.008 S2 SC-F SHOULD be sent frequently in order to ensure that MPAN is synchronized at every group member.

S2 Multicast employs a self-healing algorithm for MPAN synchronization. The reception of an S2 SC F allows the receiving node to return an MPAN Out of Sync (MOS) indication (Either using a Nonce Report with the MOS flag or a S2 Message Encapsulation with the MOS extension, refer to Section 4.2.6.5.10, Section 4.2.6.5.16) if necessary.

CC:009F.01.00.12.009 Thus, a sending node SHOULD NOT push an up-to-date MPAN before sending an S2 Multicast frame to a new S2 Multicast receiver.

CC:009F.01.00.11.028 A sending node MUST maintain the MPAN inner state according to Table 4.9.

Table 4.9: Sending node MPAN maintenance

Frame types	Description	MPAN increase after transmission
S2 Singlecast	S2 SC frame	0
S2 Unacknowledged Multicast	S2 MC frame	1
S2 Acknowledged Multicast	S2 MC frame followed by an S2 SC-F frame to each S2 Multicast group member	3

CC:009F.01.00.11.029 A receiving node MUST maintain the MPAN inner state according to Table 4.10.

Table 4.10: Receiving node MPAN maintenance

Frame types	Description	MPAN increase after transmission
S2 Singlecast	S2 SC frame	0
S2 Multicast	S2 MC frame	1
S2 Singlecast Follow-up	S2 SC-F frame	1

CC:009F.01.00.13.002 If the Receiver is unable to decrypt the S2 MC frame with the current MPAN, the Receiver MAY try decrypting the frame with one or more of the subsequent MPAN values, stopping when decryption is successful or the maximum number of iterations is reached.

CC:009F.01.00.12.00A The maximum number of iterations performed by a receiving node MUST be 5.

CC:009F.01.00.11.02A If the maximum number of iterations is reached without successful decryption, the MOS flag MUST be set for the actual Group ID. The MOS state is reported back to the sending node only if the node receives a SC-F for the actual group.

CC:009F.01.00.11.0A1 An always listening node MUST support being member of at least 5 multicast groups at the same time (5 MPAN table entries)

MPAN synchronization and state maintenance examples are given in Figure 4.17 and Figure 4.18.

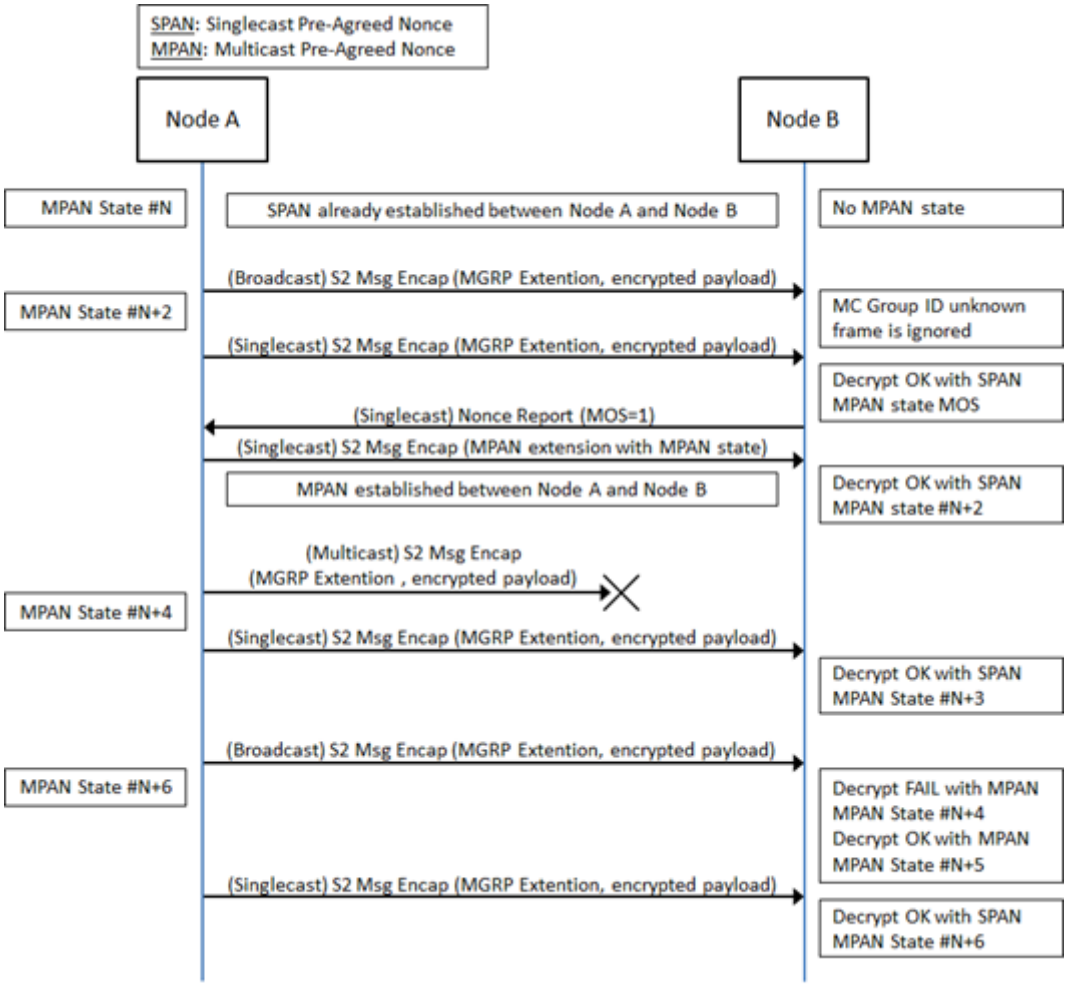


Figure 4.17: Next MPAN calculation example

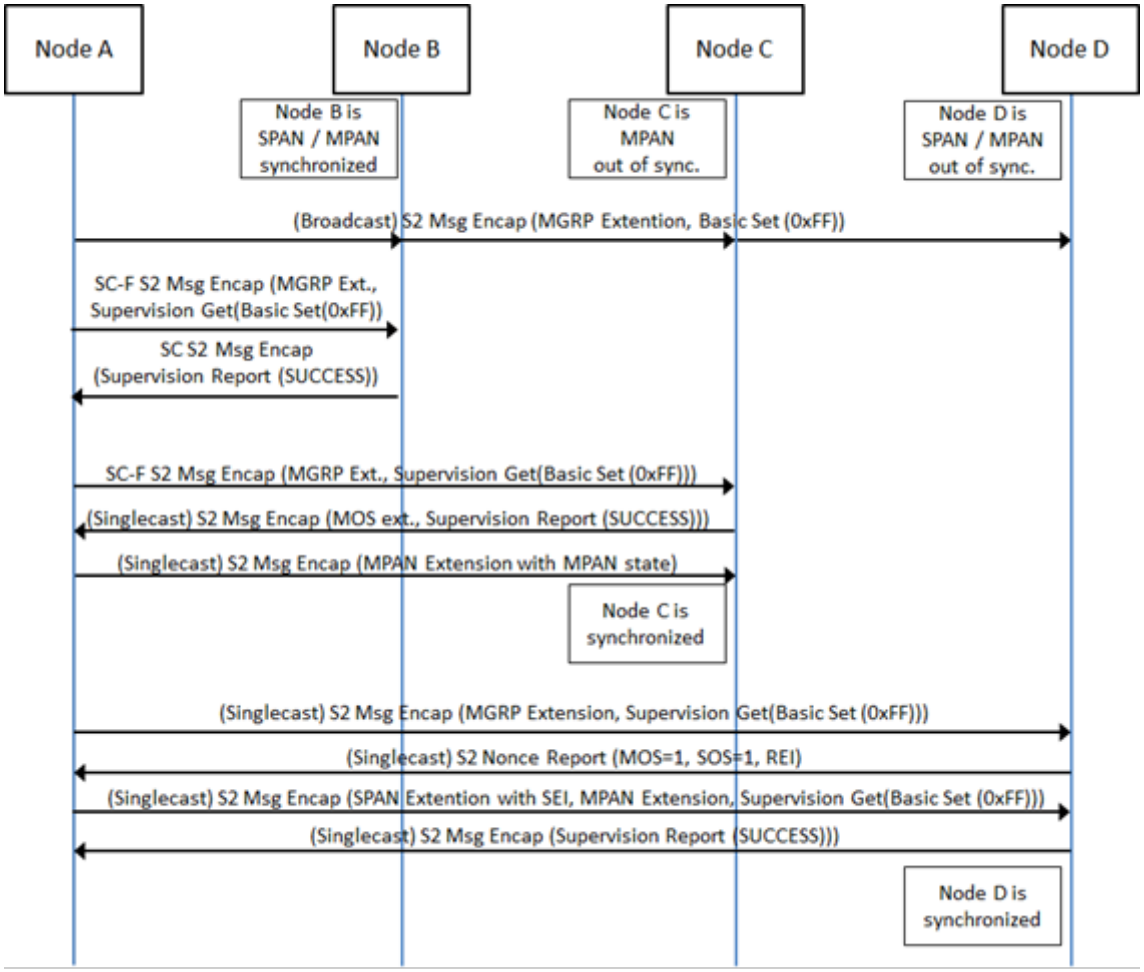


Figure 4.18: Multicast communication example with receivers out of sync and Supervision

Figure 4.18 shows Node C returning a Supervision Report with the MOS extension after the Supervision encapsulated Singlecast follow-up. In this case, a node MUST NOT return a Nonce Report with the MOS flag and subsequently another frame carrying the Supervision Report.

CC:009F.01.00.11.0A2

The following behavior is REQUIRED when a node is in MOS state and receives a Singlecast Follow-up:

CC:009F.01.00.11.0A3

- If the Singlecast Follow-up uses Supervision encapsulation, the receiving node MUST return a Supervision Report with the MOS extension.

CC:009F.01.00.11.0A4

- If the Singlecast Follow-up does not use Supervision encapsulation, the receiving node MUST return a Nonce Report with the MOS field set to 1.

CC:009F.01.00.11.0A5

4.2.6.5.4 MPAN table

CC:009F.01.00.12.00B A node SHOULD maintain the information indicated in Table 4.11 for each multicast group. The actual implementation format is out of scope of this specification.

Table 4.11: MPAN table entry, example

Information	Description
Owner NodeID	The NodeID of the node distributing this MPAN (rx only)
Group ID	Unique ID chosen by the owner node
MPAN Inner State	MPAN inner state used for encryption and decryption
MPAN entry state	MPAN_FREE • This entry is currently not in use MPAN_USED • This entry is currently in use MPAN_MOS • This entry is out of sync • Waiting to send Nonce report in respose to singlecast follow-up

CC:009F.01.00.11.0AC A supporting node MUST support at least 1 multicast session as a receiver (1 MPAN table entry).

4.2.6.5.5 Adding an S2 Multicast group member

A group member can be added to a S2 Multicast group by sending a Singlecast follow-up message to the new NodeID after a S2 Multicast message. The newly added NodeID will notify the sender that it is not synchronized for the given Group ID and it will be synchronized when receiving the next singlecast frame.

CC:009F.01.00.12.017 S2 Multicast will not work with sleeping nodes A sending node SHOULD NOT try to add sleeping nodes to a Multicast group.

4.2.6.5.6 Removing an S2 Multicast group member

A node can be removed from an S2 Multicast group by shuffling the MPAN state. At the next S2 Multicast frame, the newly removed node will set in MOS state and wait for re-synchronization in the next singlecast messages. When the newly removed node receives subsequent singlecast messages
CC:009F.01.00.11.099 without MPAN extension, the newly removed node MUST forget about the Multicast group ID.

4.2.6.5.7 Handling a missing S2 Multicast frame

An S2 MC frame may be missing due to simple radio interference or a deliberate delay attack.

In either case, the MPAN needs to be re-synchronized. Further, the MPAN needs to be advanced immediately to close the time window where an attacker can replay the stolen S2 MC frame.

The transmission of a singlecast follow-up frame ensures re-synchronization and prevents a potential delay attack. Figure 4.17 outlines an example of the MPAN state changes when an S2 MC frame is missing.

In the case both multicast and singlecast follow-up frames are intercepted for a delay attack, the attacker has the possibility to replay the multicast frame at a later time. Using the Supervision Command Class in singlecast follow-up frames prevent this attack as the receiving node needs to report the application level status in a new singlecast frame.

4.2.6.5.8 Message encapsulation commands

This section presents the commands of the Security 2 Command Class used for message encapsulation.

4.2.6.5.9 Security 2 Nonce Get Command

This command is used to request a fresh Nonce.

CC:009F.01.01.11.001 The Security 2 Nonce Report Command MUST be returned in response to this command.

CC:009F.01.01.11.002 A node sending this command MUST accept a delay up to <Previous Round-trip-time to peer node> + 250 ms before receiving the Security 2 Nonce Report Command.

CC:009F.01.01.11.003 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Security Header = SECURITY_2_NONCE_GET (0x01)							
Sequence Number							

Sequence Number (1 byte)

CC:009F.01.01.11.004 A sending node MUST specify a unique sequence number starting from a random value. Each message MUST carry an increment of the value carried in the previous outgoing message.

CC:009F.01.01.11.005 A receiving node MUST validate the Sequence Number and the actual sender's NodeID before accepting this command.

CC:009F.01.01.11.006 A receiving node MUST use this field for duplicate detection. Refer to Section 4.2.6.5.17.

4.2.6.5.10 Security 2 Nonce Report Command

This command is used to advertise a fresh Nonce in preparation for secure communication. This
CC:009F.01.02.11.001 command MUST NOT be issued unless it is done in response to the S2 Nonce Get or the S2 Message Encapsulation Command.

CC:009F.01.02.11.002 A sending node MUST set at least one of the MOS and SOS flags to the value '1' in this command. The command MUST NOT be sent if both the MOS and SOS flags are cleared.

CC:009F.01.02.11.003 A receiving node MUST update the MPAN and/or SPAN of the node sending this command by adding a SPAN and/or MPAN extension to the next Security 2 Message Encapsulation Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = SECURITY_2_NONCE_REPORT (0x02)							
Sequence Number							
Reserved						MOS	SOS
Receiver's Entropy Input Byte 1 (Optional)							
...							
Receiver's Entropy Input Byte N (Optional)							

Sequence Number (1 byte)

CC:009F.01.02.11.004 A sending node MUST specify a unique sequence number starting from a random value. Each message MUST carry an increment of the value carried in the previous outgoing message.

CC:009F.01.02.11.005 A receiving node MUST use this field for duplicate detection. Refer to Section 4.2.6.5.17.

Reserved

CC:009F.01.02.11.006 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

SOS - SPAN out of Sync (1 Bit)

CC:009F.01.02.11.007 When set by a sending node, the value '1' MUST indicate that the sending node does not have a SPAN established for the receiving node or was unable to decrypt the most recently received singlecast Security 2 Message Encapsulation Command from the destination of this command.

CC:009F.01.02.11.008 The value '0' MUST indicate that there was no problem decrypting the most recently received singlecast Security 2 Message Encapsulation Command.

CC:009F.01.02.11.009 If the SOS flag is set to '1', the REI field MUST be included in the command.

If the SOS flag is set to '0', the REI field MUST NOT be included in the command.

CC:009F.01.02.11.00A A receiving node MUST establish a new Singlecast Pre-Agreed Nonce (SPAN) and return a Security 2 Message Encapsulation Command with the SPAN extension, if this flag is set to '1' by a sending node.

MOS - MPAN Out of Sync (1 Bit)

CC:009F.01.02.11.00B When set by a sending node, the value '1' MUST indicate that the sending node does not have a MPAN state for Multicast group used in the most recently received singlecast follow-up Security 2 Message Encapsulation Command from the destination of this command.

CC:009F.01.02.11.00C The value '0' MUST indicate that there was no problem decrypting the most recently received multicast Security 2 Message Encapsulation Command that was followed by a singlecast follow-up.

CC:009F.01.02.11.00D A sending node MUST NOT advertise the MOS flag for a given (source NodeID, Group ID) tuple more than one time.

CC:009F.01.02.11.00E If this flag is set to '1' by a sending node, a receiving node MUST transfer the Multicast Pre-Agreed Nonce (MPAN) of the most recent singlecast follow-up transmission in a subsequent singlecast message if the sending node belongs to the multicast group. The MPAN MUST be transferred by sending a singlecast Security 2 Message Encapsulation Command with the MPAN extension.

Receiver's Entropy Input (REI) (16 Bytes)

This field is optional. When present, this field is used to carry the Receiver's Entropy Input in preparation for new S2 transmissions based on the SPAN. Refer to Section 4.2.6.5.1.

CC:009F.01.02.11.00F If the SOS flag is set to '1', the REI field MUST be included in the command.

If the SOS flag is set to '0', the REI field MUST NOT be included in the command.

4.2.6.5.11 Security 2 Message Encapsulation Command

CC:009F.01.03.11.001 The Security 2 Nonce Report Command MUST be returned in response to this command if the receiving node fails decrypting the command or if the message contains an MGRP extension with a Group ID that is unknown or out of sync (MOS).

CC:009F.01.03.13.001 This command MAY be issued via Z-Wave Multicast addressing.

CC:009F.01.03.11.002 A receiving node MUST NOT return a response if this command is received via Z-Wave Multicast addressing. The Z-Wave Multicast frame and the broadcast NodeID are both considered Z-Wave Multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Security Header = SECURITY_2_MESSAGE_ENCAPSULATION (0x03)							
Sequence Number							
Reserved						Encrypted Extension	Exten-sion
Extension 1 Length							
More to follow	Critical	Extension 1 Type					
Extension 1 Byte 1							
...							
Extension 1 Byte K							
Extension 2 Length							
More to follow	Critical	Extension 2 Type					
Extension 2 Byte 1							
...							
Extension 2 Byte L							
Encrypted Extension 1 Length							
More to follow	Critical	Encrypted Extension 1 Type					
Encrypted Extension 1 Byte 1							
...							
Encrypted Extension 1 Byte M							
Encrypted Extension 2 Length							
More to follow	Critical	Encrypted Extension 2 Type					
Encrypted Extension 2 Byte 1							
...							
Encrypted Extension 2 Byte N							
CCM Ciphertext object Byte 1							
...							
CCM Ciphertext object Byte P							

Sequence Number (1 byte)

CC:009F.01.03.11.003 A sending node MUST specify a unique sequence number starting from a random value. Each new message MUST carry an increment of the value carried in the previous singlecast command.

CC:009F.01.03.11.004 A receiving node MUST use this field for singlecast duplicate detection. Refer to Section 4.2.6.5.17.

Reserved

CC:009F.01.03.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

However, the value MUST be used as part of the AAD input (Section 4.2.6.4.6) used for authentication of the frame by the sending node as well as the receiving node.

Extension (1 bit)

This field is used to indicate if one or more non-encrypted extensions are included.

CC:009F.01.03.11.006 The value '1' MUST indicate that non-encrypted extensions are included.

The value '0' MUST indicate that non-encrypted extensions are not included.

Refer to the Extension object field description.

Encrypted Extension (1 bit)

This field is used to indicate if one or more encrypted extensions are included.

CC:009F.01.03.11.007 The value '1' MUST indicate that encrypted extensions are included.

The value '0' MUST indicate that encrypted extensions are not included.

Refer to the Encrypted Extension object field description.

[Extension Object] (N instances)

7	6	5	4	3	2	1	0
Extension Length							
More to follow	Critical	Type					
Extension Byte 1							
...							
Extension Byte L							

Extension Length (1 byte)

This field specifies the length of this extension, in bytes, including the "Extension Length" field.

Type (6 bit)

This field defines the type of this extension. Valid extension types are listed in Section 4.2.6.5.12.

Critical (1 bit)

CC:009F.01.03.11.008 A receiving node MUST discard the entire command if this flag is set to '1' and the Type field advertises a value that the receiving node does not support.

CC:009F.01.03.11.009 If this flag is set to '0' and the Type field advertises a value that the receiving node does not support, the actual extension MUST be ignored.

CC:009F.01.03.12.001 A receiving node SHOULD continue processing of the encapsulation command after the discarded extension.

More to Follow (1 bit)

CC:009F.01.03.11.00A If the More to Follow flag is set to '1', another Extension Object MUST follow this Extension Object.

Extension (Variable length)

This field carries the actual extension. Refer to Section 4.2.6.5.12. The length of this field
CC:009F.01.03.11.00B MUST comply with the length specified in the Extension Length field of this Extension Object.

[Encrypted Extension Object] (N instances)

CC:009F.01.03.11.00C The format of this object is identical to the unencrypted Extension Object. However, this object MUST be part of the encrypted payload.

CCM Ciphertext Object (P bytes)

This field contains an encrypted message. It comprises:

- (Optional) Complete Z-Wave command (comprising Command Class Identifier, Command Identifier and optional command payload)
- CCM control data
- CCM authentication tag.

The preceding Encrypted Extension fields are technically also a part of the CCM ciphertext object. But conceptually they are separate from the message and have been described as such.

4.2.6.5.12 Valid Extensions and Encrypted Extensions

The defined extension types are listed in Table 4.12.

Table 4.12: Security 2 Encapsulation Command::Extension Types

Extension Type Identifier	Format	Content Protection
0x01	SPAN Extension	Not Encrypted
0x02	MPAN Extension	Encrypted
0x03	MGRP Extension	Not Encrypted
0x04	MOS Extension	Not Encrypted

CC:009F.01.03.11.00D All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.2.6.5.13 SPAN Extension

The SPAN Extension is used by the sender to establish a SPAN by sending a Sender's Entropy Input to the Receiver. The combined Sender's and Receiver's Entropy Input may then be passed through the *NextNonce* function to generate the SPAN.

CC:009F.01.03.11.00E This extension MUST be sent unencrypted.

7	6	5	4	3	2	1	0
Length = 18							
More to follow	Critical = 1	Type = SPAN Extension = 1					
Sender's Entropy Input Byte 1							
...							
Sender's Entropy Input Byte 16							

Length (1 byte)

CC:009F.01.03.11.00F This field MUST be set to 18.

Type (6 bits)

CC:009F.01.03.11.010 This field MUST be set to the SPAN Extension type (1).

Critical (1 bit)

CC:009F.01.03.11.011 This flag MUST be set to '1'.

More to Follow (1 bit)

CC:009F.01.03.11.012 If this flag is set to '1', another unencrypted Extension Object MUST follow this unencrypted Extension Object.

If this flag is set to '0', this MUST be the last unencrypted Extension Object.

Sender's Entropy Input (16 bytes)

CC:009F.01.03.11.013 This field MUST carry the entropy input contribution used to instantiate a *NextNonce* Generator.

4.2.6.5.14 MPAN Extension

The MPAN Extension is used by the sender to establish or update an MPAN by sending the full 16 bytes MPAN state to the receiver to enable the reception of encrypted multicast transmissions.

CC:009F.01.03.11.014 This extension MUST be sent encrypted.

7	6	5	4	3	2	1	0
Length = 19							
More to follow	Critical = 1	Type = MPAN Extension = 2					
Group ID							
Inner MPAN state Byte 1							
...							
Inner MPAN state Byte 16							

Length (1 byte)

CC:009F.01.03.11.015 This field MUST be set to 19.

Type (6 bits)

CC:009F.01.03.11.016 This field MUST be set to the MPAN Extension type (2).

Critical (1 bit)

CC:009F.01.03.11.017 This flag MUST be set to '1'.

More to Follow (1 bit)

CC:009F.01.03.11.018 If this flag is set to '1', another encrypted Extension Object MUST follow this encrypted Extension Object.

If this flag is set to '0', this MUST be the last encrypted Extension Object.

Group ID (1 byte)

CC:009F.01.03.11.019 This field is used to identify MPAN instances. A sending node MUST set this value to the Group ID of the most recently transmitted multicast frame. The value MUST be in the range 0..255.

CC:009F.01.03.11.02D A sending node creating a new group SHOULD NOT use the value 0.

CC:009F.01.03.11.01A A receiving node MUST use this value in combination with the Owner NodeID for identifying the correct MPAN table entry when an MPAN table entry to use for frame decryption or MPAN updates.

CC:009F.01.03.11.01B If a receiving node already has an MPAN for the actual Group ID, that MPAN MUST be updated.

CC:009F.01.03.12.002 If this is a new Group ID, the new information MUST be stored. This may require the deletion of another MPAN table entry. It is RECOMMENDED that the least recently used entry is deleted.

Inner MPAN state (16 bytes)

This field carries the value to be used in the encryption and decryption of the next S2 Multicast frame.

4.2.6.5.15 MGRP Extension

CC:009F.01.03.11.01C The Multicast Group (MGRP) Extension MUST be included in all S2 Multicast and S2 Singlecast follow-up frames.

CC:009F.01.03.11.01D A receiving node MUST use the Group ID to select the MPAN for decrypting this message.

CC:009F.01.03.11.01E When receiving this extension, the matching MPAN (if found) MUST be incremented by one after decryption.

CC:009F.01.03.11.02B If the Group ID is not found in the receiver MPAN table and the received frame is a singlecast (follow-up), the node MUST return a Nonce Report with the MOS flag set, or alternatively, return another Security 2 Message Encapsulation Command with the MOS extension included.

CC:009F.01.03.11.020

This extension MUST NOT be sent together with the MPAN extension.

CC:009F.01.03.11.021 This extension MUST be sent unencrypted.

7	6	5	4	3	2	1	0
Length = 3							
More to follow	Critical = 1	Type = MGRP Extension = 3					
Group ID							

Length (1 byte)

CC:009F.01.03.11.022 This field MUST be set to 3.

Type (6 bits)

CC:009F.01.03.11.023 This field MUST be set to the MGRP Extension type (3).

Critical (1 bit)

CC:009F.01.03.11.024 This flag MUST be set to '1'.

More to Follow (1 bit)

CC:009F.01.03.11.025 If this flag is set to '1', another unencrypted Extension Object MUST follow this unencrypted Extension Object.

If this flag is set to '0', this MUST be the last unencrypted Extension Object.

Group ID (1 byte)

This field is used by the sender to uniquely identify which MPAN was used to encrypt this multicast frame.

4.2.6.5.16 MOS Extension

The MOS extension is used to indicate that the sending node does not have a MPAN state for the Multicast group used in the most recently received singlecast follow-up Security 2 Message Encapsulation Command from the destination of this command

CC:009F.01.03.13.002 The receiver MAY choose to re-synchronize the sending node by returning a new Security 2 Message Encapsulation Command with a MPAN extension included.

CC:009F.01.03.11.026 This extension MUST be sent unencrypted.

7	6	5	4	3	2	1	0
Length = 2							
More to follow	Critical = 1	Type = MOS Extension = 4					

Length (1 byte)

CC:009F.01.03.11.027 This field MUST be set to 2.

Type (6 bits)

CC:009F.01.03.11.028 This field MUST be set to the MOS Extension type (4).

Critical (1 bit)

CC:009F.01.03.11.02C This flag MUST be set to '0'.

More to Follow (1 bit)

CC:009F.01.03.11.02A If this flag is set to '1', another unencrypted Extension Object MUST follow this unencrypted Extension Object.

If this flag is set to '0', this MUST be the last unencrypted Extension Object.

4.2.6.5.17 Duplicate Message Detection

CC:009F.01.00.11.0AD A sending node MUST set the Sequence Number to a random value on startup. The Sequence Number MUST be incremented for the transmission of each new unique singlecast (and singlecast follow-up) transmission.

CC:009F.01.00.11.02D A receiving node MUST use the Sequence Number field in the Nonce Get, Nonce Report and Message Encapsulation commands for duplicate detection.

CC:009F.01.00.11.02E Duplicates of recently received commands MUST be discarded.

CC:009F.01.00.12.00C The following algorithm SHOULD be used for duplicate detection of singlecast messages:

CC:009F.01.00.11.02F 1. If an incoming sequence number and Peer NodeID match an entry in the SPAN table, the frame MUST be discarded.

CC:009F.01.00.11.030 2. If it is a new sequence number, the received sequence number and Peer NodeID MUST be stored in the SPAN table.

4.2.6.6 Key Management

CC:009F.01.00.11.033 S2 communication relies on two separate derived keys that MUST be shared between any nodes communicating with each other; the combined Encryption and Authentication Key denoted KeyCCM and the CTR_DRBG Key denoted `PersonalizationString`. Both keys MUST be derived from one Network Key that is exchanged between the nodes using the *CKDF-NetworkKeyExpand* function detailed in Section 4.2.6.4.13.

In S2, not all nodes share the same network key. S2 separates devices into different Security Classes, so that if one Security Class is compromised, it does not affect other Security Classes. The joining
CC:009F.01.00.11.034 node MUST advertise the supported Security Classes during the S2 bootstrapping process.

CC:009F.01.00.13.003 The including node MAY grant membership of one or more Security Classes.

CC:009F.01.00.11.035 The joining node MUST then request the granted keys, one at a time.

The Security Classes are listed in Table 4.13.

S2 Access Control and S2 Authenticated Security Classes are equivalent from a technical perspective, but do not share the same network key. This is simply to prevent compromising the Access Control key if the Authenticated key is compromised.

Client-side authentication is an alternative authentication mechanism intended for the authenticated security bootstrapping of devices which do not have a DSK. Refer to Section 4.2.6.6.3.

Table 4.13: S2 Security Class Overview

Value	Class Name	Example Devices	Controller authentication (Refer to Section 4.2.6.6.2)		Client-side authentication (Refer to Section 4.2.6.6.3)	
			Display DSK	Input DSK	Display DSK	Input DSK
2	S2 Access Control	Door locks, Garage doors	14 bytes	5 decimal digits (2 bytes)	12 bytes (Optional)	10 decimal digits (4 bytes)
			None (QR Code)	QR Code (16 bytes)	None (QR Code)	QR Code (16 bytes)
1	S2 Authenticated	Lighting and Sensors (Managed systems and security systems)	14 bytes	5 decimal digits (2 bytes)	12 bytes (Optional)	10 decimal digits (4 bytes)
			None (QR Code)	16 bytes (QR Code)	None (QR Code)	16 bytes (QR Code)
0	S2 Unauthenticated	Lighting and Sensors (Unmanaged systems)	0 to 16 bytes	0	0	0
7	S0	Legacy door locks	N/A	N/A	N/A	N/A

CC:009F.01.00.11.036 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The manufacturer decides which classes are relevant to advertise for the intended use of the product. For instance:

- A light dimmer device may advertise the S2 Unauthenticated Class and the S2 Authenticated Class as the intended classes. If a constrained key fob without display is used to create a small system, the wall controller may grant only the S2 Unauthenticated Class to the light dimmer, which then requests only the S2 Unauthenticated Class key.
- A light bulb may advertise the S2 Unauthenticated Class as its only intended class because it does not provide a DSK. Depending on the configured security level, an authentication capable gateway may accept including S2 Unauthenticated Class devices or it may reject including S2 Unauthenticated Class-only devices.
- A door lock device may advertise the S2 Access Control Class as its only intended class because it requires the highest protection level. A constrained key fob, which can grant only the S2 Unauthenticated Class key, rejects including the door lock device and returns an error indication to the user because it cannot authenticate the door lock.

4.2.6.6.1 Key Exchange

The S2 key exchange MUST comply with Table 4.14.

Table 4.14: Key exchange and key verification

Key to be exchanged	Key to use for the Key Exchange	Key to use for the Key Verification
S2.2: Access Control	ECDH Temporary Key	S2.2
S2.1: Authenticated	ECDH Temporary Key	S2.1
S2.0: Unauthenticated	ECDH Temporary Key	S2.0
S0	ECDH Temporary Key	S0

CC:009F.01.00.11.039 A number of Security Class keys may be granted to the joining node. A temporary key and SPAN MUST be used for the exchange of the Security Class keys.

CC:009F.01.00.11.03A

The temporary SPAN MUST be initialized with the Shared Nonce that is established after the S2 temporary key is established.

CC:009F.01.00.11.03B The SPAN value MUST be updated after each Security Class key exchange.

CC:009F.01.00.11.03C Verification of an assigned key (including S0) MUST always use the newly exchanged key to encrypt the S2 verification message.

CC:009F.01.00.11.040 If a joining node is unsuccessful requesting all the Security Class keys that it was granted, the node MUST abort the S2 bootstrapping entirely.

CC:009F.01.00.11.0B4 The S0 key can be exchanged with the Security 0 Command Class or with the Security 2 Command Class.The S0 message encapsulation MUST be done with the Security 0 Command Class encapsulation Command after S2 bootstrapped is completed.

4.2.6.6.2 ECDH key pairs, Device Specific Key and User Verification

A Device Specific Key (DSK) is used to protect against man-in-the-middle attacks (MITM) where a malicious attacker tries to intercept and manipulate the key exchange.

CC:009F.01.00.11.0BA A node supporting S2 MUST have a first **Learn Mode ECDH key pair**:

- This key pair is used for joining a network both with and without authentication during S2 Bootstrapping. However, if the node also has a second Learn Mode ECDH key pair, then the first key pair is only used when the S2 Bootstrapping requires authentication.
- This key pair MUST be static.

CC:009F.01.00.13.019 A node supporting S2 MAY have a second **Learn Mode ECDH key pair**:

- **This key pair is used for joining a network when the S2 Bootstrapping does not require authentication.**
- This key pair MAY be dynamic.

CC:009F.01.00.11.0BB Additionally, a node able to perform S2 bootstrapping MUST have an additional separate **Add Mode ECDH key pair**:

- This key pair is used for adding nodes into a network (controller side).
- This key pair MUST be dynamic.

CC:009F.01.00.11.0B7 A unique Authenticated Learn Mode ECDH key pair MUST be assigned to each individual node, if they request an Authenticated Security Class.

CC:009F.01.00.13.018 A node MAY create a new Unauthenticated Learn Mode ECDH key pair for every S2 bootstrapping attempt.

CC:009F.01.00.11.0B8 A node MUST keep the same Authenticated Learn Mode ECDH key pair for the lifetime of the node.

CC:009F.01.00.11.0B9 The DSK is defined as the first 16 bytes of the Authenticated ECDH Public Key of a node. An S2 node MUST respect the DSK requirements listed in [34].

The DSK may be used for out-of-band (OOB) authentication in two ways.

- The including controller uses QR code scanning to read the entire DSK off the joining device and match it with the obfuscated public key received via RF from the joining device.
- Else the including controller asks the user to visually validate that the rest of the DSK matches with the Public Key received via RF. The including controller additionally asks the user to enter the PIN code (the 5 first digits of the DSK string) in order to substitute the obfuscated bytes of the joining node's Public Key.

CC:009F.01.00.12.011 An including controller with support for the S2 Authenticated Class or the S2 Access Control Class SHOULD provide a QR code scanning capability for user friendly inclusion.

CC:009F.01.00.11.04E

If scanning capability is not available, the including controller MUST provide an interface that allows the user to enter a 5 digit PIN code and perform visual DSK string validation.

CC:009F.01.00.11.04F The requested PIN code MUST be the first 5 decimal digits of the DSK string.

4.2.6.6.3 Client-Side Authentication

CC:009F.01.00.11.06E When upgrading existing devices to support Security 2 through an over-the-air (OTA) firmware update, there is no DSK printed on the node and the upgraded node MUST therefore generate its public key and DSK internally with the updated firmware.

CC:009F.01.00.13.00C A device MAY request the use of Client-Side authentication (CSA) if it does not possess a DSK label. If the joining node requests CSA, the including controller MUST ask the user if this should be permitted, and if permitted, the including controller MUST display its own DSK.
CC:009F.01.00.11.06F

CC:009F.01.00.11.070 As stated in Table 4.13, the first 10 digits MUST be input on the Joining Node, meaning it MUST have a method of input, like a keypad.

CC:009F.01.00.11.071 Compared to controller-side authentication where the entire DSK MUST also be visually verified, CSA does come with a higher risk of having the bootstrapping attempt manipulated by an attacker. The attacker, however, has to guess 4 random bytes in one attempt.

4.2.6.6.4 Initial Key Exchange

CC:009F.01.00.11.0A6 Add mode MUST be initiated physically. Physical activation includes button press, applying power, remote user activation, etc.

CC:009F.01.00.11.052 Initial Key Exchange MUST be carried out using the ECDH temporary key with the including controller.

CC:009F.01.00.11.053 The Including Node A MUST create a new Add Mode ECDH Key Pair for each new bootstrapping process (regardless of the outcome of each bootstrapping attempt), if it supports the S2 Access Control and/or S2 Authentication Security Classes.

CC:009F.01.00.11.055 The Key Exchange process MUST follow the frame flow illustrated in Figure 4.19.

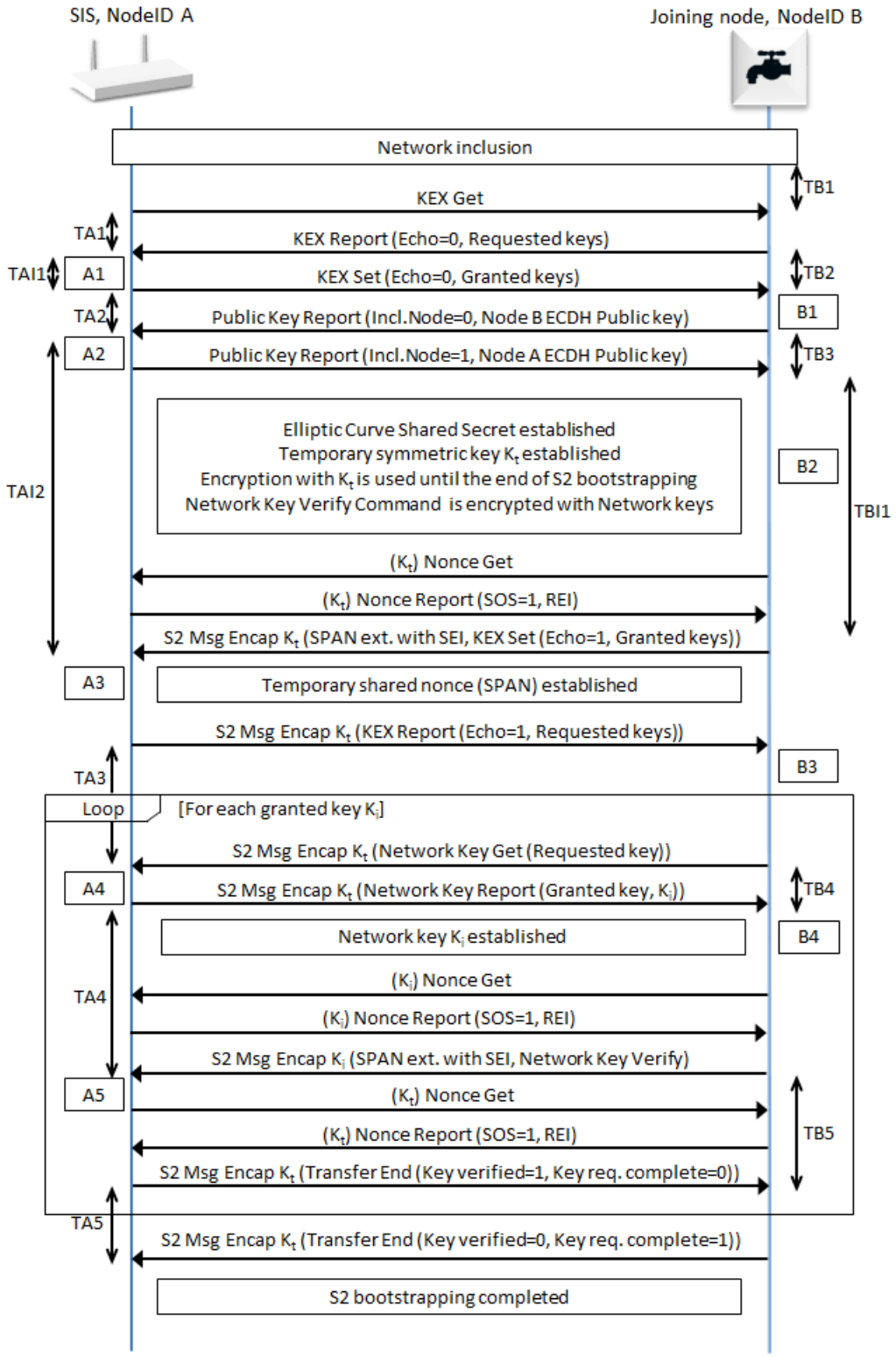


Figure 4.19: S2 bootstrapping frame flow

CC:009F.01.00.11.056 The key exchange MUST comply with the following steps:

1. **Network inclusion completed:**

   CC:009F.01.00.11.057 Immediately following a successful network inclusion or after receiving an Inclusion Controller Initiate Command (refer to Section 5.2.1.2), the Security 2 enabled controller A MUST start the S2 bootstrapping

2. **A → B: KEX Get:**

Including Node A, requests KEX Report from Joining Node B.

3. **B → A: KEX Report:**

   Sent as response to the KEX Get command, this command contains:

   a. Scheme Report - Schemes supported by Node B

   b. Curve Report - Elliptic Curves supported by Node B

   c. Requested Key List - List of Keys (such as S2 Class keys and Security 0 Network Key) which is requested by Node B

   d. Request for Client-Side authentication

4. **A1:**

   CC:009F.01.00.11.058 Node A MUST verify the KEX Report and, if required, cancel the S2 bootstrapping as described in Section 4.2.6.6.5.

   a. CC:009F.01.00.13.007 CC:009F.01.00.11.059 **Optional:** Node A MAY present a dialog allowing the installer to select which specific keys will be granted to Node B. If presented, the installer MUST either confirm a list of granted keys or cancel the security bootstrapping.

   b. CC:009F.01.00.11.0B5 If Client-Side authentication is requested and Node A supports inclusion using CSA, Node A MUST present a dialog asking if Client-Side authentication should be allowed.

      i. CC:009F.01.00.11.05B If Client-Side authentication is used, Node A MUST subsequently present a dialog with its own DSK.

      ii. CC:009F.01.00.13.016 CC:009F.01.00.11.093 Node A MAY reject Client-Side authentication. In this case, Node A MUST either abort the S2 bootstrapping with a KEX_FAIL_CANCEL or only grant a subset of keys that does not require CSA, e.g. Security 0 and Unauthenticated.

5. **A → B: KEX Set:**

   CC:009F.01.00.13.008 The KEX Set Command contains parameters selected by Node A. The list of class keys MAY be reduced to a subset of the list that was requested in the previous KEX Report from Node B.

   a. Scheme Set - The selected Scheme by Node A for bootstrapping

   b. Curve Set - The selected Curve by Node A for bootstrapping

   c. Key List Set - The list of granted keys to Node B by Node A

   d. Client-Side authentication - Indicating whether Client-side authentication will be used

6. **B1:**

   CC:009F.01.00.11.05D Node B MUST verify the KEX Set command and, if required, cancel security bootstrapping as described in Section 4.2.6.6.5.

   a. CC:009F.01.00.13.009 CC:009F.01.00.11.05E CC:009F.01.00.13.00A **Optional:** Node B MAY present the Node A's DSK for verification or input. If presented, installer MUST either confirm or cancel security bootstrapping. This step MAY also be carried out in step B2 instead is CSA is used.

   b. CC:009F.01.00.11.092 Node B MUST accept what it has been granted, even if it is only a subset of what was requested.

   c. CC:009F.01.00.11.091 If Node A wrongfully grants CSA without being requested, Node B MUST cancel security bootstrapping.

7. **B → A: Public Key B:**

   Public Key B is the ECDH Public Key of Node B and is used for the ECDH Key Exchange.

   a. If Authentication is required (KEX Set granted an Authenticated Security Class):

      CC:009F.01.00.11.0BC Node B MUST use its Authenticated ECDH Public key and the DSK bytes 1..2 MUST be obfuscated by zeros.

b. If Authenticated is not required (KEX Set granted no authenticated Security Class):

   Node B MUST use its Unauthenticated ECDH Public Key and transmit the full key non-obfuscated.

8. **A2:**

CC:009F.01.00.11.05F If authentication is required, Node A MUST request that the user enters the PIN code or scans the QR code from Node B in order to verify the DSK (refer to Section 4.2.6.6.2 and Section 4.2.6.6.5).

   a. If Node A was input a PIN code, it MUST substitute the bytes 1 and 2 of the Node B public key with the 2 bytes received in the PIN code. The user MUST be prompted a dialog to visually validate the bytes 3..16 of Node B's DSK.

   b. If Node A has received the 16 bytes DSK of Node B via QR scanning, it MUST substitute the first 16 bytes of Node B's Public Key with the 16 bytes received via QR code.

CC:009F.01.00.12.012 c. Node A SHOULD continue verifying the DSK input until A3 to allow ECDH calculations to take place while the user is verifying the DSK.

9. **A → B: Public Key A:**

   Public Key A is the Elliptic Curve Public Key of Node A and will be used for the temporary ECDH Key Exchange.

   a. **Mandatory:** If Client-Side authentication is used, the DSK bytes 1..4 MUST be obfuscated by zeros.

10. **B2:**

   a. **Mandatory:** If Client-Side authentication is used, the DSK MUST be input on Node B.

      i. If Node B was input a PIN code, it MUST substitute the 4 first bytes of Node A's Public Key with the 4 bytes received in the PIN code. The user MAY be prompted a dialog to visually validate the bytes 5..16 of Node A's DSK.

      ii. If Node B has received the 16 bytes DSK of Node A via QR scanning, it MUST substitute the first 16 bytes of Node A's Public Key with the 16 bytes received via QR code

11. **Elliptic Curve Shared Secret Established:**

   If B2 is passed, Node A and Node B have performed an ECDH Key Exchange, resulting in an Elliptic Curve Shared Secret.

12. **Temporary Symmetric Key Established:**

   Both Node A and Node B derive a Temporary Symmetric Key from the ECDH Shared Secret based on CKDF-TempExpand (refer to Section 4.2.6.4.11).

13. **B → A: Nonce Get:**

   Node B requests a Nonce from Node A that will allow Node B to send messages securely using the Temporary Symmetric Key.

14. **A → B: Nonce Report:**

   A's Nonce

CC:009F.01.00.11.062 15. *From this point all frames sent between Node A and Node B MUST be encrypted using the ECDH Temporary Symmetric Key (With the exception of Nonce Get / Report for each Security Class which MUST NOT be encrypted and the Network Key Verify Command, which MUST be encrypted with the most recently exchanged key. Refer to* Section 4.2.6.6.1*).*

16. **B → A: KEX Set (echo):**

   The KEX Set command received from Node A in step 5 is confirmed via the temporary secure channel.

CC:009F.01.00.11.097 a. The frame MUST be retransmitted by node B every 10 seconds for 240 seconds in total until either event is detected:

i. Step 17, KEX Report(Echo) is received

ii. KEX Fail is received

b. If neither events are detected after 240 seconds, Node B times out and aborts S2 bootstrapping silently

17. **A3:**

CC:009F.01.00.11.063 Node A MUST abort S2 bootstrapping if the KEX Set(Echo) received in step 16 is not identical to KEX Set previously sent by Node A in step 5. Refer to Section 4.2.6.6.5.

18. **A → B: KEX Report (echo):**

The KEX Report Command received from Node B in step 3 is confirmed via the temporary secure channel.

19. **B3:**

CC:009F.01.00.11.064 Node B MUST abort S2 bootstrapping if the KEX Report(Echo) received in step 18 is not identical to KEX Report previously sent by Node B in step 3. Refer to Section 4.2.6.6.5.

CC:009F.01.00.11.098 If a Node B node has been granted zero keys, Node B MUST go to step 30 and indicate to Node A that it is terminating the bootstrapping process by returning an S2 Transfer End Command. Node A and Node B will consider Node B to be included non-securely at the end of the bootstrapping.

CC:009F.01.00.11.065 *Authentication has been completed, and network key exchange begins. Steps 20 through 29 MUST be repeated for each network key Node A has granted. Key Exchange MUST follow the order described in* Section 4.2.6.6.1.

20. **B → A: Security 2 Network Key Get:**

Node B requests a specific Key from Node A.

21. **A4:**

CC:009F.01.00.11.0B2 Node A MUST cancel the S2 bootstrapping if the requested key is not in the Key List granted by Node A.

22. **A → B: Security 2 Network Key Report:**

This command returns the requested key.

23. **B4:**

CC:009F.01.00.11.094 Node B MUST cancel security bootstrapping if the received key was not requested. Refer to Section 4.2.6.6.5.

CC:009F.01.00.11.067 After receiving the key, Node B MUST perform key derivation as described in Section 4.2.6.4.13.

24. **Security 2 Network Key Established:**

Node A and Node B are now in possession of a shared network key.

25. **B → A: Nonce Get:**

Node B requests a Nonce from Node A that will allow Node B to send messages securely using the recently exchanged Network Key.

26. **A → B: Nonce Report:**

A's Nonce

27. **B → A: Security 2 Network Key Verify:**

CC:009F.01.00.11.095 This command MUST be sent encrypted by Node B with the newly received Network Key and the recently established SPAN for this key.

28. **A5:**

CC:009F.01.00.11.06A Node A MUST verify that it can successfully decrypt the Key Verify command using the newly exchanged key.

29. **A → B: Security 2 Transfer End:**

CC:009F.01.00.11.06B If Node A is able to decrypt and verify the Key Verify command, it MUST respond with Security 2 Transfer End with the field "Key verified" set to '1'.

   a. If there are more granted keys to request, Node B continues to step 20 and requests the next granted key

   b. If it was the last granted key, Node B continues to step 30.

All Keys have been requested.

30. **B → A: Security 2 Transfer End:**

CC:009F.01.00.11.06C When Node B has no more keys to request it MUST finish the secure setup by sending a Security 2 Transfer End command with the field "Key Request Complete" set to '1'.

   a. If this frame is received before all keys have been requested, the controller MUST consider the S2 Bootstrapping process failed.

CC:009F.01.00.11.06D All timeouts described in Section 4.2.6.6.6 MUST be implemented. If a node times out, it MUST silently abort the S2 bootstrapping.

4.2.6.6.5 Security 2 bootstrapping Interrupt Points

CC:009F.01.00.11.077 The including node and the joining node MUST interrupt the Security 2 bootstrapping process if any of the events outlined in Table 4.15 occur.

Table 4.15: Security 2 bootstrapping Interrupt Points

Including Node Interrupts, A1-5			
Name	**Interrupt Reason**	**KEX Fail Command Encryption**	**KEX Fail Type (see description in Table 20)**
A1	Key list is invalid or unsupported KEX Scheme is invalid or unsupported Curves are invalid or unsupported User rejects the requested keys	None	KEX_FAIL_KEX_KEY KEX_FAIL_KEX_SCHEME KEX_FAIL_KEX_CURVES KEX_FAIL_CANCEL
A2	User cancelled the bootstrapping	None	KEX_FAIL_CANCEL
A3	KEX Set(Echo) is not identical to the previous KEX Set	Temp. Key	KEX_FAIL_AUTH KEX_FAIL_DECRYPT
A4	The requested Key was not granted in the original KEX set	Temp. Key	KEX_FAIL_KEY_GET
A5	The Key Verify Command cannot be decrypted using the most recent key	Temp. Key	KEX_FAIL_KEY_VERIFY
Joining Node Interrupts, B1-4			
Name	**Interrupt Reason**	**KEX Fail Command Encryption**	**KEX Fail Type (see description in Table 20)**
B1	Key list is invalid or unsupported KEX Scheme is invalid or unsupported Curves are invalid or unsupported Node A wrongfully grants CSA	None	KEX_FAIL_KEX_KEY KEX_FAIL_KEX_SCHEME KEX_FAIL_KEX_CURVES KEX_FAIL_KEX_KEY
B2	If using CSA, user cancelled the bootstrapping	-	KEX_FAIL_CANCEL
B3	KEX Report(Echo) is not identical to the previous KEX Report	Temp. Key	KEX_FAIL_AUTH KEX_FAIL_DECRYPT
B4	The assigned key was never requested	Temp. Key	KEX_FAIL_KEY_REPORT

CC:009F.01.00.12.018 An aborted Security 2 bootstrapping process due to one of the interrupt points in Table 4.15 SHOULD be followed by a Security 2 KEX Fail Command sent to the other party of the S2 bootstrapping.

CC:009F.01.00.11.0A8 If sent, encryption MUST be used according to Table 4.15.

CC:009F.01.00.12.019 If a command is received using a Security level (non-secure, encrypted with temporary or network key) whereas the command had to be sent encrypted using another Security level, the receiving node SHOULD return a Security 2 KEX Fail Command with the KEX_FAIL_AUTH Fail Type encrypted using the received security level.

CC:009F.01.00.11.07C In any case, a node MUST NOT return any error indication if no S2 bootstrapping process is currently ongoing.

4.2.6.6.6 Security 2 bootstrapping Timeouts

CC:009F.01.00.11.07D The including Node or the joining node MUST apply timeouts according to Table 4.16.

Table 4.16: Security 2 bootstrapping Timeouts

Including Node Timers, TA1-7		
Name	**Interrupt Reason**	**Timeout (Seconds)**
CC:009F.01.00.11.07E TA1	KEX Report MUST be received after sending KEX Get	10
CC:009F.01.00.11.07F TA2	Public Key Report MUST be received after sending KEX Set	10
CC:009F.01.00.11.082 TA3	S2 Network Key Get MUST be received after sending KEX Report(Echo)	10
CC:009F.01.00.11.083 TA4	S2 Network Key Verify MUST be received after sending S2 Network Key Report	10
CC:009F.01.00.11.084 TA5	S2 Transfer End OR S2 Network Key Get MUST be received after sending S2 Transfer End	10
CC:009F.01.00.13.012 TAI1	User MAY change Key List for Advanced Joining mode	240
CC:009F.01.00.11.085 TAI2	User MUST have verified/input the DSK	240
Joining Node Timers, TB1-7		
Name	**Interrupt Reason**	**Timeout (Seconds)**
CC:009F.01.00.11.0B3 TB1	KEX Get MUST be received after non-secure inclusion	10..30 for nodes supporting S0 30 for nodes supporting S2 only
CC:009F.01.00.11.087 TB2	KEX Set MUST be received after sending KEX Report	240
CC:009F.01.00.11.088 TB3	Public Key Report MUST be received after sending Public Key Report	10
CC:009F.01.00.11.08B TB4	S2 Network Key Report MUST be received after sending S2 Network Key Get	10
CC:009F.01.00.11.08C TB5	S2 Transfer End MUST be received after sending S2 Network Key Verify	10
CC:009F.01.00.11.0AE TBI1	If Client-Side Authentication is used, the user MUST have verified/input the DSK	240

CC:009F.01.00.11.0AF 10 seconds timeouts MUST be observed with a 2 seconds tolerance. In this case, a node MUST time out between 8 and 12 seconds.

CC:009F.01.00.11.0B0 30 seconds timeouts MUST be observed with a 5 seconds tolerance. In this case, a node MUST time out between 25 and 35 seconds.

CC:009F.01.00.11.0B1 240 seconds timeouts MUST be observed with a 10 seconds tolerance. In this case, a node MUST time out between 230 and 250 seconds.

4.2.6.7 Security 2 Key Exchange commands

4.2.6.7.1 Security 2 KEX Get Command

This command is used by an including node to query the joining node for supported KEX Schemes and ECDH profiles as well as which network keys the joining node intends to request.

CC:009F.01.04.11.001 This command MUST be ignored if Learn Mode is disabled.

CC:009F.01.04.11.002 The KEX Report Command MUST be returned in response to this command if Learn Mode is enabled.

CC:009F.01.04.11.003 This command MUST NOT be issued via multicast addressing.

CC:009F.01.04.11.004 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = KEX_GET (0x04)							

4.2.6.7.2 Security 2 KEX Report Command

This command is used for two purposes during the key exchange:

1. This command is used by a joining node to advertise the network keys which it intends to request from the including node. The including node subsequently grants keys which may be exchanged once a temporary secure channel has been established.
2. After establishment of the temporary secure channel, the including node uses this command to confirm the set of keys that the joining node intends to request.

CC:009F.01.05.11.003 A receiving node MUST ignore this command if Add Node mode is not enabled.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = KEX_REPORT (0x05)							
Reserved						Request CSA	Echo
Supported KEX Schemes							
Supported ECDH Profiles							
Requested Keys							

Reserved

CC:009F.01.05.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Echo (1 bit)

CC:009F.01.05.11.005 If this flag is set to '1', the fields of this command MUST be a copy of the KEX Report received prior to the establishment of the temporary secure channel. All fields described in this section MUST be included in the echo. For the purpose of echoing only, reserved set bits and the reserved field MUST NOT be ignored or set to zero, but MUST be echoed back as received. If future versions of this Command Class append fields to the KEX Report, those fields MUST be omitted from the echo.

CC:009F.01.05.11.006 If this flag is set to '1', the command MUST be sent securely via the temporary secure channel, i.e. encrypted using the `TempKeyCCM` and `TempPersonalizationString`.

CC:009F.01.05.11.007 The including node MUST NOT set this flag to '0' when sending this command and MUST ignore this command if this flag is set to '1' when received.

CC:009F.01.05.11.008 The including node MUST return a KEX Set Command in response to this command if the "Echo" flag is set to '0' and is performing S2 Bootstrapping.

CC:009F.01.05.11.009

The joining node MUST NOT set this flag to '1' when sending this command and MUST ignore this command if this flag is set to '0' when received.

Request CSA (1 bit)

If this flag is set, the joining node is requesting the use of Client-side Authentication (CSA).

CC:009F.01.05.11.019 This flag MUST be set to 0 if none of the S2 Authenticated and S2 Access Control Security Classes are requested

CC:009F.01.05.11.01A This flag MUST be set to 0 if the sending node has a DSK label printed on itself.

CC:009F.01.05.11.01B This flag MUST be set to 1 by a node only if it has been OTA firmware upgraded to support S2 and requests S2 Authenticated or S2 Access Control Security Class.

Supported KEX Schemes (1 byte)

This field is used to advertise the KEX Schemes supported by the node.

The field MUST be treated as a bitmask and MUST comply with the format indicated in Table 4.17:

Table 4.17: Supported KEX Schemes

Bit	KEX Scheme	Description
0, 2..7	Reserved	Reserved
1	KEX Scheme 1	Indicates if Scheme 1 is supported (Security 2 Class Keys 0..2)

All other bits are reserved and MUST be set to zero by a sending node.

CC:009F.01.05.11.00E If the KEX scheme is supported the corresponding bit MUST be set to '1'.

If the KEX scheme is not supported the corresponding bit MUST be set to '0'.

CC:009F.01.05.11.00F A node supported the Security 2 Command Class MUST support KEX Scheme 1.

Supported ECDH Profiles (1 byte)

This field is used to advertise the supported ECDH Profiles by the joining node

CC:009F.01.05.11.010 The field MUST be treated as a bitmask and MUST comply with the format indicated in Table 4.18:

Table 4.18: Supported ECDH Profiles

Bit	ECDH Profile	KEX Scheme	Public Key Length
0	Curve25519	Indicates support for Curve25519 [1]	32 Bytes

All other bits are reserved and MUST be set to zero by a sending node.

CC:009F.01.05.11.013 If the ECDH Profile is supported the corresponding bit MUST be set to '1'

If the ECDH Profile is not supported the corresponding bit MUST be set to '0'

CC:009F.01.05.11.014 Curve25519 MUST be supported by a node supporting the Security 2 Command Class.

Requested Keys (1 byte)

This field is used by a joining node to advertise the keys that the manufacturer finds most appropriate for the actual type of product.

CC:009F.01.05.13.001 The joining node MAY request a subset of the keys defined in Table 4.19.

CC:009F.01.05.11.015 The field MUST be treated as a bitmask and MUST comply with the format indicated in Table 4.19:

Table 4.19: Requested Keys

Security Level (1 highest - 4 lowest)	Bit	KEX Scheme	Key	Indicates support for
1	2	KEX Scheme 1	S2.2	S2 Access Control Class
2	1	KEX Scheme 1	S2.1	S2 Authenticated Class
3	0	KEX Scheme 1	S2.0	S2 Unauthenticated Class
4	7	KEX Scheme 1	S0	S0 Secure legacy devices

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

CC:009F.01.05.11.017 If the Key is requested the corresponding bit MUST be set to '1'.

If the Key is not requested the corresponding bit MUST be set to '0'.

CC:009F.01.05.11.018 A node supporting the Security 2 Command Class MUST support at least one Key. A node may support multiple keys.

4.2.6.7.3 Security 2 KEX Set Command

This command is used for two purposes:

1. During initial key exchange this command is used by an including node to grant network keys to a joining node. The joining node subsequently requests the granted keys once a temporary secure channel has been established.

   CC:009F.01.06.11.001 The including node MUST send the command non-securely.

2. After establishment of the temporary secure channel, the joining node issues this command to the including node to securely state its intention to request the keys that were granted previously.

   CC:009F.01.06.11.002 The joining node MUST send the command securely via the temporary secure channel, i.e. encapsulated using the `TempKeyCCM` and `TempPersonalizationString`.

CC:009F.01.06.11.003 This command MUST be ignored if Learn mode and Add Node mode are both disabled.

CC:009F.01.06.11.004 The including node MUST return the Security 2 KEX Report Command in response to this command unless it is to be ignored.

CC:009F.01.06.11.005 This command MUST NOT be issued via multicast addressing.

CC:009F.01.06.11.006 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = KEX_SET (0x06)							
Reserved						Request CSA	Echo
Supported KEX Schemes							
Supported ECDH Profile							
Granted Keys							

Reserved

CC:009F.01.06.11.007 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Echo (1 bit)

CC:009F.01.06.11.008 If this flag is set to '1', the fields of this command MUST be an identical copy of the KEX Set Command received prior to the establishment of the temporary secure channel.

CC:009F.01.06.11.009

The joining node MUST set this flag to '1'.

CC:009F.01.06.11.00A The including node MUST abort the S2 bootstrapping if it receives a KEX Set Command with this flag set to '0'.

CC:009F.01.06.11.00B If the "Echo" flag is set to '1' and the Add Node mode is enabled, the including node MUST return a KEX Report Command with the "Echo" flat set to '1' in response to this command.

CC:009F.01.06.11.00D The including node MUST set this flag to '0'.

CC:009F.01.06.11.00E The joining node MUST abort the S2 bootstrapping if it receives a KEX Set Command with this flag set to '1'.

Request CSA (1 bit)

If this flag is set to '1', the including node is permitting the use of Client-side Authentication (CSA).

Selected KEX Scheme (1 byte)

CC:009F.01.06.11.010 This field is used to specify the Scheme that the including node is granting to the joining node. Exactly one bit MUST be set to '1'. Reserved bits MUST be examined for the purpose of verifying that exactly one bit is set. Several bits set MUST trigger an error as indicated in Section 4.2.6.6.5.

For field format, refer to Table 4.17.

Selected ECDH Profile (1 byte)

CC:009F.01.06.11.011 This field specifies the ECDH Profile selected by the including node for ECDH Key Exchange. The field MUST carry exactly one of the bits listed in Table 18. Reserved bits MUST be examined for the purpose of verifying that exactly one bit is set.

For field format, refer to Table 4.18.

Granted Keys (1 byte)

CC:009F.01.06.11.012 If the Echo field is set to '0', this field MUST specify the keys which the including node is granting to the joining node.

CC:009F.01.06.11.013 The value of this field MUST be the same set or a subset of the Requested Keys field advertised in the KEX Report by the joining node during initial S2 bootstrapping.

CC:009F.01.06.11.014 If the Echo field is set to '1', this field MUST advertise the keys which the including node granted to the joining node in the previous KEX Set Command sent by the including node to the joining node.

For field format, refer to Table 4.19.

CC:009F.01.06.11.015 A joining node MUST NOT issue Network Key Get requests for keys that are not specified in this field.

CC:009F.01.06.11.016 An including node MUST return a Security 2 KEX Fail Command and abort S2 bootstrapping if it subsequently receives a Network Key Get request for keys that are not specified in this field.

4.2.6.7.4 Security 2 KEX Fail Command

This command is used to advertise an error condition to the other party of am S2 bootstrapping process.

The interrupt points that may trigger the transmission of this command are defined in Section 4.2.6.2.6.

CC:009F.01.07.11.001 This command MUST be ignored if Learn mode and Add Node mode are both disabled.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = KEX_FAIL (0x07)							
KEX Fail Type							

KEX Fail Type (1 byte)

This field MUST advertise one of the types defined in Table 4.20.

Table 4.20: Security 2 KEX Fail::KEX Fail Type

Value	KEX Fail Type Identifier	Description
0x01	KEX_FAIL_KEX_KEY	Key failure indicating that no match exists between requested/granted keys in the network.
0x02	KEX_FAIL_KEX_SCHEME	Scheme failure indicating that no scheme is supported by controller or joining node specified an invalid scheme.
0x03	KEX_FAIL_KEX_CURVES	Curve failure indicating that no curve is supported by controller or joining node specified an invalid curve.
0x05	KEX_FAIL_DECRYPT	Node failed to decrypt received frame.
0x06	KEX_FAIL_CANCEL	User has cancelled the S2 bootstrapping.
0x07	KEX_FAIL_AUTH	The Echo KEX Set/Report frame did not match the earlier exchanged frame. A command is received using the wrong Security level.
0x08	KEX_FAIL_KEY_GET	The joining node has requested a key, which was not granted by the including node at an earlier stage.
0x09	KEX_FAIL_KEY_VERIFY	Including node failed to decrypt and hence verify the received frame encrypted with exchanged key.
0x0A	KEX_FAIL_KEY_REPORT	The including node has transmitted a frame containing a different key than what is currently being exchanged.

4.2.6.7.5 Security 2 Public Key Report Command

This command is used by both the including and the joining node to establish the Elliptic Curve Shared Secret. This is needed to establish the temporary secure channel that enables transfer of all other keys.

CC:009F.01.08.11.001 This command MUST be ignored if Learn mode and Add Node mode are both disabled.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = PUBLIC_KEY_REPORT (0x08)							
Reserved							Including Node
ECDH Public Key 1 (MSB)							
...							
ECDH Public Key N (LSB)							

Reserved

CC:009F.01.08.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Including Node (1 bit)

The Including Node flag advertises if the sending node is the including node or the joining node.

CC:009F.01.08.11.003 When sent by the including node this flag MUST be set to '1'.

CC:009F.01.08.11.004 The joining node MUST abort S2 bootstrapping if this flag is set to '0' in a received command.

CC:009F.01.08.11.005 When sent by the joining node this flag MUST be set to '0'.

CC:009F.01.08.11.006 The including node MUST abort S2 bootstrapping if this flag is set to '1' in a received command.

ECDH Public Key (N bytes)

Device Specific ECDH Public Key of the sending node.

CC:009F.01.08.11.007 The public key MUST be unique for each node. The length of this field is determined by the chosen ECDH profile. Refer to Table 4.18.

CC:009F.01.08.11.008 If Controller-Side Authentication is used, the DSK bytes 1..2 MUST be obfuscated with zeros (0x00) by the joining node

If Client-Side Authentication (CSA) is used, the DSK bytes 1..4 MUST be obfuscated with zeros (0x00) by the including node.

CC:009F.01.08.11.00B If no authentication is used (S2 Unauthenticated and/or S0 classes are granted only), both joining and including nodes ECDH Public Keys MUST be transmitted in their integrality

4.2.6.7.6 Security 2 Network Key Get Command

CC:009F.01.09.11.001 This command is used by a joining node to request one key from the including node. One instance of this command MUST be sent for each key that was granted by the including node.

CC:009F.01.09.11.002 The command MUST be sent security encapsulated using the `TempKeyCCM` and `TempPersonalizationString`.

CC:009F.01.09.11.003 This command MUST be ignored unless the Add Node mode is enabled.

CC:009F.01.09.11.004 The Security 2 Network Key Report Command MUST be returned in response to this command unless this command is ignored.

CC:009F.01.09.11.005 This command MUST NOT be issued via multicast addressing.

CC:009F.01.09.11.006 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = SECURITY_2_NETWORK_KEY_GET (0x09)							
Requested Key							

Requested Key (1 byte)

This field is used to request a network key.

CC:009F.01.09.11.007 Only one key MUST be requested at a time, i.e. only 1 bit MUST be set to '1'. This field MUST be encoded according to Table 4.19.

4.2.6.7.7 Security 2 Network Key Report Command

This command is used by an including node to transfer one key to the joining node.

CC:009F.01.0A.11.001 The command MUST be sent security encapsulated using the `TempKeyCCM` and `TempPersonalizationString`.

CC:009F.01.0A.11.002 This command MUST be ignored unless the Learn mode is enabled.

CC:009F.01.0A.11.003 The joining node MUST store the received network key in non-volatile memory.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = SECURITY_2_NETWORK_KEY_REPORT (0x0A)							
Granted Key							
Network Key byte 1							
...							
Network Key byte 16							

Granted Key (1 byte)

CC:009F.01.0A.11.004 This field is used to indicate which Network Key is carried in the command and MUST be encoded according to Table 4.19.

Network Key (16 bytes)

This field carries the granted Network Key.

4.2.6.7.8 Security 2 Network Key Verify Command

This command is used by a joining node to verify a newly exchanged key with the including node.

CC:009F.01.0B.11.007 This command MUST be sent security encapsulated using the Key that was just exchanged and MUST be encrypted using the derived `KeyCCM` and `PersonalizationString`.

CC:009F.01.0B.11.004 The Security 2 Transfer End Command MUST be returned in response to this command unless it is to be ignored.

CC:009F.01.0B.11.005 The joining node MUST NOT consider the actual key exchange to be complete until a Security 2 Transfer End command has been received from the including node.

CC:009F.01.0B.11.006 This command MUST be ignored if Learn mode and Add Node mode are both disabled.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = SECURITY_2_NETWORK_KEY_VERIFY (0x0B)							

4.2.6.7.9 Security 2 Transfer End Command

This command is used by the including node to complete the verification of each individual key exchange while the joining node uses this command to complete the S2 bootstrapping process after all granted keys have been successfully exchanged.

CC:009F.01.0C.11.001 The joining node MUST send this command after all granted keys have been verified.

CC:009F.01.0C.11.002 This command MUST be ignored if Learn mode and Add Node mode are both disabled.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = SECURITY_2_TRANSFER_END (0x0C)							
Reserved						Key Verified	Key Request Complete

Reserved

CC:009F.01.0C.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Key Verified (1 bit)

CC:009F.01.0C.11.004 The including node MUST set this flag to '1' if it has successfully verified the Key Verify Command using the newly exchanged network key.

CC:009F.01.0C.11.005

This flag MUST be set to '0' in all other cases.

CC:009F.01.0C.11.009 If this field is set to 0, a receiving node MUST abort S2 bootstrapping and consider that S2 bootstrapping process failed.

Key Request Complete (1 bit)

CC:009F.01.0C.11.006 The joining node MUST set this flag to '1' if it has completed exchanging all granted keys.

CC:009F.01.0C.11.007 This flag MUST be set to '0' in all other cases.

CC:009F.01.0C.11.008 The including node MUST consider S2 bootstrapping to be successfully completed if it receives this command with this flag set to '1' and all keys have been exchanged.

4.2.6.8 Discovery of Security Capabilities Commands

A controlling node may discover the Command Class capabilities of a securely included node.

CC:009F.01.00.13.015 The advertised capabilities MAY depend on the actual security level used to request capabilities.

Likewise, the advertised capabilities at a given security level may depend on the S2 bootstrapping security level.

4.2.6.8.1 Security 2 Commands Supported Get Command

This command is used to query the command classes that a joining node supports via secure communication.

CC:009F.01.0D.11.001 Security 2 encryption MUST be used when transmitting this command.

CC:009F.01.0D.11.002 The Security 2 Commands Supported Report Command MUST be returned in response to this command. The response MUST be sent encrypted, using the same Security 2 Class key and encapsulation as was used for this command.

CC:009F.01.0D.11.003 A node receiving this command on its highest granted Security Class MUST respond with the Security 2 Commands Supported Report Command containing all supported secure command classes.

CC:009F.01.0D.11.004 A node receiving this command on any other Security Class than its highest granted Security Class MUST respond with the Security 2 Commands Supported Report Command containing no command classes.

CC:009F.01.0D.11.007 A node receiving the Security Commands Supported Get of the (non S2) Security 0 Command Class (using S0 key and S0 Message Encapsulation Command), MUST respond with the Security 0 Commands Supported Report containing no command classes.

CC:009F.01.0D.11.006 This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = SECURITY_2_COMMANDS_SUPPORTED_GET (0x0D)							

4.2.6.8.2 Security 2 Commands Supported Report Command

This command is used to advertise the commands that a joining node supports via secure communi-
CC:009F.01.0E.11.001 cation. Security 2 encryption MUST be used when transmitting this command.

CC:009F.01.0E.11.002 Transport Service segmentation MUST be used if the command is longer than the available payload length of a single Z-Wave MAC frame.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = SECURITY_2_COMMANDS_SUPPORTED_REPORT (0x0E)							
Command Class 1							
...							
Command Class N							

CC:009F.01.0E.11.003 This command MUST NOT advertise command classes that the joining node can control in other
CC:009F.01.0E.12.001 nodes. Network management user interfaces SHOULD discover control capabilities of a node via the Association Group Information (AGI) Command Class.

As per Section 4.2.6.6, a node may be assigned up to four keys for the S2 Unauthenticated, S2 Authenticated, S2 Access Control and S0 Classes, respectively.

For instance, a light dimmer may accept non-securely transmitted commands if it is not S2 bootstrapped, while the same dimmer will require secure communication for commands if it is S2 bootstrapped. This allows a Security 2 enabled light dimmer to be compatible with first-generation non-secure Z-Wave networks while it will operate fully secure when included in a S2 network at a later time.

In another example, a door lock may be designed to work only with the S0 network key or the S2 Access Control Class key. Inclusion in a network without security bootstrapping will not allow operation of the lock and if bootstrapped with the S2 Access Control Class, S0 encrypted commands will also be ignored.

CC:009F.01.0E.11.00F The Security 0 and Security 2 Command Class MUST NOT be advertised in this command

The Transport Service Command Class MUST NOT be advertised in this command.

CC:009F.01.0E.13.002 A sending node MAY terminate the list of supported command classes with the COMMAND_CLASS_MARK command class identifier.

CC:009F.01.0E.11.007 A receiving node MUST stop parsing the list of supported command classes if it detects the COMMAND_CLASS_MARK command class identifier in the Security 2 Commands Supported Report.

Command Class (N bytes)

This field advertises the command classes that the node supports.

CC:009F.01.0E.11.008 A normal Command Class identifier MUST be one byte long in the range (0x20-0xEE). An extended Command Class identifier MUST be two bytes long where the first byte is in the range (0xF1-0xFF), while the second byte is in the range 0x00-0xFF.

CC:009F.01.0E.13.003 A joining node MAY advertise extended command classes.

CC:009F.01.0E.11.009 An including node MUST accept extended command classes.

4.2.7 Security 2 (S2) Command Class, version 2

The Security 2 Command Class is a framework for allowing nodes to communicate securely in a Z-Wave network.

CC:009F.02.00.11.001 Security 2 (S2) Command Class, version 2 is backwards compatible with the *Security 2 (S2) Command Class, version 1*. Fields and commands not described in this version MUST remain unchanged from version 1.

4.2.7.1 Network Layer Security (NLS) Introduction

Network Layer Security or NLS is a feature embedded within the Security 2 Command Class and provides a secure alternative for performing certain lower level mesh network protocol operations previously not addressed by S2 Security.

Some example operations that NLS covers:

- Network exclusion
- Remove Failed Node
- Replace Failed Node
- Request Node Neighbors

CC:009F.02.00.11.002 A node supporting Security 2 Command Class version 2 MUST support NLS.

CC:009F.02.00.13.001 NLS MAY be enabled at a later point in time from the Security 2 bootstrapping process. This allows existing nodes on the network with *Security 2 (S2) Command Class, version 1* to be updated via OTA and enable NLS without being re-included into the network.

CC:009F.02.00.11.009 NLS MUST NOT be disabled once enabled for a network.

CC:009F.02.00.13.002 NLS is supported on a node-by-node basis and networks MAY operate with a mixture of NLS enabled and disabled. Legacy nodes MAY continue to operate without NLS support.

CC:009F.02.00.11.003 If NLS is enabled for a node on the network that includes a SIS, the SIS MUST use the secure NLS commands to perform low level network protocol operations.

CC:009F.02.00.11.004 If NLS is enabled for a node on the network, the node MUST allow for a physical way (button, touchscreen, power toggle sequence) to reset the node locally and wipe its network settings. This prevents orphaned nodes without operating controllers from being stuck in their old network.

CC:009F.02.00.11.005 An NLS-enabled node MUST accept protocol commands [in accordance with the list below] only when communication is using its highest Security Class granted during security bootstrapping.

CC:009F.02.00.11.006 A SIS or Primary Controller supporting NLS MUST accept protocol commands [in accordance with the list below]

- from an NLS-enabled node: only when communication is using the highest common Security Class shared between the SIS or Primary Controller and the sending node (i. e. its highest granted Security Class),
- otherwise: no matter which security level.

CC:009F.02.00.11.007 The NLS state MUST be disabled during exclusion and reset.

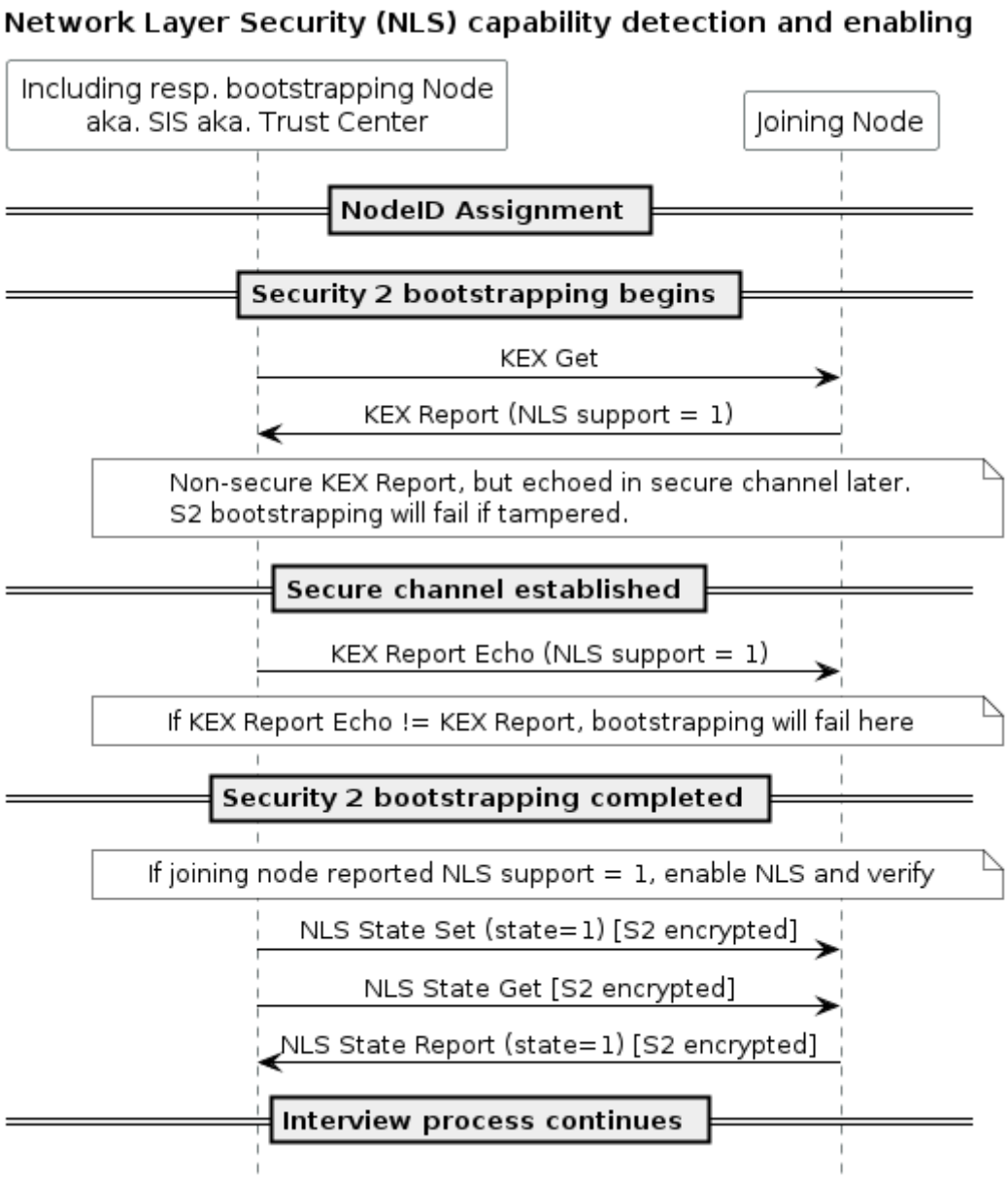


Figure 4.20: Inclusion with NLS

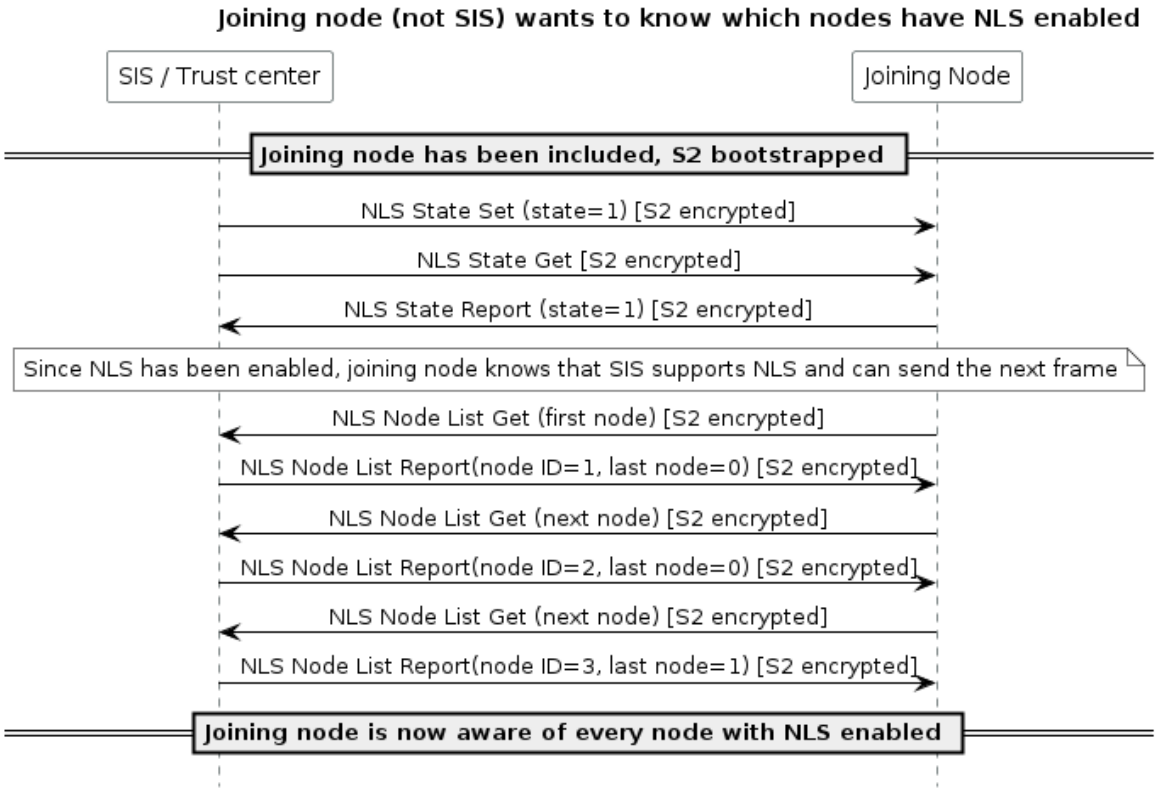


Figure 4.21: Joining node asks for NLS enabled nodes

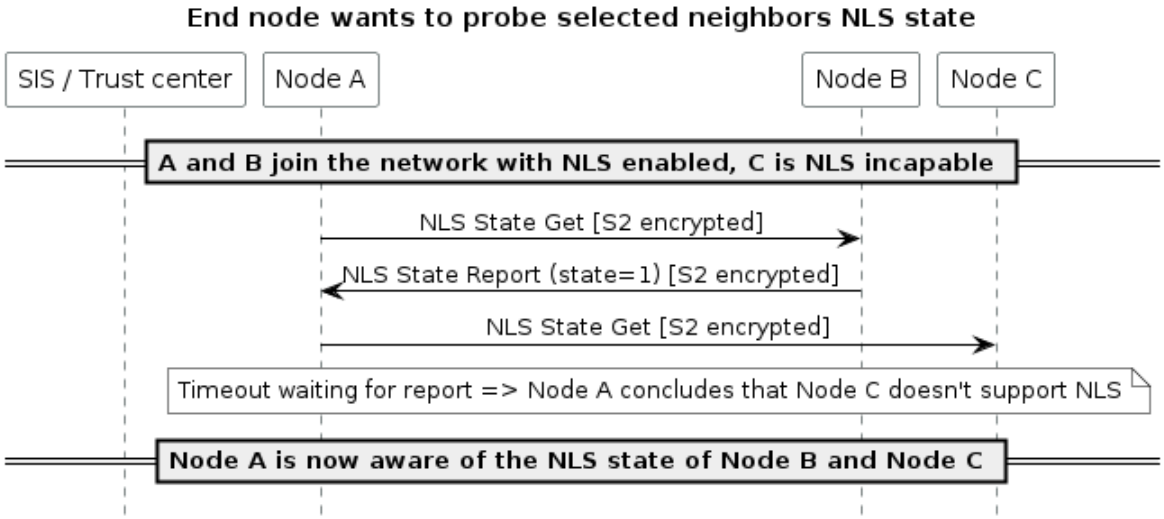


Figure 4.22: Node A probes other nodes for NLS state

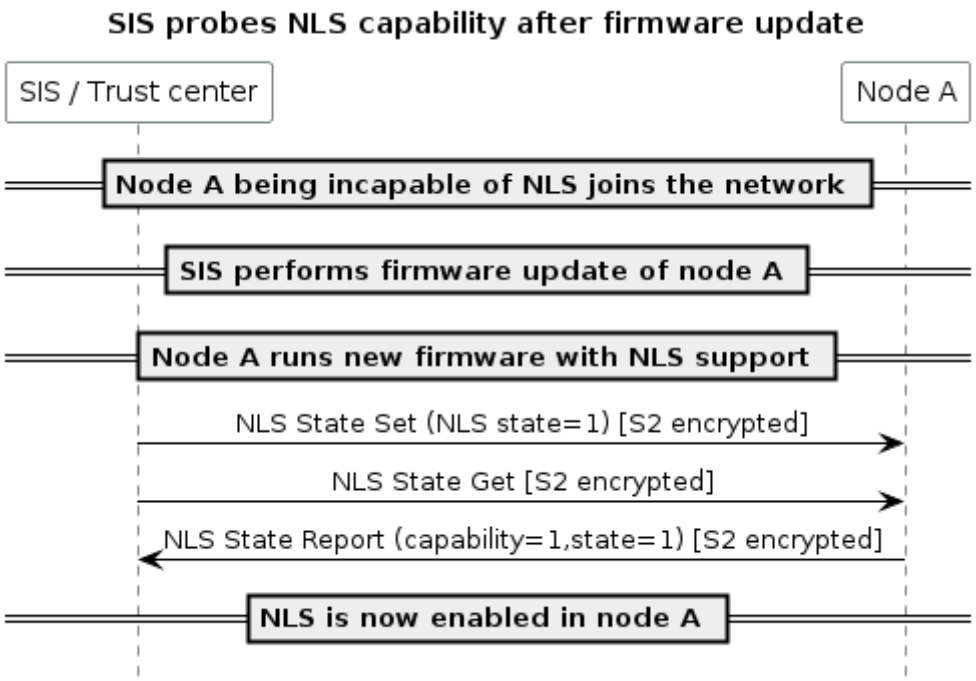


Figure 4.23: SIS probes NLS capability after firmware update

4.2.7.2 Compatibility considerations

CC:009F.02.00.11.008 Compatibility consideration requirements from version 1 MUST also be observed by a version 2 supporting node.

The following command has been extended to report whether a node supports NLS or not:

- *KEX Report Command*

The following commands are added for managing NLS:

- *NLS Node List Get Command*
- *NLS Node List Report Command*
- *NLS State Get Command*
- *NLS State Report Command*
- *NLS State Set Command*

4.2.7.3 Security considerations

CC:009F.02.00.12.001 NLS increases the security on the network layer and it is therefore RECOMMENDED to enable this for as many devices as possible taking the interoperability considerations below into account.

4.2.7.4 Interoperability considerations

A Z-Wave network consisting of both NLS enabled devices and non-NLS devices is considered a mixed network and will have certain limitations.

Example: a primary controller supporting NLS will be able to communicate with both NLS and non-NLS devices, but an inclusion controller that does not support NLS will be able to exchange network layer frames only with non-NLS devices and devices where NLS is not enabled yet.

4.2.7.5 Commands covered by NLS

4.2.7.5.1 Z-Wave network layer commands always covered by NLS

Z-Wave Protocol Command Class:

- 0x0C Assign Return Route Command
- 0x0D New Node Registered Command
- 0x0E New Range Registered Command
- 0x10 Automatic Controller Update Start Command
- 0x12 Set SUC Command
- 0x13 Set SUC ACK Command
- 0x15 Static Route Request Command
- 0x18 NOP Power Command
- 0x19 Reserve Node IDs Command
- 0x1A Reserved IDs Command
- 0x1F Nodes Exist Command
- 0x20 Nodes Exist Reply Command
- 0x22 Set NWI Mode Command
- 0x23 Exclude Request Command
- 0x24 Assign Return Route Priority Command
- 0x25 Assign SUC Return Route Priority Command

4.2.7.5.2 Z-Wave network layer commands covered by NLS after secure bootstrapping

Z-Wave Protocol Command Class:

- 0x04 Find Nodes in Range Command
- 0x05 Get Nodes in Range Command
- 0x06 Range Info Command
- 0x07 Command Complete Command
- 0x09 Transfer Node Information Command
- 0x0A Transfer Range Information Command
- 0x0B Transfer End Command
- 0x0F Transfer New Primary Controller Complete Command
- 0x11 SUC Node ID Command
- 0x14 Assign SUC Return Route Command

4.2.7.5.3 Z-Wave network layer commands never covered by NLS

No Operation Command Class:

- No payload

Z-Wave Protocol Command Class:

- 0x01 Node Information Frame Command
- 0x02 Request Node Information Frame Command
- 0x03 Assign IDs Command
- 0x08 Transfer Presentation Command
- 0x16 Lost Command
- 0x17 Accept Lost Command
- 0x26 SmartStart Included Node Information Command
- 0x27 SmartStart Prime Command
- 0x28 SmartStart Inclusion Request Command

4.2.7.5.4 Z-Wave LR network layer commands never covered by NLS

Z-Wave Long Range Command Class:

- 0x00 No Operation Command
- 0x01 Node Information Frame Command
- 0x02 Request Node Information Frame Command
- 0x03 Assign IDs Command
- 0x23 Exclude Request Command
- 0x26 SmartStart Included Node Information Command
- 0x27 SmartStart Prime Command
- 0x28 SmartStart Inclusion Request Command
- 0x29 Exclude Request Confirmation Command
- 0x2A Non Secure Inclusion Step Complete Command

4.2.7.6 KEX Report Command

This command is used for two purposes during the key exchange:

1. This command is used by a joining node to advertise the network keys which it intends to request from the including node. The including node subsequently grants keys which may be exchanged once a temporary secure channel has been established.
2. After establishment of the temporary secure channel, the including node uses this command to confirm the set of keys that the joining node intends to request.

CC:009F.02.05.11.001 A receiving node MUST ignore this command if Add Node mode is not enabled. Refer to [35] for details.

Table 4.21: Kex Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = KEX_REPORT (0x05)							
Reserved					NLS support	Request CSA	Echo
Supported KEX schemes							
Supported ECDH schemes							
Requested keys							

NLS support (1 bit)

This field is used to advertise that the sending node supports Network Layer Security (NLS).

CC:009F.02.05.11.002 The value 1 MUST indicate that the sending node supports NLS.

The value 0 MUST indicate that the sending node does not support NLS.

4.2.7.7 NLS Node List Get Command

This command is used to request a list of nodes that have NLS enabled and their granted keys from the SIS.

Nodes in the network will know that the SIS supports this command if they got NLS enabled during S2 bootstrapping.

CC:009F.02.0F.11.001 This command MUST be sent securely.

Table 4.22: NLS Node List Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = NLS_NODE_LIST_GET (0x0F)							
Request							

Request (8 bit)

This field is used to request the first node or the next node in the list.

CC:009F.02.0F.11.002 The value 0 MUST indicate that the sending node is requesting the first node in the list.

The value 1 MUST indicate that the sending node is requesting the next node in the list.

All other values are reserved.

4.2.7.8 NLS Node List Report Command

This command is used to advertise the list of nodes that have NLS enabled and their granted keys.

CC:009F.02.10.11.001 This command MUST be sent securely.

Table 4.23: NLS Node List Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = NLS_NODE_LIST_REPORT (0x10)							
Reserved							Last node
ID (MSB) of node N							
ID (LSB) of node N							
Granted keys bitmask of node N							
NLS state of node N							

Last node (1 bit)

This field indicates whether the node contained in the report is the last node in the list.

CC:009F.02.10.11.002 The value 0 MUST indicate that the received node is not the last node in the list.

The value 1 MUST indicate that the received node is the last node in the list.

All other values are reserved.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

ID of node (16 bits)

This field is used to indicate the NodeID of the node being advertised.

Granted keys bitmask of node (8 bits)

This field is used to advertise a bitmask indicating the granted keys for the current NodeID being reported.

NLS state of node (8 bits)

This field is used to advetise the NLS state for the current NodeID being reported.

4.2.7.9 NLS State Get Command

This command is used by a controller to query a node for the state of NLS.

A command that can be sent to any node in the network and will return the state of NLS in that node (Enabled/Disabled)

CC:009F.02.11.11.001 This command MUST be sent securely.

Table 4.24: NLS State Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = NLS_STATE_GET (0x11)							

4.2.7.10 NLS State Report Command

This command reports whether NLS is enabled or disabled for the sending node.

CC:009F.02.12.11.001 This command MUST be sent securely.

Table 4.25: NLS State Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = NLS_STATE_REPORT (0x12)							
Reserved						NLS state	Capability

Capability (1 bit)

This field advertises whether a sending node supports NLS or not.

It enables an including node to securely verify support for NLS.

NLS state (1 bit)

This field is used to advertise whether NLS is enabled or disabled.

CC:009F.02.12.11.002 The value 1 MUST indicate that NLS is enabled in the sending node.

The value 0 MUST indicate that NLS is disabled in the sending node.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.2.7.11 NLS State Set Command

This command sets the NLS state in the destination node, if supported.

CC:009F.02.13.11.001 This command MUST be sent securely.

Table 4.26: NLS State Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SECURITY_2 (0x9F)							
Command = NLS_STATE_SET (0x13)							
NLS state							

NLS state (8 bits)

This field is used to set whether NLS must be enabled.

CC:009F.02.13.11.002 The value 1 MUST indicate that NLS is enabled in the sending node.

All other values are reserved.

4.2.8 Supervision Command Class, version 1

The Supervision Command Class allows a sending node to request application-level delivery confirmation from a receiving node. The delivery confirmation includes relevant application-level status information in the confirmation message.

4.2.8.1 Terminology

The controlling application **initiates** an operation by sending a **Supervision encapsulated command** to a supporting node. The supporting node returns an **immediate confirmation** for the reception while advertising its ability to perform the requested operation. The confirmation may advertise the **application status** <Working>, <Success>, <No Support> or <Fail> depending on the condition of the supporting node. One or more application status updates may follow later to report problems, delays or the completion of the requested operation.

A door lock application is used as example in the following. Other uses may apply.

A magnetic lock may return the <Success> indication immediately since it takes only a few milliseconds to apply power to the electromagnet. In that case, no status updates are needed later.

An electro-mechanical lock may need time to complete its movement. A <Working> indication is issued along with the expected duration, e.g. 5 seconds. Ideally, the movement is completed and a <Success> indication is returned 5 seconds later. A motor defect or something jamming the movement may prevent the lock from completing the requested operation. A <Fail> indication is returned in that case. If the motor runs slower than anticipated, another <Working> indication may be issued when e.g. 75% of the expected duration has elapsed; reporting the new expected duration. If it runs faster, a <Success> indication may be issued before it was expected. Both cases allow a GUI progress bar to respond instantly rather than waiting for a polling response.

4.2.8.2 Compatibility Considerations

This command class is used as an integrated part of the Security 2 Command Class but may also be used for non-secure applications.

CC:006C.01.00.23.002 The Supervision Command Class MAY be used for solitary commands such as Set, Remove and unsolicited Report commands.

CC:006C.01.00.21.003 The Supervision Command Class MUST NOT be used for immediate session-like command flows such as Get↔Report command exchanges where a REPORT is expected immediately after the GET. Since the receiver is waiting for the report, it can resend the GET if it does not receive one. Unsolicited REPORT (or other similar) commands from an end device SHOULD be encapsulated with Supervision Command Class to ensure the controller is aware of a change of state.

CC:006C.01.00.23.001 The Supervision Get Command MAY carry multiple commands grouped with the Multi Command encapsulation command.

4.2.8.2.1 Node Information Frame (NIF)

CC:006C.01.00.21.001 A supporting node MUST always advertise the Supervision Command Class in its NIF and Multi Channel Capability Report, regardless of the inclusion status and security bootstrapping outcome.

4.2.8.2.2 Encapsulated Commands

Supervision Command Class is intended for encapsulating solitary commands; it is to say commands that do not require any response to be returned.

It exists Set type Commands that may optionally require to return a status or optional command such as Notification Report or handshake mechanism. When receiving such a Set type command in a Supervision Get Command:

CC:006C.01.00.21.002
- The Supervision Report MUST be returned by a receiving node.

CC:006C.01.00.23.003
CC:006C.01.00.22.001
- Additional commands MAY be returned. These commands SHOULD be returned before the Supervision Report Command

CC:006C.01.00.51.001 A sending node MUST NOT use handshake mechanisms for Set Commands when using Supervision encapsulation.

CC:006C.01.00.23.004 It is OPTIONAL for a receiving node to return an answer to a Set type Command which requires a response to be returned if received with Supervision encapsulation.

4.2.8.3 Supervision Get Command

This command is used to initiate the execution of a command and to request the immediate and future status of the process being initiated.

CC:006C.01.01.11.001 The Supervision Report Command MUST be returned in response to this command unless it is to be ignored.

CC:006C.01.01.11.002 The <SUCCESS> status MUST be advertised in the Supervision Report Command if the operation is completed immediately.

CC:006C.01.01.11.003 If the requested operation is accepted but cannot be completed immediately, the <WORKING> status MUST be advertised in the Supervision Report Command along with the expected duration for the operation. The "Status Updates" field MUST be consulted to determine if status updates are to be advertised in the future.

CC:006C.01.01.13.003 This command MAY be issued via multicast addressing.

CC:006C.01.01.11.004 A receiving node MUST NOT ignore the encapsulated command if this command is received via multicast addressing.

CC:006C.01.01.11.005 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SUPERVISION (0x6C)							
Command = SUPERVISION_GET (0x01)							
Status Updates	Reserved	Session ID					
Encapsulated Command Length							
Encapsulated Command 1							
...							
Encapsulated Command N							

Reserved

CC:006C.01.01.11.006 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Status Updates (1 bit)

This flag is used to allow a receiving node to advertise application status updates in future Supervision Report Commands.

As an example, this flag must be set to allow a supporting node to immediately advertise the <WORKING> status in a Supervision Report Command and subsequently advertise the <SUCCESS> status in another Supervision Report Command once the operation has been completed.

CC:006C.01.01.11.007 The value of this field MUST comply with Table 4.27.

Table 4.27: Supervision Get::Status Updates

Value	Required Behavior
'0'	Only return a report now
'1'	Return a report now and more later if needed

Session ID (6 bits)

The same command may be received multiple times, e.g. due to retransmissions.

CC:006C.01.01.11.008 A sending node MUST increment this field each time a new unique Supervision Get Command is issued.

CC:006C.01.01.13.004 A sending node MAY use the same Session ID for a multicast and singlecast follow-up carrying the same encapsulated command. A sending node MAY also use the same Session ID for all destinations of singlecast follow-up commands.

CC:006C.01.01.11.00D A receiving node MUST ignore duplicate singlecast commands having the same Session ID.

CC:006C.01.01.12.00E In case a Supervision Report command was not returned on a Supervision Get, the Get sending node SHOULD repeat one or a few times the command with an incremented Session ID.

CC:006C.01.01.11.00F A receiving node MUST return a Supervision Report with Status NO_SUPPORT to the sender if the Supervision Get command was not sent using the receiving node's highest granted Security Class while the command encapsulated within Supervision Get encapsulation requires its highest security.

CC:006C.01.01.11.00A A receiving node MUST abort an active operation in favor of a new command with a new Session ID if that new command affects the same resources as the active operation.

CC:006C.01.01.13.001 A receiving node MAY abort an active operation in favor of a new command with a new Session ID even if the new command does not affects the same resources as the active operation, e.g. if the node does not have resources to handle multi-session state management.

Encapsulated Command Length (1 Byte)

This field is used to specify the length of the encapsulated command.

CC:006C.01.01.11.00B The value MUST specify the number of bytes in the Encapsulated Command field.

Encapsulated Command (N bytes)

This field is used to carry an encapsulated command.

CC:006C.01.01.11.00C The length of this field MUST be in the range 1..255 bytes.

CC:006C.01.01.13.002 The field MAY carry a Multi Command Encapsulated Command which contains multiple commands.

4.2.8.4 Supervision Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SUPERVISION (0x6C)							
Command = SUPERVISION_REPORT (0x02)							
More Status Updates	Reserved	Session ID					
Status							
Duration							

Reserved

CC:006C.01.02.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

More Status Updates

This field is used to advertise if more Supervision Reports follow for the actual Session ID.

CC:006C.01.02.11.002 The value of this field MUST comply with Table 4.28.

Table 4.28: Supervision Report::More Status Updates

Value	Required Behavior
'0'	This is the last report
'1'	More reports follow according to the advertised duration

Session ID (1 Byte)

This field MUST carry the same value as the Session ID field of the Supervision Get Command which initiated this session.

Status (8 bit)

This field is used advertise the current status of the command process.

CC:006C.01.02.11.013 This field MUST reflect the actual application status of the received encapsulated command.

CC:006C.01.02.11.004 If a Multi Command encapsulated group of commands is being supervised, an error indication (such as <No Support> or <Fail>) MUST be issued if just one of the Multi Command encapsulated commands triggers an error condition.

CC:006C.01.02.11.005 The <Success> indication MUST signify that all commands carried in a Multi Command encapsulation commands have completed successfully.

CC:006C.01.02.11.006 The value of this field MUST comply with Table 4.29.

Table 4.29: Supervision Report::Status Identifiers

Value	Identifier	Description
0x00	NO_SUPPORT	The command is not supported by the receiver. CC:006C.01.02.11.007 A zero Duration value MUST be advertised. CC:006C.01.02.11.008 This identifier MUST be advertised if one or more Multi Command encapsulated commands are not supported.
0x01	WORKING	The command was accepted by the receiver and processing has started. CC:006C.01.02.11.009 A non-zero Duration value MUST be advertised. CC:006C.01.02.11.00A If processing is completed instantly, the receiver MUST skip advertising the WORKING status and return the SUCCESS status instead. CC:006C.01.02.11.00B If Status Updates was set to 1 in the Get Command, this command MUST be followed by another Supervision Report when the duration has elapsed. The new status MUST be one of WORKING, FAIL or SUCCESS. CC:006C.01.02.13.001 The duration MAY be cut short by sending the new Supervision Report before the original predicted duration has elapsed. CC:006C.01.02.11.00C If used for Multi Command encapsulated commands, the advertised Duration value MUST represent the slowest of the commands.
0x02	FAIL	The command was accepted by the receiver but processed with a resulting application status which differs from the supervised command requested Examples include but are not limited to: The node may not be ready to perform the requested operation (e.g. a door lock cannot lock if the door is open) *or* the command processing reached an unexpected situation (e.g. a door lock being jammed while working) *or* there is an application specific limitation (e.g. an irrigation controller which only accepts one open valve at a time). CC:006C.01.02.11.00D A zero Duration value MUST be advertised. CC:006C.01.02.11.00E This status identifier MUST be advertised if the processing of one or more Multi Command encapsulated commands failed.
0xFF	SUCCESS	The requested command has been completed and the application status is as the supervised command requested. CC:006C.01.02.12.001 A controlling application SHOULD NOT verify the application status, e.g. by sending a Get Command, after receiving this status identifier. CC:006C.01.02.11.00F A zero Duration value MUST be advertised. CC:006C.01.02.11.010 This identifier MUST be advertised if the processing of all Multi Command encapsulated commands was successful.

CC:006C.01.02.11.011 All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Duration (8 bits)

CC:006C.01.02.11.012 The Duration field MUST advertise the time needed to complete the current operation. The encoding of the Duration field MUST be according to Table 4.30.

Table 4.30: Supervision Report::Duration

Duration	Description
0x00	0 seconds. (Already at the Target Value.)
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80-0xFD	1 minute (0x80) to 126 minutes (0xFD) in 1 minute resolution.
0xFE	Unknown duration
0xFF	Reserved

4.2.8.5 Examples and use-cases

4.2.8.5.1 Set Type commands

CC:006C.01.00.11.002 A supporting node MUST return the Supervision status of solitary Set type commands for all its supported Command Classes.

For actuator control Set commands with a corresponding Get type command to read back the value(s),
CC:006C.01.00.11.003 this means that the response codes MUST be used as follows:

- SUCCESS: The application understood the command and completed the requested operation. The values specified in the Set Command would be returned if the supporting node would return an answer to the corresponding Get Command. The only exception are special values (e.g. value to set to the last non-zero level) representing another value.
- WORKING: The application understood the command and started performing the requested operation, but the controller needs to wait before the target value is reached. The supporting node would not return the values specified in the Set Command yet if it returned an answer to the corresponding Get Command
- FAIL: The application understood the command and cannot perform the requested operation (e.g. invalid value or mechanical failure). The supporting node will not return the values specified in the Set Command if it returned an answer to the corresponding Get Command.

The Door Lock Operation Set Command is an example of an actuator control Set command with a corresponding Get Command.

For actuator control Start and Stop commands or commands without a corresponding Get type read
CC:006C.01.00.11.004 back, this means that the response codes MUST be used as follows:

- SUCCESS: The application understood the command and completed the requested operation; i.e. successfully started or stopped the requested operation.
- FAIL: The application did not understand the command or could not perform the requested operation (if applicable)

The Multilevel Switch Start Level Change Command, Multilevel Switch Stop Level Change Command and the Powerlevel Test Node Set are examples of actuator control Start/Stop commands.

CC:006C.01.00.11.005 For configuration Set commands, this means that the response codes MUST be used as follows:

- SUCCESS: The application understood the command and accepted all the parameter(s) and value(s). The values specified in the Set Command would be returned if the supporting node would return an answer to the corresponding Get Command.
- FAIL: The application had one or more error while parsing or applying the parameters (e.g. the command was partially or completely ignored due to invalid values)

The Door Lock Configuration Set Command, Configuration Set and Association Set Command are examples of configuration commands.

4.2.8.5.2 Powerlevel Test Node Set Command

CC:006C.01.00.12.001 The Powerlevel Test Node Set Command SHOULD be considered as a Start/Stop command or command without a corresponding Get Command. The corresponding Get Command (Powerlevel Test Node Get Command) is used to query subsequent results and Supervision cannot replace the need for a controller to issue a subsequent Get Command.

CC:006C.01.00.12.002 A receiving node SHOULD return SUCCESS when starting the Powerlevel Test Node test.

4.2.8.5.3 Report/Notification Type Commands

CC:006C.01.00.13.003 A supporting node MAY use Supervision encapsulation for issuing an unsolicited Report/Notification type Command. For example, a sleeping node issuing a sensor reading receives a delivery confirmation immediately and can return to sleep as soon as the Supervision Report is received.

CC:006C.01.00.11.006 A node MUST return a Supervision status of a Report/Notification type command for any Command Classes, regardless whether it is supported, controlled or neither.

CC:006C.01.00.12.003 A controlling node SHOULD return a SUCCESS or NO_SUPPORT status when receiving such commands.

4.2.8.5.4 Remove Type commands

A supporting node MUST return the Supervision status of solitary Remove type commands for all its supported Command Classes. The remove type commands must be treated as Set type of commands.

CC:006C.01.00.11.007 For Remove command, this means that the response codes MUST be used as follows:

- SUCCESS: The application understood the command and removed the specified parameter(s) and values. The values specified in the Remove Command would not be returned if the supporting node would return an answer to the corresponding Get Command.
- FAIL: The application had one or more error while applying the command (e.g. the command was partially or completely ignored due to invalid values)

The Association Remove Command and Schedule Remove Command are examples of remove commands.

4.2.9 Supervision Command Class, version 2

The Supervision Command Class allows a sending node to request application-level delivery confirmation from a receiving node. The delivery confirmation includes relevant application-level status information in the confirmation message.

4.2.9.1 Compatibility considerations

CC:006C.02.00.21.001 Compatibility consideration requirements from version 1 MUST also be observed by a version 2 supporting node.

CC:006C.02.00.21.002 Supervision Command Class, version 2 is backwards compatible with Supervision Command Class, version 1. Fields and commands not described in this version MUST remain unchanged from version 1.

The Supervision Report Command has been extended to enable expedited message delivery between Wake Up Destination and Wake Up Node (refer to the Wake Up Command Class). It leverages sleeping nodes using Supervision encapsulation to tell them to initiate a Wake Up Period.

4.2.9.2 Supervision Report Command

This command is used to advertise the status of one or more command process(es).

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SUPERVISION (0x6C)							
Command = SUPERVISION_REPORT (0x02)							
More Status Updates	Wake Up Request	Session ID					
Status							
Duration							

All fields not described below remain unchanged from version 1

Wake Up Request (1 bit)

CC:006C.02.02.11.001 This field is used to indicate to the receiving node that it MUST start a Wake Up Period.

CC:006C.02.02.11.002 If the receiving node does not support the Wake Up On Demand functionality, this field MUST be ignored.

The Wake Up On Demand functionality is part of the Wake Up Command Class, version 3.

CC:006C.02.02.11.003 A node supporting the Wake Up On Demand functionality MUST return a Wake Up Notification Command if the Wake Up destination node issued a this command with the Wake Up Request bit set to 1.

CC:006C.02.02.11.004 A node supporting the Wake Up On Demand functionality MUST ignore the Wake Up Request field if the Supervision Report is not issued by the Wake Up Destination.

4.2.9.2.1 Wake up on Demand functionality

The Wake Up on Demand functionality is shown in Figure 4.24. This functionality allows the controller to issue important commands to the sleeping device before its next Wake Up Period.

CC:006C.02.02.13.001 When the controlling node receives a Supervision encapsulated frame from the sleeping node, it MAY speed up the delivery of some important commands by setting the *Wake Up Request* bit to 1, if the sleeping node supports this functionality.

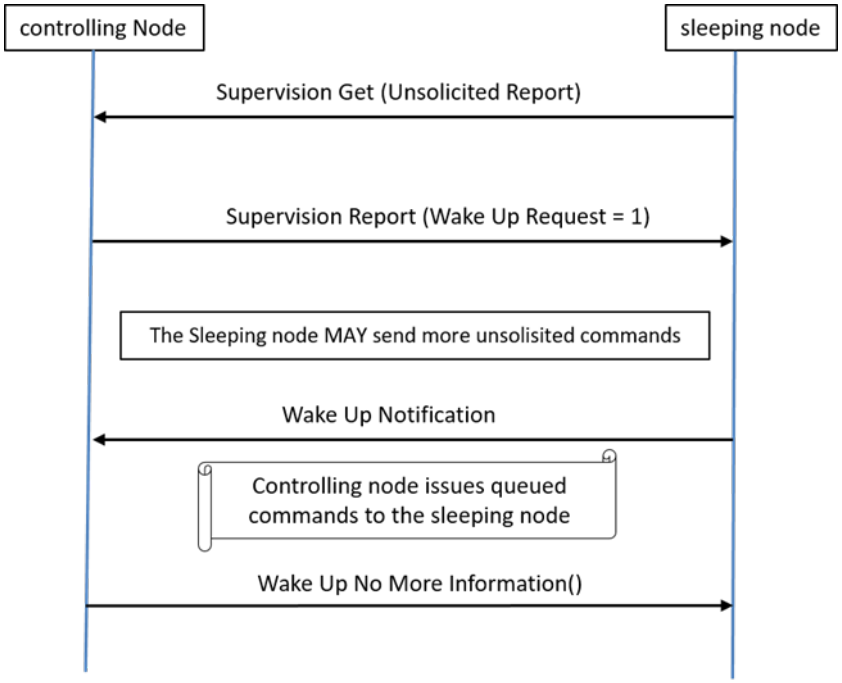


Figure 4.24: Wake Up on Demand functionality

4.2.10 Transport Service Command Class, version 1

THIS COMMAND HAS BEEN OBSOLETED

New implementations MUST use the Transport Service Command Class version 2.

The Transport Service Command Class Version 2 redefines the frame formats used by the command class.

4.2.11 Transport Service Command Class, version 2

The Transport Service Command Class supports the transfer of datagrams larger than the Z-Wave frame.

The Transport Service Command Class, version 2 is defined by [20].

The following sections provide additional requirements and frame flows.

4.2.11.1 Compatibility considerations

A node supporting the Transport Service Command Class, version 2:

CC:0055.02.00.21.001 • MUST NOT send Transport Service segments with the Payload field longer than 39 bytes.

CC:0055.02.00.21.002 • MUST accept datagrams up to 117 bytes long (3 segments of 39 bytes each).

CC:0055.02.00.23.001 • MAY accept larger datagrams.

4.2.11.1.1 Node Information Frame (NIF)

A node supporting the Transport Service Command Class, version 2:

CC:0055.02.00.21.003 • MUST always advertise this Command Class in its NIF, regardless of the inclusion status and security bootstrapping outcome.

CC:0055.02.00.21.004 • MUST NOT advertise this Command Class in its S0/S2 Supported Command Class list or in the Multi Channel End Point capabilities.

4.2.11.2 Example Frame flows

CC:0055.02.00.11.001 A supporting node MUST comply with the following frame flows.

4.2.11.2.1 As things should always work - the default case

- Node A initiates a 117-byte frame transmission to Node B
  - Node A sends FirstSegment(datagram size = 117, Session ID = 10, Payload = bytes 1..39)
  - Node B receives the FirstSegment with valid Transport Service FCS (16-bit checksum) and valid MPDU FCS (8 or 16 bits checksum, depending on transmission speed)
    - Node B creates a tracking list for the datagram; bytes 1..39 are marked as received
    - Node B starts segment rx timer
  - Node A sends SubsequentSegment(datagram size = 117, datagram offset = 39, Session ID = 10, Payload = bytes 40..78)
  - Node B receives the SubsequentSegment with valid Transport Service and MPDU FCS
    - Node B updates the tracking list for the datagram; bytes 40..78 are marked as received
    - Node B (re-)starts segment rx timer
  - Node A sends SubsequentSegment(datagram size = 117, datagram offset = 78, Session ID = 10, Payload = bytes 79..117)
    - Node A starts a segment_complete tx timer
  - Node B receives SubsequentSegment with valid Transport Service and MPDU FCS
    - Node B updates the tracking list for the datagram; bytes 79..117 are marked as received, indicating that this was the last segment

        * Node B checks the tracking list for missing segments; none found
- Node B sends SegmentComplete(Session ID = 10)
- Node A receives SegmentComplete(Session ID = 10) with valid MPDU FCS (8 or 16 bits checksum, depending on transmission speed)

4.2.11.2.2 Losing first segment of a long message

- Node A initiates a 117-byte frame transmission to Node B:
  - Node A sends FirstSegment(datagram size = 117, Session ID = 10, Payload = bytes 1..39).
  - Node B receives FirstSegment invalid Transport Service or MPDU FCS (the command is ignored).
  - Node A sends SubsequentSegment(datagram offset = 39)
  - Node B receives the SubsequentSegment correctly (valid Transport Service and MPDU FCS)
  - Node B sends SegmentWait(Pending segments=0) because no session is open.
  - Node A waits and restarts the transmission from the FirstSegment.

4.2.11.2.3 Losing subsequent segment

- Node A initiates a 117-byte frame transmission to Node B
  - Node A sends FirstSegment(datagram size = 117, Session ID = 10, Payload = bytes 1..39)
  - Node B receives the FirstSegment correctly
    * Node B creates a tracking list for the datagram; bytes 1..39 are marked as received
    * Node B starts segment rx timer
  - Node A sends SubsequentSegment(datagram offset = 39)
  - Node B receives SubsequentSegment with invalid Transport Service or MPDU FCS. (the command is ignored)
  - Node A sends SubsequentSegment(datagram offset = 78)
    * Node A starts segment_complete tx timer
  - Node B receives SubsequentSegment correctly
    * Node B updates the tracking list for the datagram, indicating that this was the last segment
    * Node B checks tracking list for the datagram; bytes 40..78 are missing
  - Node B sends SegmentRequest(datagram offset = 39)
  - Node A receives SegmentRequest(datagram offset = 39) correctly
  - Node A send SubsequentSegment(datagram offset = 39)
  - Node B receives SubsequentSegment(datagram offset = 39) correctly
    * Node B updates the tracking list for the datagram
    * Node B checks the tracking list for missing segments; none found
    * Node B clears segment rx timer
- Node B sends SegmentComplete(Session ID = 10)
- Node A receives SegmentComplete(Session ID = 10) correctly

4.2.11.2.4 Losing last segment

- Node A initiates a 117-byte frame transmission to Node B
  - Node A sends FirstSegment(datagram size = 117, Session ID = 10, Payload = bytes 1..39)
  - Node B receives FirstSegment
    - Node B creates a tracking list for the datagram; bytes 1..39 are marked as received
    - Node B starts segment rx timer
  - Node A sends SubsequentSegment(datagram offset = 39)
  - Node B receives SubsequentSegment correctly
    - Node B updates the tracking list for the datagram; bytes 40..78 are marked as received
    - Node B (re-)starts segment rx timer
  - Node A sends (the last) SubsequentSegment(datagram offset = 78)
    - Node A starts segment_complete tx timer
  - Node B receives SubsequentSegment with invalid Transport Service or MPDU FCS. (the command is ignored)
  - Node B segment rx timer times out.
    - Node B checks tracking list for the datagram; bytes 79..117 are missing
  - Node B sends SegmentRequest(datagram offset = 78)
    - Node B starts a segment rx timer to wait for the SubsequentSegment frame
    - If the segment rx timer times out, Node B bails out: discard all received segments and return to idle (e.g. sender may be down or sleeping)
  - Node A receives SegmentRequest(datagram offset = 78)
  - Node A sends SubsequentSegment(datagram offset = 78)
  - Node B receives SubsequentSegment(datagram offset = 78)
    - Node B updates the tracking list for the datagram; bytes 79..117 are marked as received
    - Node B checks tracking list for missing segments; none found
  - Node B sends SegmentComplete(Session ID = 10)
  - Node A receives SegmentComplete(Session ID = 10) correctly
    - Node A stops the segment_complete tx timer

4.2.11.2.5 Losing SegmentComplete

- Node A initiates a 117-byte frame transmission to Node B
  - Node A sends FirstSegment(datagram size = 117, Session ID = 10, Payload = bytes 1..39)
  - Node B receives FirstSegment correctly
    - Node B creates tracking list for the datagram; bytes 1..39 are marked as received
    - Node B starts a segment rx timer
  - Node A sends SubsequentSegment(datagram offset = 39)
  - Node B receives SubsequentSegment(datagram offset = 39) correctly
    - Node B updates the tracking list for the datagram; bytes 40..78 are marked as received
    - Node B (re-)starts segment rx timer
  - Node A sends SubsequentSegment(datagram offset = 78)

    - Node A starts a segment_complete timer
- Node B receives SubsequentSegment(datagram offset = 78) correctly.
    - Node B updates the tracking list for the datagram; bytes 79..117 are marked as received, indicating that this was the last segment
    - Node B checks tracking list for missing segments; none found
- Node B sends SegmentComplete(Session ID = 10)
- Node A receives SegmentComplete with an invalid MPDU FCS. (the command is ignored)
- Node A segment_complete tx timer times out
    - Node A sends SubsequentSegment(datagram offset = 78) one more time
        - Node A starts a segment_complete timer again.
        - If the segment_complete timer times out, Node A bails out: return "Error" callback to calling application
- Node B receives SubsequentSegment(datagram offset = 78) correctly
    - Node B updates the tracking list for the datagram; bytes 79..117 are marked as received, indicating that this was the last segment
    - Node B checks tracking list for missing segments; none found
- Node B sends SegmentComplete(Session ID = 10) once more
- Node A receives SegmentComplete(Session ID = 10) correctly
- Node A stops the segment_complete timer

5 Network-Protocol Command Classes

5.1 Network-Protocol Command Class Overview

General Command Class overview and rules are described in *Application Command Classes* and are valid for the Command Classes presented in this document.

No additional considerations apply for the Network-Protocol Command Classes.

5.2 Network-Protocol Command Class Definitions

5.2.1 Inclusion Controller Command Class, version 1

The Inclusion Controller Command Class is used after a node's network inclusion between the SIS and an inclusion controller to inform each other of the remaining setup required for the included node.

Examples of such setup operations could be Z-Wave Plus Lifeline configuration or Security 2 bootstrapping.

If the S2 bootstrapping is handled by a SIS after the Z-Wave network inclusion has been handled by an inclusion controller, the joining node will detect two different NodeIDs for Network inclusion and S2 bootstrapping. The NodeID of the including controller is not relevant for the authentication of the
CC:0074.01.00.11.001 joining node. Therefore, the joining node MUST NOT abort the S2 bootstrapping in response to a changing NodeID.

CC:0074.01.00.11.002 The SIS, inclusion controller and joining node MUST follow the frame flow illustrated in Figure 1.

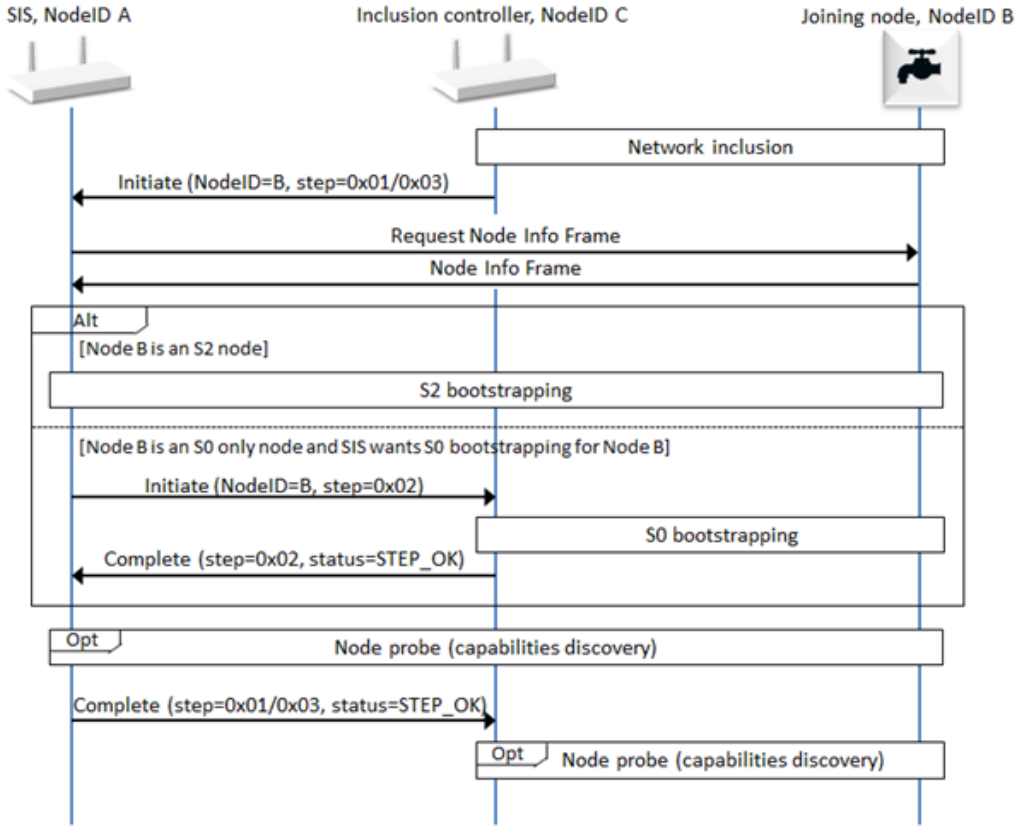


Figure 5.1: Inclusion Controller Frame Flow

CC:0074.01.00.11.003 The SIS, inclusion controller and joining node MUST comply with the following steps:

1. Inclusion Controller, C, performs network inclusion of Joining Node, B.

CC:0074.01.00.11.004
2. Inclusion Controller, C, MUST send Inclusion Controller Initiate to SIS, A, immediately following the network inclusion.

CC:0074.01.00.11.005
CC:0074.01.00.11.006
3. SIS, A, MUST request a Node Info Frame from Joining Node, B.
4. Joining Node, B, MUST respond to SIS, A, with a Node Info Frame

Option 1: If Joining Node B supports S2:

CC:0074.01.00.11.007
1. SIS, A, MUST start the Security 2 bootstrapping as described in *Security 2 (S2) Command Class, version 1*, including user dialogs.

CC:0074.01.00.11.008

2. Joining Node, B, MUST accept being S2 bootstrapped by the SIS

Option 2: If Joining Node, B does not support S2 and supports S0

7. If SIS, A wants S0 bootstrapping performed for Joining Node, B, it will send an Inclusion Controller Initiate(S0_INCLUSION) to Inclusion Controller, C

CC:0074.01.00.11.00A

8. Inclusion Controller, C MUST perform S0 bootstrapping if it has the S0 network key after receiving Inclusion Controller Initiate(S0_INCLUSION)

CC:0074.01.00.11.00B

9. Inclusion Controller, C MUST return an Inclusion Controller Complete to SIS, A to indicate if S0 bootstrapping attempt took place and if it was successful.

Following the Security bootstrapping, regardless whether it failed, successful or was not applicable:

CC:0074.01.00.12.001

10. SIS, A, SHOULD perform any probing needed of the Joining Node, B.

CC:0074.01.00.11.009

CC:0074.01.00.12.002

11. SIS, A, MUST send an Inclusion Controller Complete Command to the Inclusion Controller, C.

12. Inclusion Controller, C, SHOULD perform any probing needed of the Joining Node, B.

5.2.1.1 Compatibility considerations

5.2.1.1.1 Node information frame (NIF)

CC:0074.01.00.21.002

A supporting node MUST always advertise the Inclusion Controller Command Class in its NIF, regardless of the security bootstrapping outcome when having the SIS or Inclusion Controller role.

CC:0074.01.00.23.001

A supporting node MAY keep or remove the Inclusion Controller Command Class in/from its NIF if it has the primary or secondary controller role.

5.2.1.1.2 Legacy controllers

If an Inclusion Controller that does not support the Inclusion Controller Command Class includes a new node in a network, the SIS will never receive an Inclusion Controller Initiate Command. If no Initiate Command has been received approximately 10 seconds after a new node has been added to a network, the SIS SHOULD start interviewing the newly included node (step 10 above).

CC:0074.01.00.22.001

CC:0074.01.00.21.003

If an Inclusion Controller includes a node and the SIS does not support the Inclusion Controller Command Class, the Inclusion Controller MUST perform S0 bootstrapping immediately after inclusion if applicable.

5.2.1.2 Inclusion controller initiate command

This command is used to ask a receiving node to perform specific steps in the inclusion/bootstrapping process.

The initiate command asks the controller to perform a specific step of the inclusion process. The Inclusion Controller Initiate Command is first sent from an inclusion controller to the SIS, then the

CC:0074.01.01.13.001

SIS MAY choose to perform the rest of the inclusion by itself or it MAY ask the inclusion controller to perform one or more of the inclusion steps.

CC:0074.01.01.11.001

This command MUST be sent through highest common Security Class of the SIS and Inclusion Controller, if no common Security Class exists, non-secure is allowed. Inclusion Controllers MUST send this command following a successful network inclusion. It also means that if the SIS receives this command at a less-secure than the highest common Security class, it MUST ignore this command. E.g. Non-secure Initiate Commands MUST be ignored by the SIS, unless the SIS has a record of

CC:0074.01.01.11.005

including the Inclusion Controller non-securely.

CC:0074.01.01.11.002

This command MUST NOT be issued via multicast addressing. A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.1: Inclusion Controller Initiate Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INCLUSION_CONTROLLER (0x74)							
Command = INITIATE (0x01)							
Node ID							
Step ID							

Node ID

CC:0074.01.01.11.003 This field is used to indicate the NodeID of the node being included. The receiving node MUST perform the steps on the NodeID indicated by this field

Step ID

CC:0074.01.01.11.004 This field is used to indicate which step is to be performed on the specified. The field MUST comply with :Table 5.2

Table 5.2: Inclusion Controller Initiate::Step ID encoding

Value	Identifier	Description
0x01	PROXY_INCLUSION	This Value MUST be used only when: • The sending node is the inclusion controller • The receiving node is the SIS This Value is used to indicate the SIS that it MUST take over the node inclusion and perform S2 bootstrapping if releveant. The SIS MUST return an Inclusion Controller Complete Command when the step has been completed. The SIS MAY ask the inclusion controller to perform some of the steps by itself before returning an Inclusion Controller Complete Command.
0x02	S0_INCLUSION	This value MUST be used only when: • The sending node is the SIS • The receiving node is the inclusion controller This value is used to indicate to the inclusion controller that it MUST perform S0 bootstrapping. The inclusion controller MUST reply with an Inclusion Controller Complete Command when the S0 bootstrapping has been performed (or attempted).
0x03	PROXY_INCLUSION_REPLACE	This value MUST be used only when: • The sending node is the inclusion controller • The receiving node is the SIS This value is identical to PROXY_INCLUSION but is used in case the newly included node has replaced a failed node. This value is used to indicate the SIS that it MUST take over the node inclusion and perform S2 bootstrapping if relevant. The SIS MUST return an Inclusion Controller Complete Command when the step has been completed. The SIS MAY ask the inclusion controller to perform some of the steps by itself before returning an Inclusion Controller Complete Command.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

5.2.1.3 Inclusion controller complete command

CC:0074.01.02.11.001 This command MUST be sent after a controller has completed the requested inclusion steps.

CC:0074.01.02.11.002 This command MUST be sent using the highest common Security Class of the SIS and Inclusion Controller. If no common Security Class exists, non-secure transmission is allowed.

CC:0074.01.02.11.003 An inclusion controller MUST perform optional node interview after receiving a Inclusion Controller
CC:0074.01.02.11.004 Complete Command with Step ID, PROXY_INCLUSION. A SIS MUST do its device probe before sending the COMPLETE command with step ID PROXY_INCLUSION.

Table 5.3: Inclusion Controller Complete Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INCLUSION_CONTROLLER (0x74)							
Command = COMPLETE (0x02)							
Step ID							
Status							

Step ID

This field is used to indicate the step that has been completed.

CC:0074.01.02.11.005 A sending node MUST set this field to the same value as the last received Inclusion Controller Initiate Command.

Status

CC:0074.01.02.11.006 This field is used to indicate the status of the advertised Step ID. It MUST comply with Table 5.4

Table 5.4: Inclusion Controller Complete::Status encoding

Value	Status CODE identifier	Description
0x01	STEP_OK	The performed step was completed without error.
0x02	STEP_USER_REJECTED	The step was rejected by user
0x03	STEP_FAILED	The step failed, because of a communication or protocol error.
0x04	STEP_NOT_SUPPORTED	The step failed, because it Is not supported by the sending node.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

5.2.2 IP Configuration Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED New implementations MUST NOT use the IP configuration Command Class. Refer to the Z/IP and Network management Command Classes.

The IP Configuration Command Class is used to configure network identifiers for IPV4 devices. The intended use of the command class is illustrated in the figure below.

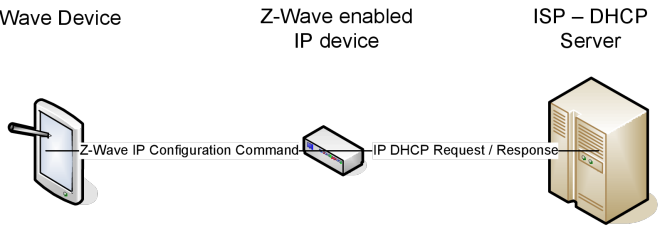


Figure 5.2: Configuration of network identifiers for IPV4 devices

In the figure the Z-Wave Remote to the left, sends an IP Configuration Command to the Z-Wave enabled IP device, telling it to acquire its configuration using DHCP. The Z-Wave enabled IP device will now perform a standard DHCP IP request to the DHCP server over an IP based network.

Another example might be where the Z-Wave Remote statically configures the Z-Wave enabled IP device with fixed IP, subnet, DNS etc. by sending an IP Configuration Command.

Note that this class is only intended for IPV4 and not IPV6 support.

5.2.2.1 IP configuration set command

The IP Configuration Set Command used to configure IPV4 settings in a device.

Table 5.5: IP Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION (0x9A)							
Command = IP_CONFIGURATION_SET (0x01)							
Reserved						Auto IP	Auto DNS
IP Address 1							
IP Address 2							
IP Address 3							
IP Address 4							
Subnet Mask 1							
Subnet Mask 2							
Subnet Mask 3							
Subnet Mask 4							
Gateway 1							
Gateway 2							
Gateway 3							
Gateway 4							
DNS1 1							
DNS1 2							
DNS1 3							
DNS1 4							
DNS2 1							
DNS2 2							
DNS2 3							
DNS2 4							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Auto IP (1bit)

If Auto IP bit is set, the following fields are ignored: IP Address, Subnet Mask, and Gateway. And are allocated by DHCP or BOOTP instead.

Auto DNS (1bit)

The Auto DNS if set indicates to ignore DNS1 and DNS2 and allocate DNS by DHCP instead. Note that some devices might not support Auto DNS without Auto IP set.

IP Address (32 bit)

The IP Address indicates the static IP address of the device itself. The first byte is the most significant byte.

Subnet mask (32 bits)

The Subnet Mask determines the portion of the IP address that represents the subnet. The first byte is the most significant byte.

Gateway (32 bits)

The Gateway indicates the default gateway that serves as an access point to another network. The first byte is the most significant byte.

DNS1 (32 bits)

The DNS1 allows the use of domain name system (DNS) server names instead of using numerical IP addresses for management packet routing. In case the device will not need DNS, and SHOULD NOT query it from DHCP then leave field as all zeroes. The first byte is the most significant byte.

DNS2 (32 bits)

The DNS2 provides a secondary DNS server name. In case only one DNS server is available or the device will not need DNS then leave field as all zeroes. The first byte is the most significant byte.

5.2.2.2 IP configuration get command

The IP Configuration Get Command is used to request the IPV4 settings in a device.

The IP Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing. A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.6: IP Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION (0x9A)							
Command = IP_CONFIGURATION_GET (0x02)							

5.2.2.3 IP configuration report command

The IP Configuration Report Command used to return IPV4 settings in a device.

Table 5.7: IP Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION (0x9A)							
Command = IP_CONFIGURATION_REPORT (0x03)							
Reserved						Auto IP	Auto DNS
IP Address 1							
IP Address 2							
IP Address 3							
IP Address 4							
Subnet Mask 1							
Subnet Mask 2							
Subnet Mask 3							
Subnet Mask 4							
Gateway 1							
Gateway 2							
Gateway 3							
Gateway 4							
DNS1 1							
DNS1 2							
DNS1 3							
DNS1 4							
DNS2 1							
DNS2 2							
DNS2 3							
DNS2 4							
LeaseTime 1							
LeaseTime 2							
LeaseTime 3							
LeaseTime 4							

Refer to explanation of parameters in IP Configuration Set Command description.

Lease Time (32 bits)

The lease time specifies the time the IP address has been granted, if Auto IP is being used (in seconds). If the device does not know its lease period it MUST return 0 for the lease time fields.

5.2.2.4 IP configuration DHCP release command

The IP Configuration DHCP Release Command used to release the DHCP lease.

Table 5.8: IP Configuration DHCP Release Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION (0x9A)							
Command = IP_CONFIGURATION_RELEASE (0x04)							

5.2.2.5 IP configuration DHCP renew command

The IP Configuration DHCP Renew Command used to force the renewal of the DHCP lease.

Table 5.9: IP Configuration DHCP Renew Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION (0x9A)							
Command = IP_CONFIGURATION_RENEW (0x05)							

5.2.3 Mailbox Command Class, version 1

The Mailbox Command Class is intended for IP based gateway deployments with distributed mailbox resources. One example is a constrained gateway device which is offloaded by another IP host with sufficient memory to host the Mailbox Service. The Mailbox Service may be hosted by a LAN host or an Internet server.

The Mailbox Command Class allows any mailbox capable device to either make itself into a Mailbox Service, or utilize another Mailbox Service in the network.

5.2.3.1 Mailbox framework

The Mailbox Command Class describes a framework that consists of two specific Mailbox Modes described below:

1. Mailbox Proxy, which forwards mailbox requests to a Mailbox Service.
2. Mailbox Service, which accepts the forwarded mailbox requests and stores them until the designated recipient announces that it is awake.

A mailbox device MAY support one or both of the two Mailbox Modes. However, a mailbox device MUST NOT take both Mailbox Modes in a network.

Before configuring Mailbox Proxy forwarding, a configuring node MUST ensure that the forwarding and receiving devices support their respective required modes. The information can be found using the Mailbox Configuration Get Command and Mailbox Configuration Report Command.

5.2.3.1.1 Mailbox proxy

The Mailbox Proxy device forwards all received frames that are destined for a non-listening node to the configured Mailbox Service. Before forwarding the frame, it MUST be attempted to send the frame to the node first as it may be awake following a manual activation or inclusion. If the Mailbox Proxy can deliver the frame to the non-listening node, the Mailbox Proxy MUST NOT forward the frame to the Mailbox Service.

The Mailbox Proxy MUST support the Wake Up Command Class.

5.2.3.1.2 Mailbox service

The Mailbox Service serves as a conventional mailbox, with the addition that it may receive forwarded frames from a Mailbox Proxy. A Mailbox Service may have a finite mailbox queue capacity, which is reported in the Mailbox Configuration Report. The Mailbox Service MUST NOT communicate with a Z/IP client directly, since it may not be able to route messages to the client.

5.2.3.1.3 Frame flow

Figure 5.3 illustrates the communication between a Z/IP Client (1) attempting communication to a non-listening node (4). The communication is passing through the Mailbox Proxy (2) which initially will attempt direct communication with (4). If failing to reach (4), the frame will be forwarded to the Mailbox Service (3) using the Mailbox Queue Command with Push Operation.

Following the Mailbox Queue push, the Mailbox Service will send a Mailbox Queue Command with Waiting Operation to the proxy, piggybacking the original UDP command on the message. The Proxy will build a ”NACK Waiting” Z/IP Command targeted for the Z/IP node, based on the piggy backed message from the Proxy Service. The Proxy Service MUST also append the Expected Delay header extension to the ”NACK Waiting” Z/IP Command. This step MUST be repeated every 60s seconds as long as the message is in the mailbox.

Upon wake-up, the non-listening node (4) will transmit a Wake Up Notification to the Mailbox Proxy (2), which must be configured using the Wake Up Command Class. Whenever the Mailbox Proxy (2) receives a Wake Up Notification, the notification will be forwarded as a Z/IP Packet to the Mailbox Service (4). The Mailbox Service inspects the queue to see if there are any frames for (4) and responds with either an empty Mailbox Queue Command Pop operation with ”Last” bit set to 1 or any frames that may be in queue, finishing with the last frame having ”Last” bit set to 1.

Mailbox Proxy (2) receives the Mailbox Queue Pop frame on which it performs a Virtual Node Rewrite to match the original sender of the UDP frame of the Mailbox Queue Pop command. The frame is sent from the virtual node to (4) followed by a ”Wake Up No More Information” Command. Any eventual reports will be replied to the virtual node that forwards them to (1). The proxy MUST send a Mailbox Queue Command with ACK operation to the Proxy Service when it has delivered the frame and potentially the ”No more information”

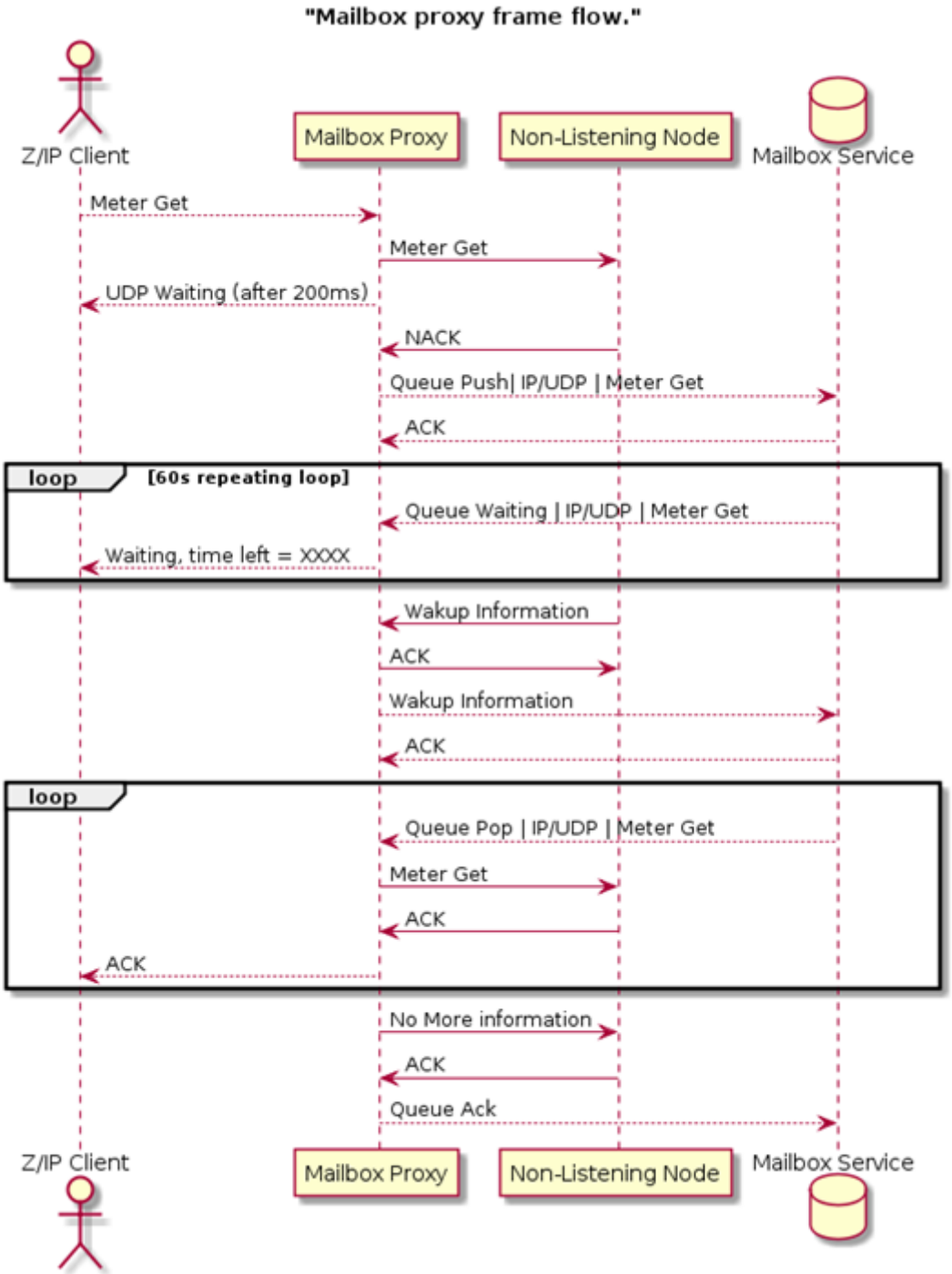


Figure 5.3: Mailbox Frame Flow

5.2.3.2 Mailbox configuration get command

The Mailbox Configuration Get Command is used to request the Mailbox configuration from a supporting device.

The Mailbox Configuration Report command MUST be returned in response to a Mailbox Configuration Get command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.10: Mailbox Configuration Get Comand

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX (0x69)							
Command = MAILBOX_CONFIGURATION_GET (0x01)							

5.2.3.3 Mailbox configuration set command

Table 5.11: Mailbox Configuration Set Comand

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX (0x69)							
Command = MAILBOX_CONFIGURATION_SET (0x02)							
Reserved					Mode		
Forwarding Destination Ipv6 Address – Byte 1							
...							
Forwarding Destination Ipv6 Address – Byte 16							
UDP Port Number – Byte 1							
UDP Port Number – Byte 2							

Reserved (5 bits)

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Mode (3 bits)

The Mode field is used to advertise the Mailbox mode to be configured in the node. This field MUST be encoded according to Table 5.12.

Table 5.12: Mailbox Configuration Set::Mode encoding

Value	Description
0x00	Disable Mailbox Service Disable Mailbox Proxy forwarding
0x01	Enable Mailbox Service
0x02	Enable Mailbox Proxy forwarding

Forwarding Destination Ipv6 Address (16 bytes)

If the Mailbox Proxy Forwarding is enabled in the Mode field, the Forwarding Destination Ipv6 Address field MUST specify the Forwarding Destination Ipv6 Address. The field MUST specify an Ipv6 formatted address of the Mailbox Service to receive forwarded mailbox packages. If the Forwarding Destination is identified by an Ipv4 address this field MUST be formatted as an Ipv4-mapped Ipv6 address RFC 4291.

If the Mailbox Proxy Forwarding is not enabled in the Mode field, the Forwarding Destination Ipv6 Address MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

UDP Port Number (2 bytes)

This field indicates the UDP Port number of the Mailbox Service running at the Forwarding Destination.

If the Mailbox Proxy Forwarding is not enabled in the Mode field, this field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

5.2.3.4 Mailbox configuration report command

Table 5.13: Mailbox Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX (0x69)							
Command = MAILBOX_CONFIGURATION_REPORT (0x03)							
Reserved			Supported Modes		Mode		
Mailbox Capacity – Byte 1							
Mailbox Capacity – Byte 2							
Forwarding Destination Ipv6 Address – Byte 1							
...							
Forwarding Destination Ipv6 Address – Byte 16							
UDP Port Number – Byte 1							
UDP Port Number – Byte 2							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Supported Modes (2 bits)

The Supported Modes bit field is used to advertise the functionalities supported by the node. This field MUST be encoded according to Table 5.14

Table 5.14: Mailbox Configuration Report::Supported Modes encoding

Value	Description
0x01	Mailbox Service supported
0x02	Mailbox Proxy supported

Mode (3 bits)

Refer to Section 5.2.3.3.

Mailbox Capacity (2 bytes)

This field advertises the number of frames (at a maximum of 1280 bytes per frame) that may be stored in the mailbox while waiting for a Wake Up Notification.

A value of 0 MUST indicate that the mailbox will only support mailbox forwarding to another Mailbox Service.

A value of 0xFFFF MUST indicate that the mailbox in effect have no storage limitation.

Forwarding Destination Ipv6 Address (16 bytes)

Refer to Section 5.2.3.3.

UDP Port Number (2 bytes)

Refer to Section 5.2.3.3.

5.2.3.5 Mailbox queue command

The Mailbox Queue Command is a container for various operations between a mailbox proxy and a Mailbox Service.

Table 5.15: Mailbox Queue Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX (0x69)							
Command = MAILBOX_QUEUE (0x04)							
Reserved				Last	Operation		
Queue Handle							
Mailbox Entry – Byte 1							
...							
Mailbox Entry – Byte N							

Reserved (6 Bit)

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Last (1 bit)

The Last field is used to indicate if the current mailbox frame is the last in the queue for the specific device. The Last bit only applies when the ”Pop” Operation is used.

The value 1 MUST indicate that the frame is the last on the queue.

The value 0 MUST indicate that more frames will follow.

Operation (3 bits)

The encoding of the Operation field MUST be according to Table 5.16.

Table 5.16: Mailbox Queue::Operation

Value	Description
0x00	Push. Queue a message from the proxy to the Mailbox Service
0x01	Pop. Dequeue a message from the Mailbox Service to the Mailbox Proxy for delivery on the PAN
0x02	Waiting. Service->Proxy: send waiting messages to the client.
0x03	Ping. Service->Proxy: send UDP ping messages to the client.
0x04	ACK. Proxy->Service: Frame has been delivered. Service->Proxy: Frame has been queued.
0x05	NACK. Proxy->Service: Frame was not queued. Wait for ACK before attempting queuing. Service->Proxy: Node is not responding. Keep in queue.
0x06	Queue Full. Proxy->Service: The capacity of the Mailbox Service has been reached. Wait until queue has been emptied.

All other values are reserved and MUST NOT be used by a sending node.

Reserved values MUST be ignored by a receiving node.

Queue Handle (8 bits)

The Queue Handle field is used to identify the queue this message belongs to. A service uses this handle with the source IP of the MAILBOX_QUEUE message to identify the queue to which a

message belongs to.

Mailbox Entry (N Bytes)

The Mailbox Entry field contains the entire received UDP Package. Including, ZIP headers and Z-Wave Payload.

To avoid duplicate entries, the Mailbox Service MUST maintain a list of CRC16 checksums for each mailbox entry. All mailbox entries MUST be unique, if a matching CRC16 exists for an incoming package, the incoming package MUST be discarded.

When WAITING timer elapses the mailbox MUST send a WAITING message to all clients that has posted entries to the mailbox.

5.2.3.6 Mailbox wake up notification command

This command allows a mailbox proxy resource to notify a Mailbox Service resource that a wake up device is currently awake.

A Mailbox Proxy resource MAY send this command to a Mailbox Service resource.

A Mailbox Service resource MUST NOT send this command to a mailbox proxy resource.

Table 5.17: Mailbox Wake Up Notification Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX (0x69)							
Command = MAILBOX_WAKEUP_NOTIFICATION (0x05)							
Queue Handle							

Queue Handle (8 bits)

This field is used to specify the actual queue handle to send notification to.

5.2.3.7 Mailbox failing node command

This command allows a mailbox proxy resource to notify a Mailbox Service resource that a wake up device is no longer available.

A Mailbox Proxy resource MAY send this command to a Mailbox Service resource.

A Mailbox Service resource MUST NOT send this command to a mailbox proxy resource.

Table 5.18: Mailbox Failing Node Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX (0x69)							
Command = MAILBOX_NODE_FAILING (0x06)							
Queue Handle							

Queue Handle (8 bits)

This field is used to specify the actual queue.

A receiving Mailbox Service resource MUST discard all state information and enqueued messages for the actual queue.

5.2.3.8 Frame flow diagrams Examples

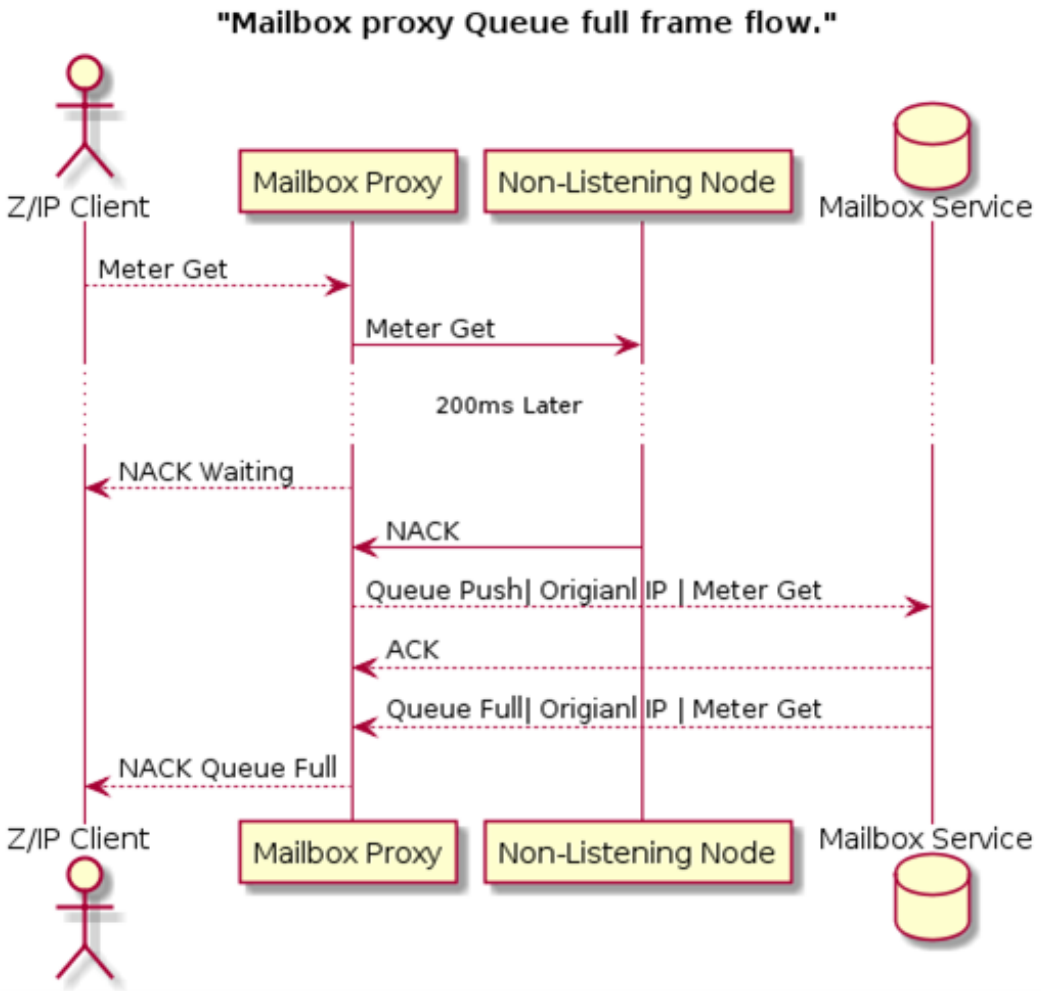


Figure 5.4: Mailbox proxy queue full frame flow

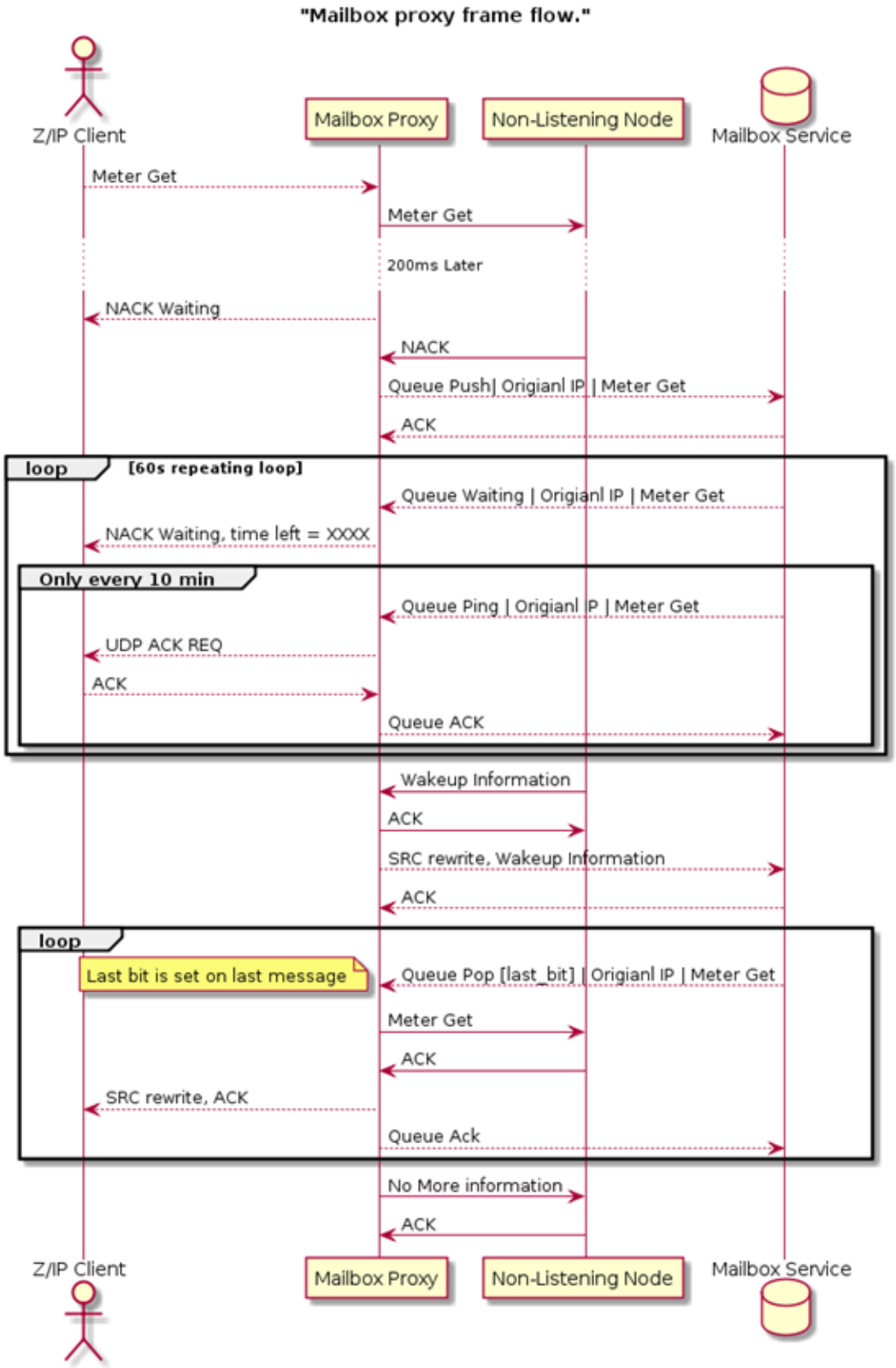


Figure 5.5: Normal frame flow

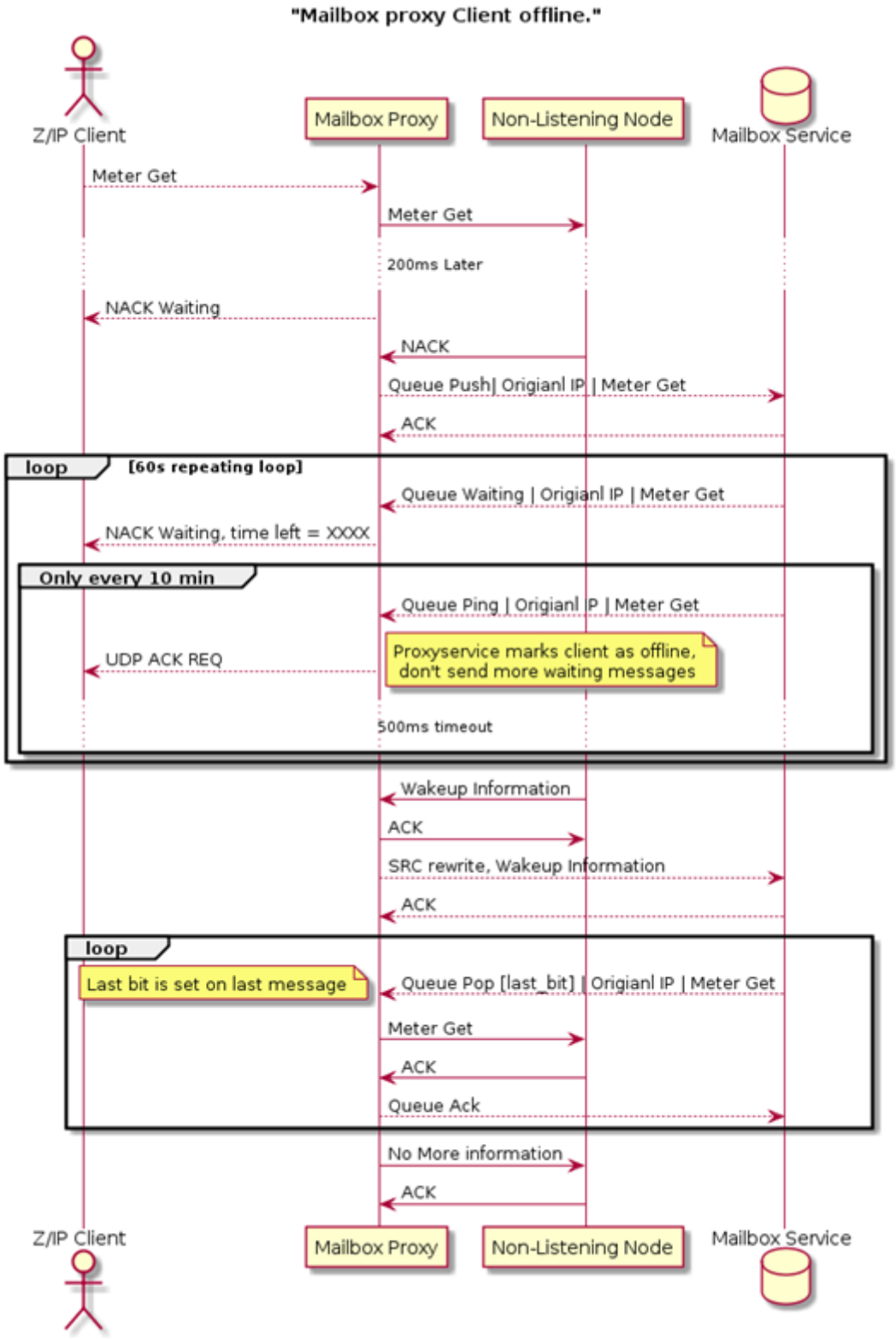


Figure 5.6: Z/IP Client goes offline and stops replying to UDP ping

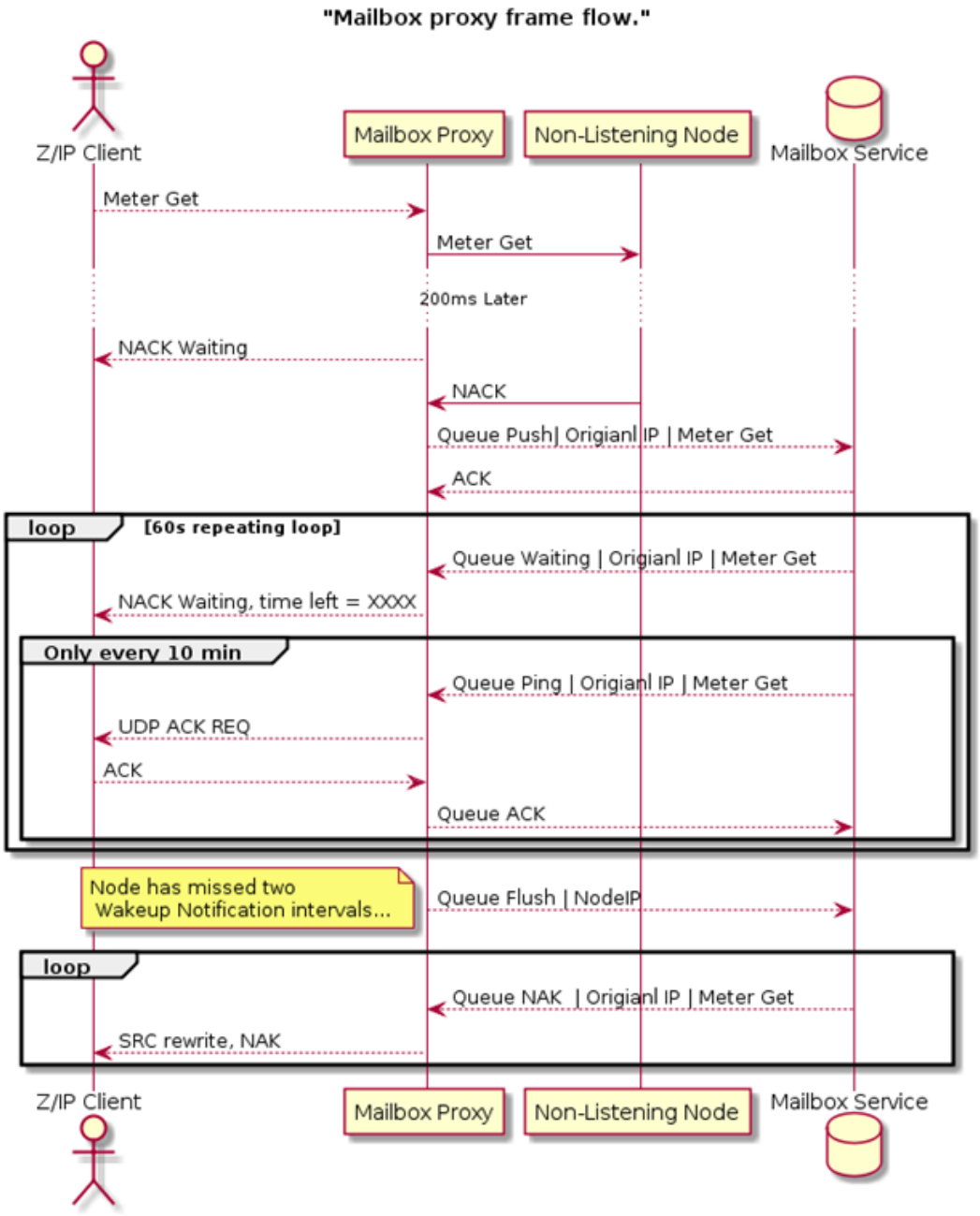


Figure 5.7: Sleeping node misses 2 Wake Up intervals and proxy tells service to flush queue

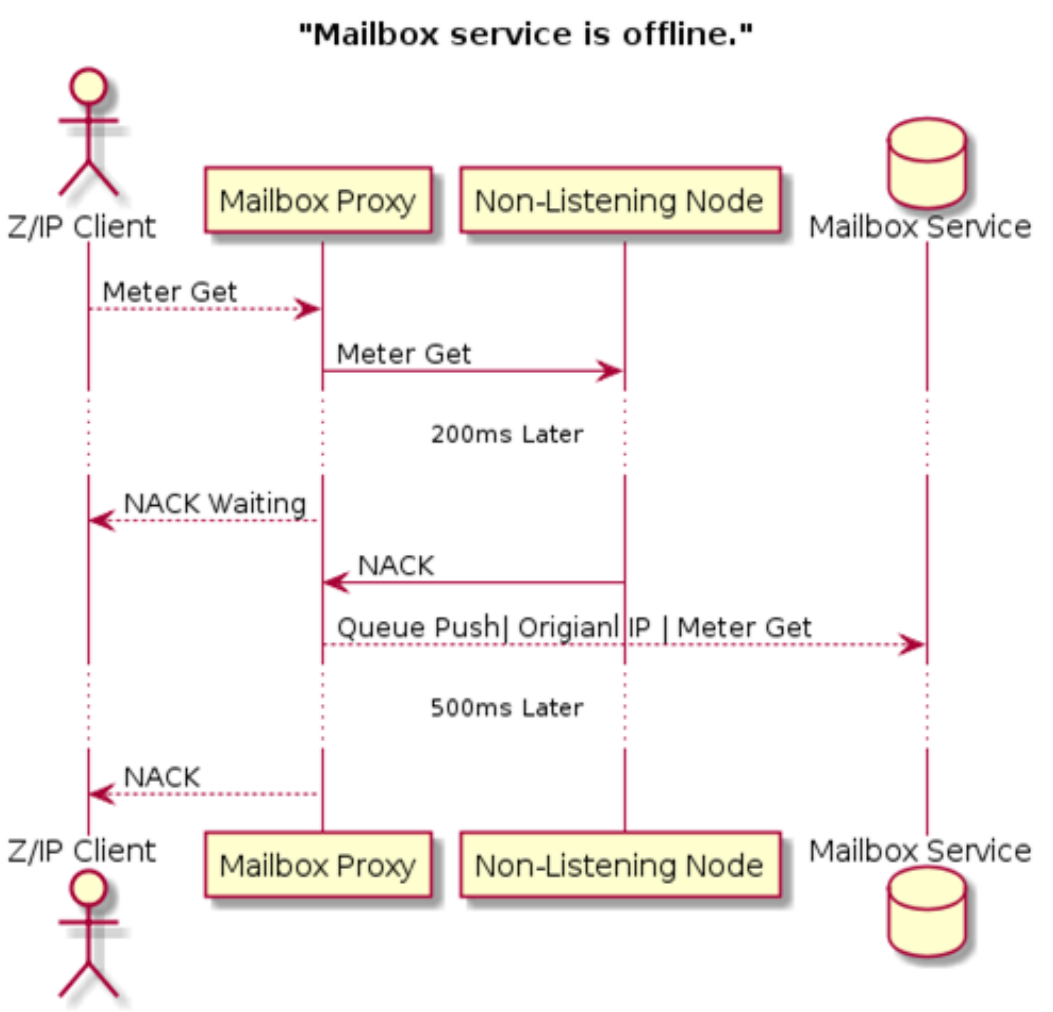


Figure 5.8: Mailbox Service is offline

5.2.4 Mailbox Command Class, version 2

The Mailbox Command Class, version 2 introduces a better handling of Wake Up periods for Wake Up nodes.

Z/IP Clients are partly responsible for issuing controlling commands to supporting nodes and conduct the minimum required interview for each Command Class of a supporting node. A Z/IP Gateway supporting the Mailbox Command Class, version 2 indicates to the Z/IP Client that the Wake Up Command Class minimum interview will be fully conducted by the Z/IP Gateway and the Z/IP Client MUST NOT send any Wake Up Command Class commands, when the mailbox service is enabled.

With the Mailbox service enabled:

- If either the Z/IP Client or Gateway supports Mailbox Command Class, version 1, the Z/IP Client MUST issue a Wake Up No More Information Command when it has completed its interview of a sleeping node.
- If both the Z/IP Client or Gateway support Mailbox Command Class, version 2 or newer, the Z/IP Client MUST NOT issue a Wake Up No More Information Command when it has completed its interview of a sleeping node.

5.2.4.1 Examples and frame flows

5.2.4.1.1 Node interview process

The node interview process is shown in Figure 5.9 and Figure 5.10. The requirement applies as soon as a client has received a Node Add Status (ADD_NODE_STATUS_DONE) command, regardless of who initiated the node inclusion.

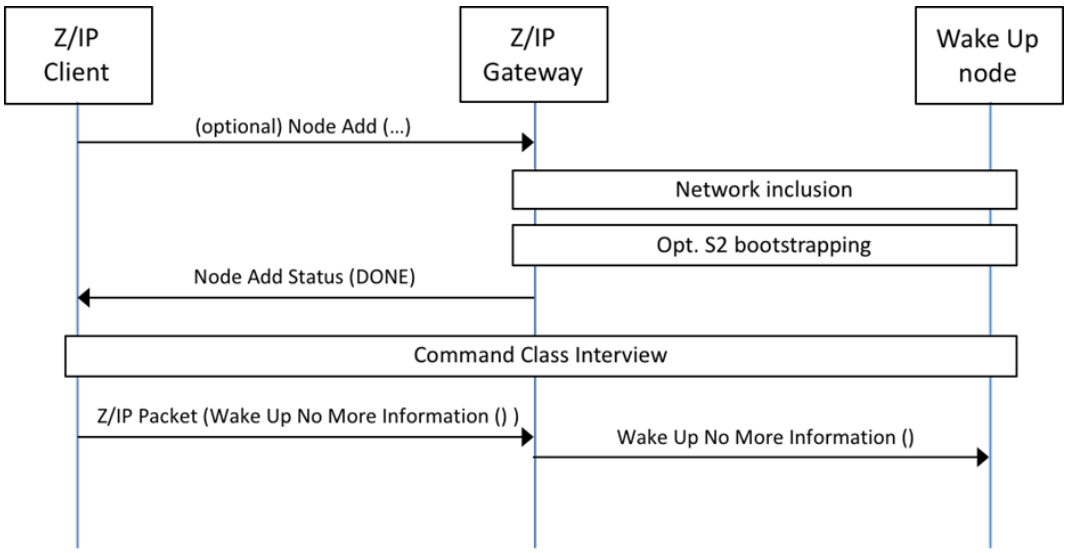


Figure 5.9: Node interview with Mailbox v1

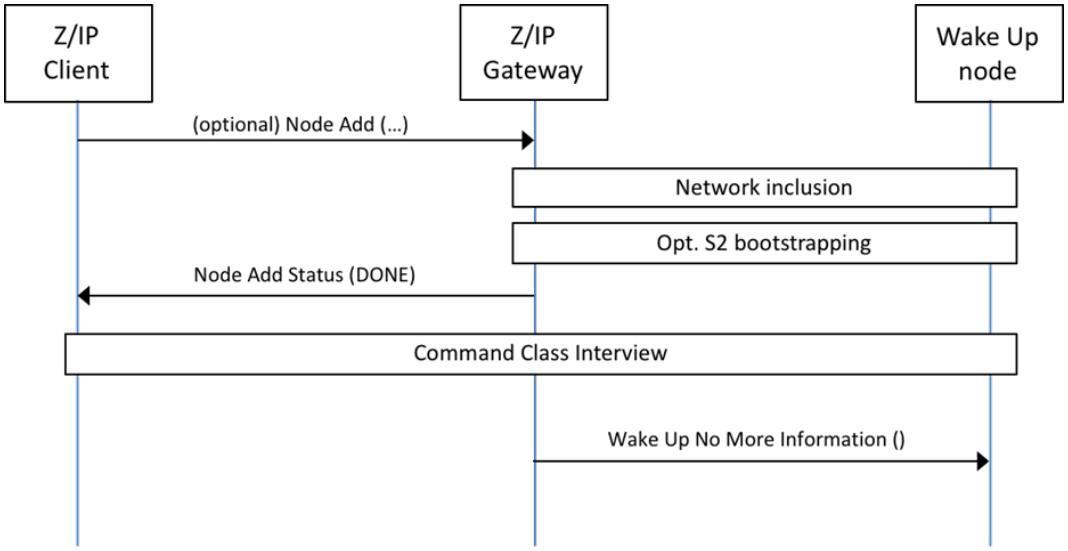


Figure 5.10: Node interview with Mailbox v1

5.2.5 Network Management Command Class, version 1

5.2.5.1 Compatibility considerations

The commands defined in the following sections may span more than the available payload length
CC:0000.00.00.21.001 in Z-Wave frames. If the command payload does not fit in a single frame, commands MUST be fragmented using the Transport Service Command Class.

CC:0000.00.00.21.002 When using IP transport, the IP UDP data segment length limit of 1280 bytes MUST be respected.

There is a risk that a controlling node would try to issue Network Management commands to a controller which does not support functionality due to its Network role (i.e. Secondary controller). A
CC:0000.00.00.22.001 controller SHOULD adjust its NIF (or S0/S2 Commands Supported Report Command) based on its network role after inclusion.

CC:0000.00.00.21.004 When a node has the SIS, Primary controller or Inclusion controller role, it MUST support:

- Network Management Inclusion Command Class
- Network Management Basic Command Class
- Transport Service Command Class

CC:0000.00.00.21.005 When a node has the secondary controller role, it MUST support:

- Network Management Basic Command Class

The Z-Wave Network Management commands are organized as follows

Table 5.19: Z-Wave Network Management Commands

Command Class	Purpose
Network Management Proxy	The command class is used to report the list of nodes present in a Z-Wave Network and report the secure/non-secure capabilities of each of those nodes Version 2 of this command class extends the node capability reporting to Multi Channel End Points.
Network Management Basic Node	The command class is used to remotely control network management operations related to including supporting nodes into a Z-Wave network. The available functionalities are : • Enable Learn mode • Request a node to broadcast its Node Information Frame • Request a node to request a network topology update to the SUC • Reset a controller to the factory default state Version 2 of this command class extends the learn mode activation commands In order to support S2 and adds the following functionality: • Request a node to report its S2 DSK.

Table 5.20: Z-Wave Network Management Commands 2

Command Class	Purpose
Network Management Inclusion	This command class is used to remotely control network management operations related to including other nodes into a Z-Wave network. The available functionalities are : • Enable Add mode • Remove a node from the network • Remove a Failed NodeID from the network • Replace a Failed NodeID in the network • Request the node to ask a specific node to perform a Neighbor update. • Instruct the supporting node to assign a return route to another end node node • Instruct the supporting node to remove return routes in another end node node Version 2 of this command class extends the Add/Remove/Replace commands to support S2 and adds the following functionality: • A supporting node can be instructed which S2 keys to grant to a joining node. • A supporting node can be provided a DSK input for S2 authentication.
Network Management Primary	This command class is used to remotely trigger a controller change operation.
Network Management Installation and maintenance	This command class is used for maintenance and optimization purposes. The available functionalities are : • Manipulate priority routes (working routes) • Request network statistics recorded by the node.

5.2.5.1.1 Sequence number management

The following text applies to all sequence numbers used by Network Management Command Classes.

CC:0000.00.00.21.006 Each sequence number MUST be generated from an 8-bit counter that is incremented by 1 whenever a new sequence number is generated. When a node powers up, the sequence counter MUST be initialized to a random value.

CC:0000.00.00.23.001 All command classes referring to this section MAY use the same global counter.

CC:0000.00.00.21.007 When responding to a request command, a responding node MUST echo the sequence number used by the requesting node.

CC:0000.00.00.21.008 When receiving response to a request command, the requesting node MUST verify that the response carries the same sequence number as the request command.

5.2.5.2 Scope of network management

Network management commands may be used in a number of scenarios. Three scopes have been identified:

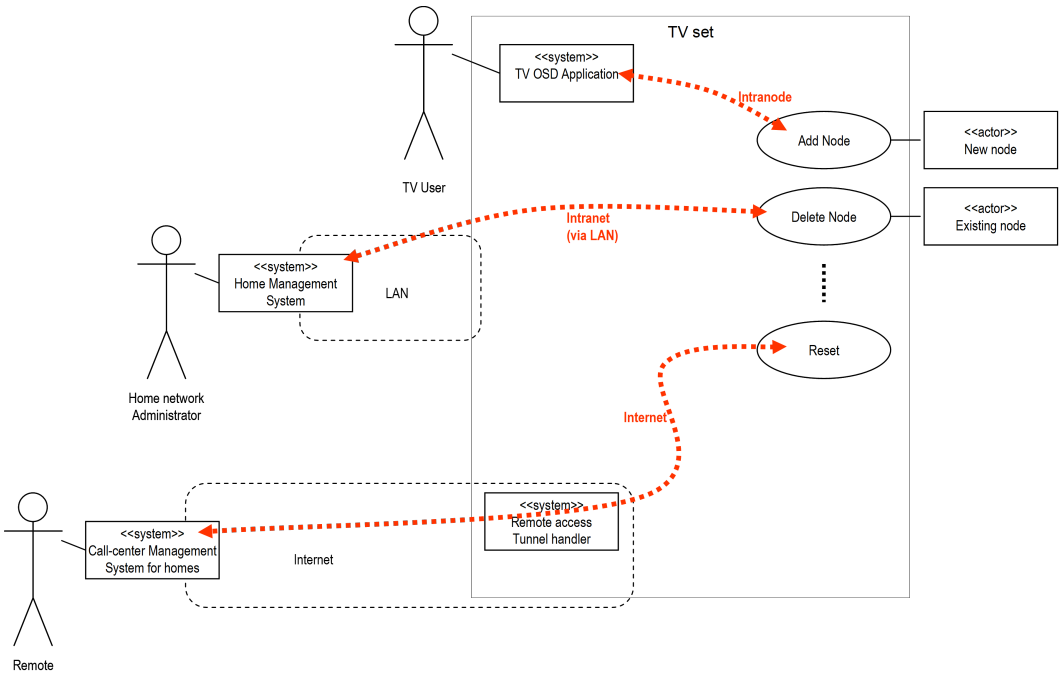


Figure 5.11: Scope of network management

5.2.5.2.1 Intranode

When used in an intranode configuration, the network management command classes are primarily used for implementation convenience. As an example, a software module of the Z/IP Gateway application may be used to provide a standard IP-based interface for other Linux applications inside a set-top box. In this way an application programmer does not have to bother about serial port communication, Telnet command parsing, etc.

5.2.5.2.2 Intranet (LAN)

Managed building automation systems may implement one central network manager controlling a number of geographically distributed Z/IP Gateways via the network management command classes. Each Z/IP Gateway may be instructed to perform local inclusion or exclusion of nodes; thus creating a large infrastructure segmented into subnets.

5.2.5.2.3 Internet (WAN)

The help desk of a service provider may provide support from a remote call center via the Internet. This enables the deployment of border routers, remote controls and plug-in modules in consumer environments without relying on the technical interest and/or capabilities of the user.

5.2.5.3 Security considerations

CC:0000.00.00.42.001 Network management is a powerful toolbox. From an application level, it SHOULD be ensured that the user does not unintentionally reset the controller or remove nodes.

CC:0000.00.00.41.001 At the same time it MUST be ensured that it is not possible for unauthorized persons to inject malicious commands into the network, e.g. resetting the primary controller to default factory settings.

CC:0000.00.00.41.002 All Network Management Command Class MUST be sent securely when used on a Z-Wave network, using at least Z-Wave Security 0 Command Class, version 1. When used on the LAN side other means of security should be used.

If the network management commands are carried in IP packets over Z-Wave, a minimum level of security is automatically applied since S0 network security is mandatory for all Z/IP traffic.

CC:0000.00.00.42.002 When Z-Wave network management commands are carried over IP LAN and WAN media (intranet & internet) the IP traffic SHOULD be using secure communication. A Z/IP Gateway MAY allow a LAN-based IP host to send un encrypted Network Management commands to a controller via the Z/IP Gateway. Support for un encrypted Network Management commands SHOULD be disabled by default and after a factory reset.

5.2.5.3.1 Designing for single-threading and limited transmit buffer

In order to support constrained CPU platforms, the Z-Wave API has been designed for single-threaded
CC:0000.00.00.41.003 operation. A node MUST ignore Network Management command if already processing or executing another Network Management command.

CC:0000.00.00.42.003 A node SHOULD NOT ignore the command if it is identical to the command currently being processed/executed (e.g. Add Node Command with mode: Stop when Add Mode is active)

CC:0000.00.00.41.004 A node MUST return status messages to the node that actually initiated the operation.

CC:0000.00.00.41.005 An controlling node MUST time out waiting for a status message. The time out SHOULD depend on
CC:0000.00.00.42.004 the actual command. If not receiving a status message within the defined time out for a command, the node SHOULD re-send the Network Management command using the same sequence number to allow the target node to detect duplicates.

CC:0000.00.00.43.001 A receiving node MAY return a ”busy” indication. Doing so could however lead to transmit buffer overflows. Care should be taken to avoid this during implementation.

The Z-Wave Ack does not necessarily indicate that the command is being executed, but that it has
CC:0000.00.00.41.006 been received by the protocol. The sending application MUST wait for the Network Management command callback, or time out.

5.2.5.4 Network management proxy command class, version 1

The Network Management Proxy Command Class provides functions to access basic network information such as the list of nodes currently present in the Z-Wave network.

5.2.5.4.1 Node list get command

This command is used to request the network node list from local storage in a node.

CC:0052.01.01.11.001 The Node List Report Command MUST be returned in response to this command.

CC:0052.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:0052.01.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.21: Node List Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NODE_LIST_GET (0x01)							
Seq No							

Seq No (1 byte)

Refer to *Sequence number management*.

5.2.5.4.2 Node list report command

This command carries node data requested with the Node List Get Command.

CC:0052.01.02.11.001 CC:0052.01.02.12.001 In addition, when a node has been added to or removed from the network or when the Z/IP Gateway has acquired the SIS role, the Z/IP Gateway MUST send an unsolicited Node List Report with the new network information to the unsolicited destination.

CC:0052.01.02.11.002 If the unsolicited destination itself has initiated the node addition or removal, this command SHOULD NOT be sent.

CC:0052.01.02.11.004 The Z/IP Gateway MAY send an unsolicited Node List Report when it is ready after power reset. If no unsolicited destination has been set, the gateway MUST NOT send a Node List Report upon network changes.

Table 5.22: Node List Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NODE_LIST_REPORT (0x02)							
Seq No							
Status							
Node List Controller ID							
Node List Data 1							
...							
Node List Data 29							

Seq No 1byte)

Refer to *Sequence number management*.

Status 8 bits)

CC:0052.01.02.11.003 This field indicates the status of Node List data carried in the command. The field MUST take one of the following values:

- 0x00: The Node List Data contains the latest updated node list.
- 0x01: The Node List Data may be outdated.

Node List Controller ID (1 byte)

The Node List Controller ID is a NodeID pointing at a controller, which keeps latest updated node list. The value 0x00 indicates that Node List Controller ID is unknown.

CC:0052.01.02.12.002 The Node List Controller SHOULD provide up-to-date information, but the actual freshness of data depends on the network construction. If a portable controller is primary there may be no access to the most recent network data. In that case the user may have to manually wake up the portable controller and initiate a controller replication to an always listening secondary controller.

No explicit Z-Wave route is provided for reaching the Node List Controller. The requesting node may use methods such as explorer discovery or Controller Network Update if the node does not already hold a working route to the indicated Node List Controller.

The Node List Controller ID may not support Network Management Proxy Command Class.

Node List Data (29 bytes)

This field carries a complete bitmap presenting all included nodes as a set bit ('1') while unused NodeIDs are presented as a ('0'). The first bit in the bitmap represents NodeID 1; the last bit represents NodeID 232.

A receiving node can use the Node Info Cached Get Command to get information on individual node properties.

5.2.5.4.3 Node info cached get command

This command is used to request node capabilities that have been cached by another node. The command works as a proxy function provided by the node list controller. The purpose is to preserve the bandwidth of the Z-Wave network and to provide access to properties of sleeping nodes.

CC:0052.01.03.11.001 The Node Info Cached Report Command MUST be returned in response to this command.

CC:0052.01.03.13.001 A Z/IP client MAY issue the Node Info Cached Get command as an IPv4 broadcast or an IPv6 'all
CC:0052.01.03.11.002 routers' multicast packet. A Z/IP Gateway MUST accept such a packet and return a Node Info Cached Report in response.

CC:0052.01.03.11.003 A Node Info Cached Report returned by a Z/IP Gateway in response to an IP multicast packet MUST be delayed by a random delay in the range 0..450msec as more than one Z/IP Gateway may be responding.

CC:0052.01.03.11.004 The Z/IP Gateway MUST respond to an IP multicast by returning a unicast IP packet.

CC:0052.01.03.13.002 A Z/IP Client MAY time out waiting for Node Info Cached Report commands after 500msec.

Table 5.23: Node Info Cached Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NODE_INFO_CACHED_GET (0x03)							
Seq No							
Reserved				Max Age			
NodeID							

Seq No (1 byte)

Refer to *Sequence number management*.

Reserved

CC:0052.01.03.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Max Age (4 bits)

The maximum age of the Node Info frame, given in 2^n minutes. If the cache entry does not exist
CC:0052.01.03.12.001 or if it is older that the value given in this field, the Z/IP Gateway SHOULD attempt to get a fresh Node Info Frame before responding to this command.

A value of 15 means infinite, i.e. No Cache Refresh. A value of 0 means force update. The values 1..15 allow for cache timeouts in the range 2min, 4min, ..., 11days – and infinite.

NodeID (1 byte)

CC:0052.01.03.11.006 This field MUST indicate the NodeID for which the receiving node is to return cached data.

The value 0x00 MUST be interpreted as the ID of the queried network management node.

5.2.5.4.4 Node info cached report command

This command is used for returning cached node information.

Table 5.24: Node Info Cached Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NODE_INFO_CACHED_REPORT (0x04)							
Seq No							
Status				Age			
List.	Z-Wave Protocol Specific Part						
Opt. Func.	Z-Wave Protocol Specific Part						
Reserved							
Basic Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							

*) Command classes may be extended –> spanning two bytes for one command class

Seq No (1 byte)

Refer to *Sequence number management*.

Status (4 bits)

This field is used to indicate the Node Info Cached information status. This field MUST comply with
CC:0052.01.04.11.001 Table 5.25.

Table 5.25: Node Info Cached Report::Status parameter encoding

Value	Status identifier	Description
0x00	STATUS_OK	The requested NodeID could be found and up-to-date information is returned.
0x01	STATUS_NOT_RESPONDING	The requested NodeID could be found but fresh information could not be retrieved.
0x02	STATUS_UNKNOWN	The NodeID is unknown.

Age (4 bits)

This field indicates the age of the Node Info frame, i.e. the time elapsed since the data has been
CC:0052.01.04.11.002 received by the actual node. This field MUST be expressed in "2^n minutes". This field's value MUST be rounded down, i.e.12 minutes MUST be reported as 2^3 = 8 minutes and not as 2^4 = 16 min.

List (1 bit)

The Optional Functionality bit indicates if true (== '1') the node supports more command classes in addition to the ones covered by the device classes listed in this message. The additional command classes follow the device class fields.

Reserved

CC:0052.01.04.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Z-Wave Protocol Specific Part

CC:0052.01.04.11.004 This field is the protocol specific part of the NIF. It MUST be set as received in the Node Information Frame.

Basic Device Class (1 byte)

This field indicates the Basic Device Class of the actual node.

Generic Device Class (1 byte)

This field indicates the Generic Device Class of the actual node. The Generic Device Classes for Z-Wave Plus are listed in [34] and Section 7.

Specific Device Class (1 byte)

This field indicates the Specific Device Class of the actual node. The Specific Device Classes for Z-Wave Plus are listed in [34] and Section 7.

Command Class (N bytes)

This field indicates the command classes implemented by the actual node.

CC:0052.01.04.11.005 The Security Scheme 0 Mark MUST be used to delimit Command Classes available non-securely and securely.

CC:0052.01.04.11.006 The Support/Control Mark MUST be used before and after the Security Scheme 0 Mark if it was present in the node's NIF.

CC:0052.01.04.11.007 A Command Class field structure example is shown in Table 5.26. The field MUST comply with Table 5.27.

Table 5.26: Command Class field structure example

Description	Command Class field content							
...	7	6	5	4	3	2	1	0
Non-secure Supported Command Classes	Command Class 1 *)							
Non-secure Supported Command Classes	...							
Non-secure Supported Command Classes	Command Class M *)							
Support/Control Mark	0xEF							
Non-secure Controlled Command Classes	Command Class 1 *)							
Non-secure Controlled Command Classes	...							
Non-secure Controlled Command Classes	Command Class K *)							
Security Scheme 0 Mark	0xF1							
Security Scheme 0 Mark	0x00							
S0 Secure Supported Command Classes	Command Class 1 *)							
S0 Secure Supported Command Classes	...							
S0 Secure Supported Command Classes	Command Class L *)							
Support/Control Mark	0xEF							
S0 Secure Controlled Command Classes	Command Class 1 *)							
S0 Secure Controlled Command Classes	...							
S0 Secure Controlled Command Classes	Command Class P *)							

*) Command classes may be extended –> spanning two bytes for one command class

Table 5.27: Special Command Class identifiers

Command Class ID	Description
0x20..0xEE	Command Class identifier
0xF101..0xFFFF	Extended Command Classes identifier
0xEF	Command Class Support/Control Mark Anything between this mark and the next mark is Controlled and not supported
0xF100	Security Scheme 0 Command Class Mark. Command Classes following this Mark are supported or controlled with Security Scheme 0

5.2.5.5 Network management proxy command class, version 2

5.2.5.5.1 Compatibility considerations

CC:0052.02.00.21.001 The Network Management Proxy Command Class, version 2 is backwards compatible with Network Management Proxy Command Class, version 1. A node supporting Network Management Proxy Command Class, version 2 MUST also support Network Management Proxy Command Class, version 1.

All commands not mentioned in this version remain unchanged from version 1.

The following command has been extended to support S2 bootstrapping information:

- Node Info Cached Report

The following commands have been added to support Multi Channel End Point probing:

- Network Management Multi Channel End Point Get Command
- Network Management Multi Channel End Point Report Command
- Network Management Multi Channel Capability Get Command
- Network Management Multi Channel Capability Report Command
- Network Management Multi Channel Aggregated Members Get Command
- Network Management Multi Channel Aggregated Members Report Command

5.2.5.5.2 Node info cached report command

This command is used for returning cached node information.

Table 5.28: Node Info Cached Report Command v2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NODE_INFO_CACHED_REPORT (0x04)							
Seq No							
Status				Age			
List.	Z-Wave Protocol Specific Part						
Opt. Func.	Z-Wave Protocol Specific Part						
Granted Keys							
Basic Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							

*) Command classes may be extended –> spanning two bytes for one command class

Fields not described in this version remain unchanged from version 1.

Granted Keys (8 bits)

This field is used to indicate which network keys were granted during bootstrapping. This field MUST
CC:0052.02.04.11.001 be treated as a bitmask and comply with Table 5.85

Command Class (N bytes)

Refer to Section 5.2.5.4.4 and Table 5.27.

CC:0052.02.04.11.002 The Security Command Class Mark (0xF100) MUST indicate command classes supported using the highest listed Security Key in the Granted Key field value.

5.2.5.5.3 Network management multi channel end point get command

This command is used to query the number of Multi Channel End Points and other relevant Multi Channel attributes.

CC:0052.02.05.11.001 The Network Management Multi Channel End Point Report Command MUST be returned in response to this command unless it is to be ignored.

Table 5.29: Network Management Multi Channel End Point Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_END_POINT_GET (0x05)							
Seq No							
NodeID							

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0052.02.05.11.002 This field MUST indicate the NodeID for which the receiving node is to return cached data. If the specified NodeID does not exist, this command MUST be ignored.

5.2.5.5.4 Network management multi channel end point report command

This command is used to advertise the number of Multi Channel End Points implemented by a node.

Table 5.30: Network Management Multi Channel End Point Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_END_POINT_REPORT (0x06)							
Seq No							
NodeID							
Reserved							
Res	Individual End Points						
Res	Aggregated End Points						

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0052.02.06.11.001 This field MUST indicate the NodeID for which the receiving node is to return cached data.

Reserved / Res

CC:0052.02.06.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Individual End Points (7 bits)

CC:0052.02.06.11.003 This field MUST advertise the number of individual End Points implemented by this node.

CC:0052.02.06.11.004 The value MUST be in the range 0..127. The sum of the values advertised by the Individual End Points and Aggregated End Points fields MUST be in the range 0..127.

Aggregated End Points (7 bits)

CC:0052.02.06.11.005 This field MUST advertise the number of Aggregated End Points implemented by this node.

CC:0052.02.06.11.006 The value MUST be in the range 0..127. The sum of the values advertised by the Individual End Points and Aggregated End Points fields MUST be in the range 0..127.

CC:0052.02.06.11.007 If no Aggregated End Points are implemented, this field MUST advertise the value 0 (zero).

5.2.5.5.5 Network management multi channel capability get command

This command is used to query the capabilities of one individual End Point or Aggregated End Point.

CC:0052.02.07.11.001 The Network Management Multi Channel Capability Report Command MUST be returned in response to this command unless it is to be ignored.

Table 5.31: Network Management Multi Channel Capability Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_CAPABILITY_GET (0x07)							
Seq No							
NodeID							
Res	End Point						

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0052.02.07.11.002 This field MUST indicate the NodeID for which the receiving node is to return cached data.

Res

CC:0052.02.07.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Point (7 bits)

CC:0052.02.07.13.001 This field MAY specify a valid End Point as advertised by the Multi Channel End Point Report. If the specified End Point does not exist, this command MUST be ignored.

5.2.5.5.6 Network management multi channel capability report command

This command is used to advertise the generic and specific device class and the supported command classes of one End Point.

Table 5.32: Network Management Multi Channel Capability Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_CAPABILITY_REPORT (0x08)							
Seq No							
NodeID							
Command Class Length							
Res	End Point						
Generic Device Class							
Specific Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							

*) Command classes may be extended –> spanning two bytes for one command class

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0052.02.08.11.001 This field MUST indicate the NodeID for which the receiving node is to return cached data.

Res (1 bit)

CC:0052.02.08.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Command Class Length (1 byte)

CC:0052.02.08.11.003 This field MUST advertise the length in bytes of the Command Class field.

End Point (7 bits)

CC:0052.02.08.11.004 This field MUST advertise a valid End Point as advertised by the Multi Channel End Point Report.

Generic Device class (8 bits)

This field indicates the Generic Device Class of the advertised End Point.

Specific Device class (8 bits)

This field indicates the Specific Device Class of the advertised End Point.

Command Class (N bytes)

CC:0052.02.08.11.005 This field MUST advertise Command Classes supported or controlled by the End Point in question.

Refer to Section 5.2.5.4.4 and Table 5.27.

CC:0052.02.08.11.006 The Security Command Class Mark (0xF100) MUST indicate command classes supported using the highest listed Security Key in the Granted Key field value.

5.2.5.5.7 Network management multi channel aggregated members get command

This command is used to query the members of an Aggregated End Point.

CC:0052.02.09.11.001 The Network Management Multi Channel Aggregated Members Report Command MUST be returned in response to this command unless it is to be ignored.

Table 5.33: Network Management Multi Channel Aggregated Members Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_AGGREGATED_MEMBERS_GET (0x09)							
Seq No							
NodeID							
Res	Aggregated End Point						

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0052.02.09.11.002 This field MUST indicate the NodeID for which the receiving node is to return cached data. This command MUST be ignored if the NodeID field is not valid.

Res

CC:0052.02.09.11.003 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Aggregated End Point (7 bits)

CC:0052.02.09.11.004 This field MUST specify an Aggregated End Point. This command MUST be ignored if the End Point does not exist or is not an Aggregated End Point.

5.2.5.5.8 Network management multi channel aggregated members report command

This command is used to advertise the members of an Aggregated End Point.

Table 5.34: Network Management Multi Channel Aggregated Members Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_AGGREGATED_MEMBERS_REPORT (0x0A)							
Seq No							
NodeID							
Res	Aggregated End Point						
Number of Members							
Res	Member Endpoint 1						
...							
Res	Member Endpoint N						

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0052.02.0A.11.001 This field MUST indicate the NodeID for which the receiving node is to return cached data.

Res

CC:0052.02.0A.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Aggregated End Point (7 bits)

CC:0052.02.0A.11.003 This field MUST advertise an Aggregated End Point.

CC:0052.02.0A.11.004 If the command is returned in response to a Multi Channel Aggregated Members Get, this field MUST advertise the same value as was received in the Multi Channel Aggregated Members Get command.

Number of Members (8 bits)

CC:0052.02.0A.11.005 This field MUST advertise the number of members of the aggregated End Points

Member Endpoint (N * 7 bits)

CC:0052.02.0A.11.006 This list is used to advertise the End Point members of the Aggregated End Point advertised in the Aggregated End Point field. The length of the list MUST be determined from the Number of Members field. This field MUST be omitted if the Number of Members field is set to 0.

CC:0052.02.0A.11.007 Each object in the list is a 7-bit End Point ID. The addressing bit (Res) MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

5.2.5.6 Network management proxy command class, version 3

5.2.5.6.1 Compatibility considerations

The Network Management Proxy Command Class, version 3 is backwards compatible with Network Management Proxy Command Class, version 2.

CC:0052.03.00.21.001 All commands and fields not mentioned in this version MUST remain unchanged from version 1.

The Command Class is extended with these 2 commands:

- Failed Node List Get Command
- Failed Node List Report Command

The strategy for considering that nodes are failing (or non-responsive/unlikely to respond to frames again) is implementation specific and may differ from one supporting node to another. A controlling node SHOULD allow the Replace Failed Node and Remove Failed node network management functions for nodes reported as failing.

5.2.5.6.2 Failed node list get command

This command is used to request the network node list that is marked as failing (or non-responsive).

CC:0052.03.0B.11.001 The Failed Node List Report Command MUST be returned in response to this command.

CC:0052.03.0B.11.002 CC:0052.03.0B.11.003 This command MUST NOT be issued via multicast addressing. A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.35: Failed Node List Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = FAILED_NODE_LIST_GET (0x0B)							
Seq No							

Seq No (1 byte)

Refer to *Sequence number management*.

5.2.5.6.3 Failed node list report command

This command is used to advertise the current list of failing nodes in the network.

Table 5.36: Failed Node List Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = FAILED_NODE_LIST_REPORT (0x0C)							
Seq No							
Failed Node List Data 1							
...							
Failed Node List Data 29							

Seq No (1 byte)

Refer to *Sequence number management*.

Failed Node List Data (29 bytes)

This field carries a complete bitmask representation of nodes ranging from NodeID 1 to NodeID 232.

CC:0052.03.0C.11.001 Bit 0 in byte 1 MUST represent NodeID 1

Bit 1 in byte 1 MUST represent NodeID 2

...

Bit 7 in byte 29 MUST represent NodeID 232

CC:0052.03.0C.11.002 The value 0 MUST indicate that the NodeID is either not part of the network or part of the network and fully functional.

The value 1 MUST indicate that the NodeID is part of the network and is failing (or not responding to frames)

5.2.5.7 Network management proxy command class, version 4

5.2.5.7.1 Compatibility Considerations

The Network Management Proxy Command Class, version 4 is backwards compatible with Network Management Proxy Command Class, version 3.

All commands and fields not mentioned in this version MUST remain unchanged from version 3.

This version of the Network Management Proxy Command Class introduces support for the Z-Wave Long Range protocol. The following commands are updated:

- Node List Report Command
- Node Info Cached Get Command
- Network Management Multi Channel End Point Get Command
- Network Management Multi Channel End Point Report Command
- Network Management Multi Channel Capability Get Command
- Network Management Multi Channel Capability Report
- Network Management Multi Channel Aggregated Members Get Command
- Network Management Multi Channel Aggregated Members Report Command
- Failed Node List Report Command

5.2.5.7.2 Node list report command

This command is used to advertise the list of nodes in the Z-Wave / Z-Wave Long Range network.

A Z/IP Gateway MAY send an unsolicited Node List Report when it is ready after power reset. If no unsolicited destination has been set, the gateway MUST NOT send a Node List Report upon network changes.

Table 5.37: Node List Report Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NODE_LIST_REPORT (0x02)							
Seq No							
Status							
Node List Controller ID							
Node List Data 1							
...							
Node List Data 29							
Extended Node List Length (MSB)							
Extended Node List Length (LSB)							
Extended Node List 1							
...							
Extended Node List N							

All fields not described below MUST remain unchanged from version 3.

Extended Node List Length (2 bytes)

This field is used to advertise the length in byte of the Extended Node List. A sending node SHOULD set this field to the smallest value allowing to advertise all NodeIDs present in the current network.

Extended Node List (N bytes)

This field is used to advertise the list of nodes included in the network with a NodeID greater than 255.

The length of this field (in bytes) MUST be according to the Extended Node List Length field.

This field MUST be treated as a bitmask and encoded as follow.

- Bit 0 in byte 1 MUST represent NodeID 256 (0x100)
- Bit 1 in byte 1 MUST represent NodeID 257 (0x101)
- etc.

The value 0 MUST indicate that no node has the corresponding NodeID assigned in the network. The value 1 MUST indicate there is a node with the corresponding NodeID assigned present in the network.

5.2.5.7.3 Node info cached get command

This command is used to request the capabilities of a node present in the network.

The Node Info Cached Report Command MUST be returned in response to this command.

A Z/IP client MAY issue the Node Info Cached Get command as an IPv4 broadcast or an IPv6 'all routers' multicast packet. A Z/IP Gateway MUST accept such a packet and return a Node Info Cached Report in response.

A Node Info Cached Report returned by a Z/IP Gateway in response to an IP multicast packet MUST be delayed by a random delay in the range 0..450msec as more than one Z/IP Gateway may be responding.

The Z/IP Gateway MUST respond to an IP multicast by returning a unicast IP packet.

A Z/IP Client MAY time out waiting for Node Info Cached Report commands after 500msec.

Table 5.38: Node Info Cached Get Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NODE_INFO_CACHED_GET (0x03)							
Seq No							
Reserved				Max Age			
NodeID							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

All fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field MUST indicate the NodeID for which the receiving node is to return cached data. The value 0x00 MUST be interpreted as the ID of the queried network management node.

The value 0xFF MUST indicate that the queried NodeID is indicated in the Extended NodeID field.

Extended NodeID (2 bytes)

This field is used to indicate the NodeID for which the Node Information is requested

This field MUST be set to the same value as the NodeID field by a sending node and ignored by a receiving node if the NodeID field value is in the range 0x00..0xFE

This field MUST be used in place of the NodeID if the NodeID field is set to 0xFF.

5.2.5.7.4 Network management multi channel end point get command

This command is used to query the number of Multi Channel End Points and other relevant Multi Channel attributes.

The Network Management Multi Channel End Point Report Command MUST be returned in response to this command unless it is to be ignored.

Table 5.39: Network Management Multi Channel End Point Get Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_END_POINT_GET (0x05)							
Seq No							
NodeID							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

All fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field MUST indicate the NodeID for which the number of endpoints is requested. If the specified NodeID does not exist, this command MUST be ignored.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the Extended NodeID field.

Extended NodeID (2 bytes)

This field is used to indicate the requested NodeID.

If the NodeID field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the NodeID field is set to 0xFF, this field MUST indicate the NodeID for which the number of endpoints MUST be returned.

If the specified NodeID does not exist, this command MUST be ignored.

5.2.5.7.5 Network management multi channel end point report command

This command is used to advertise the number of Multi Channel End Points implemented by a node.

Table 5.40: Network Management Multi Channel End Point Report Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_END_POINT_REPORT (0x06)							
Seq No							
NodeID							
Reserved							
Res	Individual End Points						
Res	Aggregated End Points						
Extended NodeID (MSB)							
Extended NodeID (LSB)							

Fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field MUST indicate the NodeID for which the number of endpoints is advertised.

If the specified NodeID does not exist, this command MUST be ignored.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the Extended NodeID field.

Extended NodeID (2 bytes)

This field is used to indicate the advertised NodeID.

If the NodeID field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the NodeID field is set to 0xFF, this field MUST indicate the NodeID for which the number of endpoints is advertised.

5.2.5.7.6 Network management multi channel capability get command

This command is used to query the capabilities of one individual End Point or Aggregated End Point.

The Network Management Multi Channel Capability Report Command MUST be returned in response to this command unless it is to be ignored.

Table 5.41: Network Management Multi Channel Capability Get Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_CAPABILITY_GET (0x07)							
Seq No							
NodeID							
Reserved							
Res	End Point						
Extended NodeID (MSB)							
Extended NodeID (LSB)							

Fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field MUST indicate the NodeID for which the capabilities are requested.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the Extended NodeID field.

Extended NodeID (2 bytes)

This field is used to indicate the requested NodeID.

If the NodeID field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the NodeID field is set to 0xFF, this field MUST indicate the NodeID for which the number of endpoints is advertised.

If the specified NodeID does not exist, this command MUST be ignored.

5.2.5.7.7 Network management multi channel capability report command

This command is used to advertise the generic and specific device class and the supported command classes of one End Point.

Table 5.42: Network Management Multi Channel Capability Report Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_CAPABILITY_REPORT (0x08)							
Seq No							
NodeID							
Command Class Length							
Res	End Point						
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

Fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field MUST indicate the NodeID for which an endpoint's capabilities are advertised.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the Extended NodeID field.

Extended NodeID (2 bytes)

This field is used to indicate the advertised NodeID for which an endpoint's capabilities are advertised.

If the NodeID field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the NodeID field is set to 0xFF, this field MUST indicate the NodeID for which an endpoint's capabilities are advertised.

5.2.5.7.8 Network management multi channel aggregated members get command

This command is used to query the members of an Aggregated End Point.

CC:0052.04.09.11.001 The Network Management Multi Channel Aggregated Members Report Command MUST be returned in response to this command unless it is to be ignored.

Table 5.43: Network Management Multi Channel Aggregated Members Get Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_AGGREGATED_MEMBERS_GET (0x09)							
Seq No							
NodeID							
Res	Aggregated End Point						
Extended NodeID (MSB)							
Extended NodeID (LSB)							

Fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field MUST indicate the NodeID for which the aggregated endpoints are requested.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the Extended NodeID field.

Extended NodeID (2 bytes)

This field is used to indicate the requested NodeID.

If the NodeID field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the NodeID field is set to 0xFF, this field MUST indicate the NodeID for which the number of endpoints is advertised.

If the specified NodeID does not exist, this command MUST be ignored.

5.2.5.7.9 Network management multi channel aggregated members report command

This command is used to advertise the members of an Aggregated End Point.

Table 5.44: Network Management Multi Channel Aggregated Members Report Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = NM_MULTI_CHANNEL_AGGREGATED_MEMBERS_REPORT (0x0A)							
Seq No							
NodeID							
Res	Aggregated End Point						
Number of Members							
Res	Member Endpoint 1						
...							
Res	Member Endpoint N						
Extended NodeID (MSB)							
Extended NodeID (LSB)							

All fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field MUST indicate the NodeID for which the aggregated endpoint members are advertised.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the Extended NodeID field.

Extended NodeID (2 bytes)

This field is used to indicate the advertised NodeID for which the aggregated endpoint members are advertised.

If the NodeID field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the NodeID field is set to 0xFF, this field MUST indicate the NodeID for which the aggregated endpoint members are advertised.

5.2.5.7.10 Failed node list report command

This command is used to advertise the current list of failing nodes in the network.

Table 5.45: Failed Node List Report Command v4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY (0x52)							
Command = FAILED_NODE_LIST_REPORT (0x0C)							
Seq No							
Failed Node List Data 1							
...							
Failed Node List Data 29							
Extended NodeID (MSB)							
Extended NodeID (LSB)							
Extended Failed Node List 1							
...							
Extended Failed Node List N							

All fields not described below MUST remain unchanged from version 3.

Extended Failed Node List Length (2 bytes)

This field is used to advertise the length in byte of the Extended Node List.

A sending node SHOULD set this field to the smallest value allowing to advertise all failed NodeIDs present in the current network.

Extended Failed Node List (N bytes)

This field is used to advertise the list of failing nodes with a NodeID greater than 255.

The length of this field (in bytes) MUST be according to the Extended Failed Node List Length field.

This field MUST be treated as a bitmask and encoded as follow:

- Bit 0 in byte 1 MUST represent NodeID 256 (0x100)
- Bit 1 in byte 1 MUST represent NodeID 257 (0x101)
- etc.

The value 0 MUST indicate that the NodeID is either not part of the network or part of the network and fully functional.

The value 1 MUST indicate that the NodeID is part of the network and is failing (or not responding to frames)

5.2.5.8 Network Management Basic Node Command Class, version 1

5.2.5.8.1 Default set command

This command is used to set the Controller back to the factory default state.

CC:004D.01.06.11.001 The Default Set Complete Command MUST be returned in response to this command. A receiving node MUST return the DEFAULT_SET_BUSY status if it is already busy executing another network management command.

CC:004D.01.06.11.002 This command MUST NOT be issued via multicast addressing.

CC:004D.01.06.11.003 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

CC:004D.01.06.12.001 This function SHOULD be used with care as it could render a network unusable if the primary controller in an existing network is set back to default. If a node is set to default while it is still a member of a network, the node will become a failing NodeID in that network.

Table 5.46: Default Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = DEFAULT_SET (0x06)							
Seq No							

Seq No (8 bits)

Refer to *Sequence number management*.

5.2.5.8.2 Default set complete command

This command is used to indicate if the Default Set operation was executed successfully or not.

Table 5.47: Default Set Complete Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = DEFAULT_SET_COMPLETE (0x07)							
Seq No							
Status							

Seq No (8 bits)

Refer to *Sequence number management*.

Status (8 bits)

CC:004D.01.07.11.001 This field indicates the status of the default set operation. This field MUST comply with Table 5.48.

Table 5.48: Default Set Complete::Status encoding

Value	Identifier	Description
0x06	DEFAULT_SET_DONE	The Default Set operation has been completed successfully.
0x07	DEFAULT_SET_BUSY	The Default Set operation has not been executed because the node is busy.

5.2.5.8.3 Learn mode set command

This command is used to allow a node to be added to (or removed from) the network. When a node is added to the network, the node is assigned a valid Home ID and NodeID.

This command allows a controlling application to request the transmission of Node Information Frames (NIFs) in regular intervals until included, removed or until learn mode is disabled again.

CC:004D.01.01.12.001 Learn mode SHOULD be enabled only when necessary, and it SHOULD always be disabled again as quickly as possible. However, to ensure a successful synchronization of the inclusion process the device SHOULD be able to stay in learn mode at least 5 seconds.

CC:004D.01.01.11.006 The Learn Mode Set Status Command MUST be returned in response to this command unless it is to be ignored.

CC:004D.01.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:004D.01.01.11.003 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.49: Learn Mode Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = LEARN_MODE_SET (0x01)							
Seq No							
Reserved							
Mode							

Seq No (8 bits)

Refer to *Sequence number management*.

Reserved

CC:004D.01.01.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Mode (8 bits)

CC:004D.01.01.11.005 The Mode field controls operation. This field MUST comply with Table 5.50.

CC:004D.01.01.13.001

Table 5.50: Learn Mode Set::Mode parameter encoding

Value	Identifier	Description
0x00	ZW_SET_LEARN_MODE_DISABLE	Stop the learn mode of the node. The command MAY be ignored if Learn Mode was not activated. The command MAY be ignored if network inclusion or security bootstrapping is ongoing.
0x01	ZW_SET_LEARN_MODE_CLASSIC	Start the learn mode on the controller and accept only being included in direct range
0x02	ZW_SET_LEARN_MODE_NWI	Start the learn mode on the controller and accept routed inclusion.

Examples of Learn Mode activation and deactivation are given in Figure 5.12

5.2.5.8.4 Learn mode in a controller

If the receiving node is a controller, it receives and stores the node list and routing table for the network during inclusion. This information transmitted as part of the controller replication. This function will most likely change the capabilities of the controller.

5.2.5.8.5 Learn mode set status command

This command is used to indicate the progress of the Learn Mode Set command.

Table 5.51: Learn Mode Set Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = LEARN_MODE_SET_STATUS (0x02)							
Seq No							
Status							
Reserved							
New NodeID							

Seq No (8 bits)

Refer to *Sequence number management*.

Status (8 bits)

CC:004D.01.02.11.001 This field indicates the outcome of the learn mode and MUST comply with Table 5.52.

Table 5.52: Learn Mode Status::Status parameter encoding

Value	Identifier	Description
0x06	LEARN_MODE_DONE	The learn process is complete and the controller is now included into (or excluded from) the network. If the node supports S0 or S2, it indicates that the network inclusion and security bootstrapping were completed successfully (This include the case where the node was granted no S2 key).
0x07	LEARN_MODE_FAILED	The learn process failed in some general way
0x09	LEARN_MODE_SECURITY_FAILED	The learn process is complete and the node was included in a network but security bootstrapping failed. The node is **not** operating securely.

Reserved

CC:004D.01.02.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

New NodeID (1 byte)

The NodeID assigned to the new node by another primary controller or inclusion controller.

If the node was removed from the network or if the Status field is different than LEARN_MODE_DONE, this field MUST be set to 0x00.

5.2.5.8.6 Node information send command

This command is used to trigger a receiving node to issue a Node Information Frame (NIF).

CC:004D.01.05.11.001 A node receiving this command MUST send a Node Information Frame to the indicated NodeID with the indicated transmission options. No status message is returned for this command.

CC:004D.01.05.13.001 A management application MAY use this message to make a node identify itself towards a Z-Wave
CC:004D.01.05.12.001 remote control during association operations. This command SHOULD NOT be used while learn
CC:004D.01.05.13.002 mode is activated. Instead, periodic Node Information Frame transmissions MAY be enabled along with learn mode; refer to Section 5.2.5.8.1.

Table 5.53: Node Information Send Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = NODE_INFORMATION_SEND (0x05)							
Seq No							
Reserved							
Destination NodeID							
tx Options							

Seq No (8 bits)

Refer to *Sequence number management*.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Destination NodeID (1 byte)

CC:004D.01.05.13.003 This field indicates the NodeID of the node that will receive the Node Information frame. The NodeID MAY be set to the broadcast NodeID to reach all nodes within direct range.

CC:004D.01.05.12.002 Acknowledgement SHOULD NOT be requested when broadcasting.

tx Options (1 byte)

This field allows a management application to specify if the Node Information frame is to be sent with
CC:004D.01.05.11.002 special properties. This field MUST be treated as a bitmask and MUST comply with Table 5.54.

Table 5.54: Node Information Send::Tx Options encoding

Value	Option flag identifier	Description
0x00	NULL	Transmit at normal power level without any transmit options.
0x01	TRANSMIT_OPTION_ACK	Request acknowledgment from destination node. Allow routing.
0x02	TRANSMIT_OPTION_LOW_POWER	Transmit at low output power level (1/3 of normal RF range)
0x10	TRANSMIT_OPTION_NO_ROUTE	Send only in direct range
0x20	TRANSMIT_OPTION_EXPLORE	Resolve new routes via explorer discovery if existing routes fail

CC:004D.01.05.12.003 It is RECOMMENDED for a sending node to use the TRANSMIT_OPTION_NO_ROUTE tx Option and the broadcast NodeID in this command.

5.2.5.8.7 Network update request command

This command is used to request network topology updates from the SUC/SIS node.

CC:004D.01.03.11.001 A node MUST NOT use this command if no SUC is present in the network.

The SUC can only handle one network update at a time, so care should be taken not to have multiple controllers in the network ask for updates at the same time.

This command will generate a lot of network activity that will use bandwidth and stress the SUC.
CC:004D.01.03.12.001 Therefore, network updates SHOULD be requested as seldom as possible.

CC:004D.01.03.11.002 The Network Update Request Status Command MUST be returned in response to this command.

CC:004D.01.03.11.003 This command MUST NOT be issued via multicast addressing.

CC:004D.01.03.11.004 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.55: Network Update Request Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = NETWORK_UPDATE_REQUEST (0x03)							
Seq No							

Seq No (8 bits)

Refer to *Sequence number management*.

5.2.5.8.8 Network update request status command

This command is used to indicate if the Network Update Request command execution has completed successfully or not.

Table 5.56: Network Update Request Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = NETWORK_UPDATE_REQUEST (0x03)							
Seq No							
Status							

Seq No (8 bits)

Refer to *Sequence number management*.

Status (1 byte)

CC:004D.01.04.11.001 This field is used to indicate the status of the Network Update process. This field MUST comply with Table 5.57.

Table 5.57: Network Update Request Status::Status parameter encoding

Value	Status identifier	Description
0x00	ZW_SUC_UPDATE_DONE	The update process succeeded
0x01	ZW_SUC_UPDATE_ABORT	The update process aborted because of an error
0x02	ZW_SUC_UPDATE_WAIT	The SUC node is busy
0x03	ZW_SUC_UPDATE_DISABLED	The SUC functionality is disabled
0x04	ZW_SUC_UPDATE_OVERFLOW	The controller requested an update after more than 64 changes have occurred in the network. The controller has to make a replication.

5.2.5.8.9 Use cases and frame flows

5.2.5.8.10 Z/IP client requesting a node to interrupt Learn Mode

The frame flow for interrupting Learn Mode is shown in Figure 5.12.

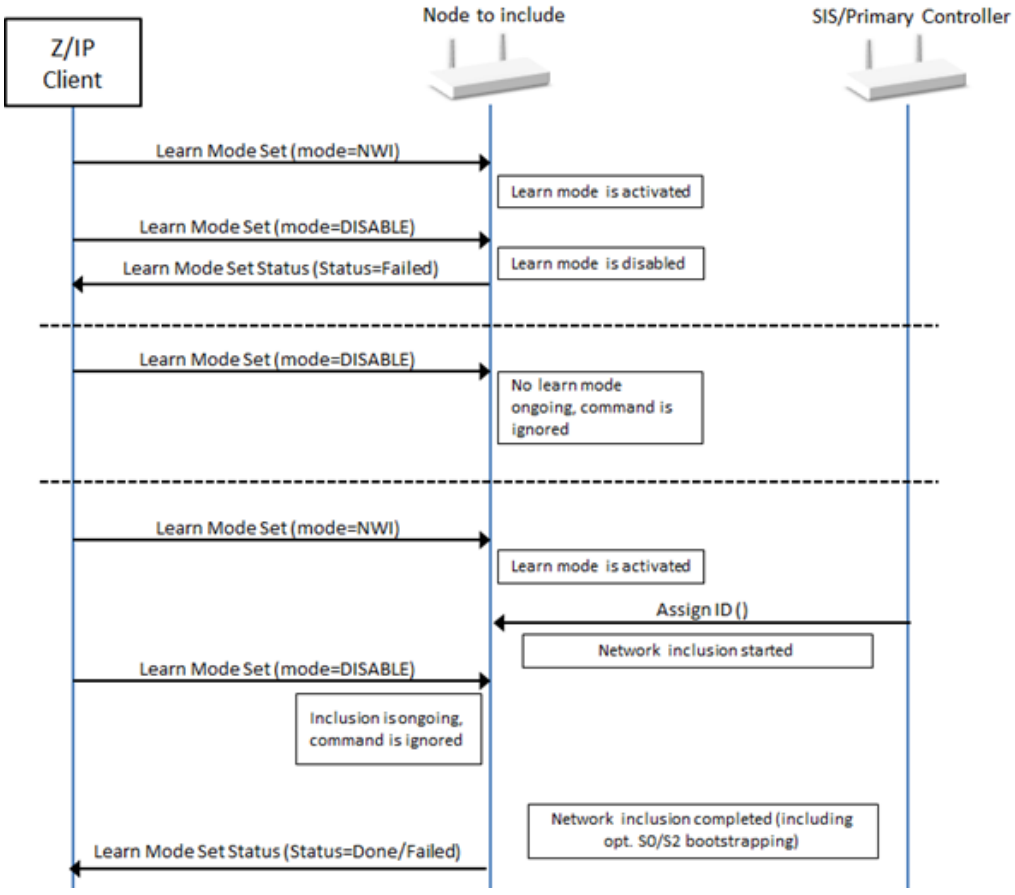


Figure 5.12: Z/IP Client interrupting learn mode

5.2.5.9 Network management basic node command class, version 2

5.2.5.9.1 Compatibility considerations

CC:004D.02.00.21.001 The Network Management Basic Command Class, version 2 is backwards compatible with Network Management Basic Command Class, version 1. A node supporting Network Management Basic Command Class, version 2 MUST also support Network Management Basic Command Class, version 1.

All commands not mentioned in this version remain unchanged from version 1.

The following commands are introduced to allow a GUI to display the DSK of a S2 node and advertise interview status:

- DSK Get Command
- DSK Report Command

The following commands have been extended to return information about the S2 bootstrapping outcome and the node interview process after activating Learn Mode:

- Learn Mode Set Command
- Learn Mode Set Status Command

5.2.5.9.2 Learn mode set command

This command is used to allow a node to be added to (or removed from) the network. When a node is added to the network, the node is assigned a valid Home ID and NodeID.

This command allows a controlling application to request the transmission of Node Information Frames (NIFs) in regular intervals until included, removed or until learn mode is disabled again.

CC:004D.02.01.12.001 Learn mode SHOULD be enabled only when necessary, and it SHOULD always be disabled again as quickly as possible. However, to ensure a successful synchronization of the inclusion process the device SHOULD be able to stay in learn mode at least 5 seconds.

CC:004D.02.01.11.001 The Learn Mode Set Status Command MUST be returned in response to this command.

CC:004D.02.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:004D.02.01.11.003 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.58: Learn Mode Set Command v2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = LEARN_MODE_SET (0x01)							
Reserved							Return interview status
Mode							

Fields not described in this version remain unchanged from version 1.

Return Interview Status (1 bit)

This field is used to request that the receiving node returns an additional Learn Mode Set Status Command when the node interview is completed.

CC:004D.02.01.11.004 The value 0 MUST indicate that the receiving node MUST return a Learn Mode Set Status Command when the learn mode is over.

The value 1 MUST indicate that the receiving node MUST return a Learn Mode Set Status Command when learn mode is over and an additional Learn Mode Set Status Command with status set to LEARN_MODE_INTERVIEW_COMPLETED when the inclusion node interview is over.

An illustration is given in Figure 5.13.

5.2.5.9.3 Learn mode set status command

This command is used to indicate the progress of the Learn Mode Set command.

Table 5.59: Learn Mode Set Status Command v2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = LEARN_MODE_SET_STATUS (0x02)							
Seq No							
Status							
Reserved							
New Node ID							
Granted Keys							
KEX Fail Type							
DSK 1							
...							
DSK 16							

Fields not described in this version remain unchanged from version 1.

Status (8 bits)

CC:004D.02.02.11.001 This field indicates the outcome of the learn mode and MUST comply with Table 5.60.

Table 5.60: Learn Mode Status version 2::Status parameter encoding

Value	Identifier	Description	Version
0x06	LEARN_MODE_DONE	The learn process is complete and the controller is now included into (or excluded from) the network. If the node supports S0 or S2, it indicates that the network inclusion and security bootstrapping were completed successfully (This include the case where the node was granted no S2 key).	1
0x07	LEARN_MODE_FAILED	The learn process failed in some general way	1
0x09	LEARN_MODE_SECURITY_FAILED	The learn process is complete and the node was included in a network but security bootstrapping failed. The node is **not** operating securely.	1
0x0A	LEARN_MODE_INTERVIEW_COMPLETED	This status is used to report that the post-inclusion interview is completed after network inclusion	2

Granted Keys (8 bits)

This field is used to indicate which network keys were granted during bootstrapping.

CC:004D.02.02.11.002 This field MUST be treated as a bitmask and comply with Table 5.85.

KEX Fail Type (8 bits)

This field is used to indicate which error occurred in case S2 bootstrapping was not successful.

CC:004D.02.02.11.003 This field MUST comply with Table 5.86.

DSK (16 bytes)

This field is used to indicate the DSK of the including controller that performed S2 bootstrapping to the node.

This information can be used for post inclusion verification.

5.2.5.9.4 DSK get command

This command is used to request the S2 DSK of a node.

CC:004D.02.08.11.001 The DSK Report Command MUST be returned in response to this command.

CC:004D.02.08.11.002 This command MUST NOT be issued via multicast addressing.

CC:004D.02.08.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.61: DSK Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = DSK_GET (0x08)							
Seq No							
Reserved							Add Mode

Seq No (8 bits)

Refer to *Sequence number management*.

Add mode (1 bit)

This field is used to request the Add Mode or Learn Mode DSK.

S2 Controllers may have 2 key pairs, one static key pair used for Learn mode (being included in a network) and one dynamic key pair changing at each bootstrapping used for Add mode (including other nodes in the network).

CC:004D.02.08.11.004 The value 0 MUST indicate that the node MUST return its Learn Mode DSK

The value 1 MUST indicate that the node MUST return its Add Mode DSK:

CC:004D.02.08.11.005 A node not supporting an Add Mode dynamic key pair MUST return its Learn Mode DSK.

5.2.5.9.5 DSK Report Command

This command is used by a node to advertise its DSK.

Table 5.62: DSK Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC (0x4D)							
Command = DSK_REPORT (0x09)							
Seq No							
Reserved							Add Mode
DSK 1							
...							
DSK 16							

Seq No (8 bits)

Refer to *Sequence number management*.

Add mode (1 bit)

This field is used to indicate if the Add Mode or Learn Mode DSK is advertised in this command.

CC:004D.02.09.11.001 The value 0 MUST indicate that the node advertises its Learn Mode DSK

The value 1 MUST indicate that the node advertises its Add Mode DSK.

DSK (16 bytes)

This field is used to transmit the S2 DSK. For details, refer to Section 4.

5.2.5.9.6 Use cases and frame flows

5.2.5.9.7 Z/IP Client requesting a node to report node interview status

The frame flow for returning status messages at the end of Learn mode and the end of the post-inclusion device interview is shown in Figure 5.13.

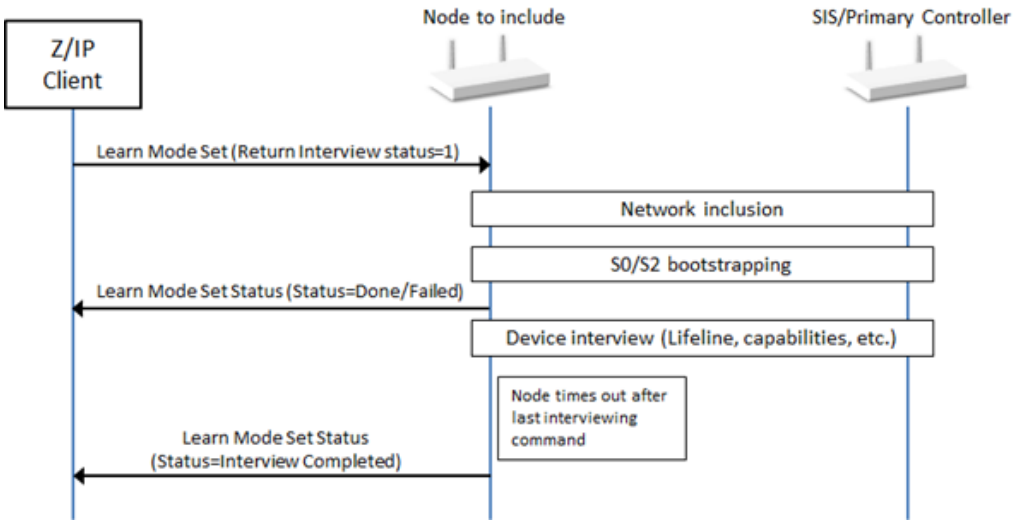


Figure 5.13: Node advertising the end of the interview process

5.2.5.10 Network management inclusion command class, version 1

The Network Management Inclusion Command Class provides functionality only available in a primary controller, inclusion controller or SIS. Since this is a dynamic property, there is a risk that a remote host tries to use commands in a controller which has become secondary in the meantime.

5.2.5.10.1 Node add command

This command is used to activate or de-activate add mode on a controller.

The process of adding a node is started by the network management application sending a Node Add command to a controller. The network management application receives a status message later
CC:0034.01.01.13.001 on indicating if the inclusion attempt was successful or not. If NWI inclusion was used, the calling application MAY re-issue this command if more nodes are to be included.

CC:0034.01.01.12.001 The Add Mode SHOULD be disabled after a certain time to avoid adding another node unexpectedly. It is RECOMMENDED to have a timer that disables the Node Add state after a given time without any activity.

CC:0034.01.01.11.001 Add Mode MUST be de-activated after any inclusion attempt, even if interrupted.

CC:0034.01.01.11.008

The Node Add Status Command MUST be returned in response to this command unless it is to be ignored.

CC:0034.01.01.11.003 This command MUST NOT be issued via multicast addressing.

CC:0034.01.01.11.004 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.63: Node Add Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD (0x01)							
Seq No							
Reserved							
Mode							
tx Options							

Seq No (8 bits)

Refer to *Sequence number management*.

Reserved

CC:0034.01.01.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

This field is use to indicate to the receiving node if the Add Mode must be activated or de-activated.

CC:0034.01.01.11.006 This field MUST comply with Table 5.64.

Table 5.64: Node Add::Mode parameter encoding

Value	Identifier	Description
0x01	ADD_NODE_ANY	Add any type of node to the network.
0x05	ADD_NODE_STOP	Stop Add Mode. The command MAY be ignored if Add Mode was not activated. The command MAY be ignored if network inclusion or security bootstrapping is ongoing.

Examples of Add Mode activation and deactivation are given in Figure 5.14.

tx Options (1 byte)

CC:0034.01.01.11.007 The tx Options field allows a controlling node to specify if transmissions MUST use special properties. This field MUST be treated as a bitmask and MUST comply with Table 5.65.

Table 5.65: Node Add::Tx Options encoding

Value	Option flag identifier	Description
0x00	ADD_NODE_ANY	Transmit at normal power level without any transmit options.
0x02	TRANSMIT_OPTION_LOW_POWER	Transmit at low output power level (1/3 of normal RF range)
0x20	TRANSMIT_OPTION_EXPLORE	Allow network-wide inclusion

CC:0034.01.01.12.002 If the Mode is set to NODE_ADD_ANY, it is RECOMMENDED to set this field to TRANSMIT_OPTION_EXPLORE.

CC:0034.01.01.13.002 Installer scenarios with a requirement for more confidential transfer of network security keys MAY set the flag TRANSMIT_OPTION_LOW_POWER. This requires that the new node is included in direct range of the including controller.

5.2.5.10.2 Node add status command

This command is used to report the result of the Node Add Command or report that a new node was included.

Table 5.66: Node Add Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD_STATUS (0x02)							
Seq No							
Status							
Reserved							
New NodeID							
Node Info Length							
List.	Z-Wave Protocol Specific Part						
Opt. Func.	Z-Wave Protocol Specific Part						
Basic Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							

*) Command classes may be extended –> spanning two bytes for one command class

Seq No (1 byte)

Refer to *Sequence number management*.

Status (1 byte)

CC:0034.01.02.11.001 This field indicates the outcome of the add mode and MUST comply with Table 5.67.

Table 5.67: Node Add Status::Status parameter encoding

Value	Status identifier	Description
0x06	ADD_NODE_STATUS_DONE	The new node has been included in the network. If the new node and controller support S0 or S2, it indicates that the network inclusion and security bootstrapping were completed successfully (This include the case where the node was granted no S2 key).
0x07	ADD_NODE_STATUS_FAILED	Transmit at low output power level (1/3 of normal RF range)
0x09	ADD_NODE_STATUS_SECURITY_FAILED	Allow network-wide inclusion

Reserved

CC:0034.01.02.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

New NodeID (1 byte)

CC:0034.01.02.11.005 This field MUST indicate the assigned NodeID to the newly added node. This field is valid if Status is different than NODE_ADD_STATUS_FAILED.

This field MUST be set to 0x00 if no NodeID was assigned to the included node.

Node Info Length (1 byte)

CC:0034.01.02.11.004

This field is used to indicate the length in bytes of the encapsulated Node Information fields. This field MUST be included in the length calculation. The value MUST indicate the length of the following fields:

- Node Info Length (this field)
- List / Z-Wave Protocol Specific Part
- Opt Func / Z-Wave Protocol Specific Part
- Basic Device Class
- Generic Device Class
- Specific Device Class
- Command Class

List. (1 bit)

Refer to *Node info cached report command*

Opt. Func. (1 bit)

Refer to *Node info cached report command*

Z-Wave Protocol Specific Part

Refer to *Node info cached report command*

Basic Device Class (1 byte)

Refer to *Node info cached report command*

Generic Device Class (1 byte)

Refer to *Node info cached report command*

Specific Device Class (1 byte)

Refer to *Node info cached report command*

Command Class (N bytes)

Refer to *Node info cached report command*

5.2.5.10.3 Node remove command

This command is used to activate or de-activate node remove mode. The remove operation only works in direct range between the controller and the node that is to be removed.

CC:0034.01.03.12.001 The Node Remove mode SHOULD be disabled after a certain time to avoid removing another node unexpectedly. It is RECOMMENDED to have a timer that disables the Node Remove mode after a given time without any activity.

CC:0034.01.03.11.001 Node Remove mode MUST be de-activated after any removal attempt, even if interrupted.

CC:0034.01.03.11.007 The Node Remove Status Command MUST be returned in response to this command unless it is ignored.

CC:0034.01.03.11.003 This command MUST NOT be issued via multicast addressing.

CC:0034.01.03.11.004 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.68: Node Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_REMOVE (0x03)							
Seq No							
Reserved							
Mode							

Seq No (1 byte)

Refer to *Sequence number management*.

Reserved

CC:0034.01.03.11.005 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Mode (1 byte)

CC:0034.01.03.11.006 This field is use to indicate to the receiving node if the node removal process must be activated or de-activated. This field MUST comply with Table 5.69.

Table 5.69: Node Remove::Mode parameter encoding

Value	Mode identifier	Description
0x01	REMOVE_NODE_ANY	Remove any type of node from the network
0x05	REMOVE_NODE_STOP	Stop the node removal process. The command MAY be ignored if the remove process was not activated. The command MAY be ignored if network exclusion is ongoing.

CC:0034.01.03.13.001

The process of removing a node is started by sending this command with Mode set to REMOVE_NODE_ANY. The removal process is complete when a Node Remove Status command with status set to NODE_REMOVE_STATUS_DONE is returned.

5.2.5.10.4 Node remove status command

This command is used to advertise the status of a node removal attempt.

Table 5.70: Node Remove Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_REMOVE_STATUS (0x04)							
Seq No							
Status							
NodeID							

Seq No (1 byte)

Refer to *Sequence number management*.

Status (1 byte)

CC:0034.01.04.11.001 This field is used to advertise status of a node removal attempt. This field MUST comply with Table 5.71.

Table 5.71: Status parameter of Node Remove Status encoding

Value	Status identifier	Description
0x06	REMOVE_NODE_STATUS_DONE	The node has now been removed and the controller is ready to continue normal operation again. Removed NodeID is returned.
0x07	REMOVE_NODE_STATUS_FAILED	The remove process failed (no node was removed)

NodeID (1 byte)

This field is used to advertise the NodeID that was attempted to be removed from the network.

CC:0034.01.04.12.001 This field SHOULD be set to 0x00 if no attempt has been made.

5.2.5.10.5 Failed node remove command

This command is used to remove a non-responding node.

CC:0034.01.07.11.001 A non-responding node is put onto the failed NodeID list by a controller when detected. In case the node responds again at a later stage, it is removed from the failed NodeID list. A node MUST be on the failed NodeID list and as an extra precaution also fail to respond before it is removed. Responding nodes MUST NOT be removed.

CC:0034.01.07.11.002 The Failed Node Remove Status Command MUST be returned in response to this command when the removal attempt has been made.

CC:0034.01.07.11.003 This command MUST NOT be issued via multicast addressing.

CC:0034.01.07.11.004 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.72: Failed Node Remove Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = FAILED_NODE_REMOVE (0x07)							
Seq No							
NodeID							

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0034.01.07.11.005 This field is used to specify the NodeID of the failing node which MUST be removed.

5.2.5.10.6 Node neighbor update request command

This command is used to instruct a node with NodeID to perform a Node Neighbor Update operation in order to update the topology on the controller.

CC:0034.01.0B.11.001 The Node Neighbor Update Status Command MUST be returned in response to this command when the neighbor search is completed.

CC:0034.01.0B.11.002 This command MUST NOT be issued via multicast addressing.

CC:0034.01.0B.11.003 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.73: Node Neighbor Update Request Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_NEIGHBOR_UPDATE_REQUEST (0x0B)							
Seq No							
NodeID							

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0034.01.0B.11.004 This field is used to specify the NodeID of the failing node which MUST perform the Node Neighbor Update operation.

5.2.5.10.7 Node neighbor update status command

This command is used to report the status of a Node Neighbor Update operation.

Table 5.74: Node Neighbor Update Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_NEIGHBOR_UPDATE_STATUS (0x0C)							
Seq No							
Status							

Seq No (1 byte)

Refer to *Sequence number management*.

Status (1 byte)

CC:0034.01.0C.11.001 This field is used to advertise status of the neighbor update operation. This field MUST comply with Table 5.75.

Table 5.75: Node Neighbor Update Status::Status encoding

Value	Status identifier	Description
0x22	NEIGHBOR_UPDATE_STATUS_DONE	New neighbor list received
0x23	NEIGHBOR_UPDATE_STATUS_FAIL	Getting new neighbor list failed

5.2.5.10.8 Return route assign command

This command is used to make a controller assign static return routes (up to 4) to an end node node. This allows the end nodes to communicate directly with other nodes.

Up to 5 different destinations can be allocated return routes. Attempts to assign new return routes when all 5 destinations already are allocated will be ignored.

Allocated return routes can only be cleared using the Return Route Delete Command.

The controller calculates the shortest routes from the end node (Source NodeID field) to the destination node (Destination NodeID field) and transmits the return routes to the end node (Source NodeID field).

CC:0034.01.0D.11.001

The Return Route Assign Complete Command MUST be returned in response to this command when the route assignment is completed.

CC:0034.01.0D.11.002 This command MUST NOT be issued via multicast addressing.

CC:0034.01.0D.11.003 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.76: Return Route Assign Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = RETURN_ROUTE_ASSIGN (0x0D)							
Seq No							
Source NodeID							
Destination NodeID							

Seq No (1 byte)

Refer to *Sequence number management*.

Source NodeID (1 byte)

This field is used to specify the NodeID of the node which will be assigned the return route.

Destination NodeID (1 byte)

This field is used to specify the destination NodeID for which the Source NodeID will have a route assigned.

5.2.5.10.9 Return route assign complete command

This command is used to indicate the status of a return route assignment attempt.

Table 5.77: Return Route Assign Complete Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = RETURN_ROUTE_ASSIGN_COMPLETE (0x0E)							
Seq No							
Status							

Seq No (1 byte)

Refer to *Sequence number management*.

Status (1 byte)

CC:0034.01.0E.11.001 This field is used to advertise status of the return route assignment attempt. This field MUST comply with Table 5.78.

Table 5.78: Return Route Assign Complete::Status encoding

Value	Option identifier	Description
0x00	TRANSMIT_COMPLETE_OK	Successfully transmitted
0x01	TRANSMIT_COMPLETE_NO_ACK	No acknowledgement is received before timeout from the destination node. Acknowledgement is discarded in case it is received after the time out.
0x02	TRANSMIT_COMPLETE_FAIL	Not possible to transmit data because the Z-Wave network is busy (jammed).
0x03	N/A	Reserved
0x04	TRANSMIT_COMPLETE_NOROUTE	No route found to assign to the destination. No frame was transmitted.
0x05	TRANSMIT_COMPLETE_VERIFIED	This status code is identical to 0x00. The route was successfully transmitted.

5.2.5.10.10 Return route delete command

This command is used to make a controller delete all static return routes from an end node. Allocated return routes can only be removed using this command. All return routes are cleared when using this command.

CC:0034.01.0F.12.001 After issuing this command, an application SHOULD issue Return Route Assign Commands to create return routes for all relevant associations.

CC:0034.01.0F.11.001 The Return Route Delete Complete Command MUST be returned in response to this command.

CC:0034.01.0F.11.002 This command MUST NOT be issued via multicast addressing.

CC:0034.01.0F.11.003 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.79: Return Route Delete Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = RETURN_ROUTE_DELETE (0x0F)							
Seq No							
NodeID							

Seq No (1 byte)

Refer to *Sequence number management*.

NodeID (1 byte)

CC:0034.01.0F.11.004 This field is used to specify the NodeID of which the return routes MUST be deleted.

5.2.5.10.11 Return route delete complete command

This command is used to indicate the status of a return route deletion attempt.

Table 5.80: Return Route Delete Complete Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = RETURN_ROUTE_DELETE_COMPLETE (0x10)							
Seq No							
Status							

Seq No (1 byte)

Refer to *Sequence number management*.

Status (1 byte)

CC:0034.01.10.11.001 This field is used to advertise status of the return route deletion attempt. This field MUST comply with Table 5.81.

Table 5.81: Return Route Delete Complete::Status encoding

Value	Option identifier	Description
0x00	TRANSMIT_COMPLETE_OK	Successfully transmitted
0x01	TRANSMIT_COMPLETE_NO_ACK	No acknowledgement is received before timeout from the destination node. Acknowledgement is discarded in case it is received after the time out.
0x02	TRANSMIT_COMPLETE_FAIL	Not possible to transmit data because the Z-Wave network is busy (jammed).

5.2.5.10.12 Use cases and frame flows

5.2.5.10.13 Z/IP Client requesting a node to interrupt Add Mode

The frame flow for interrupting Add Mode is shown in Figure 5.14.

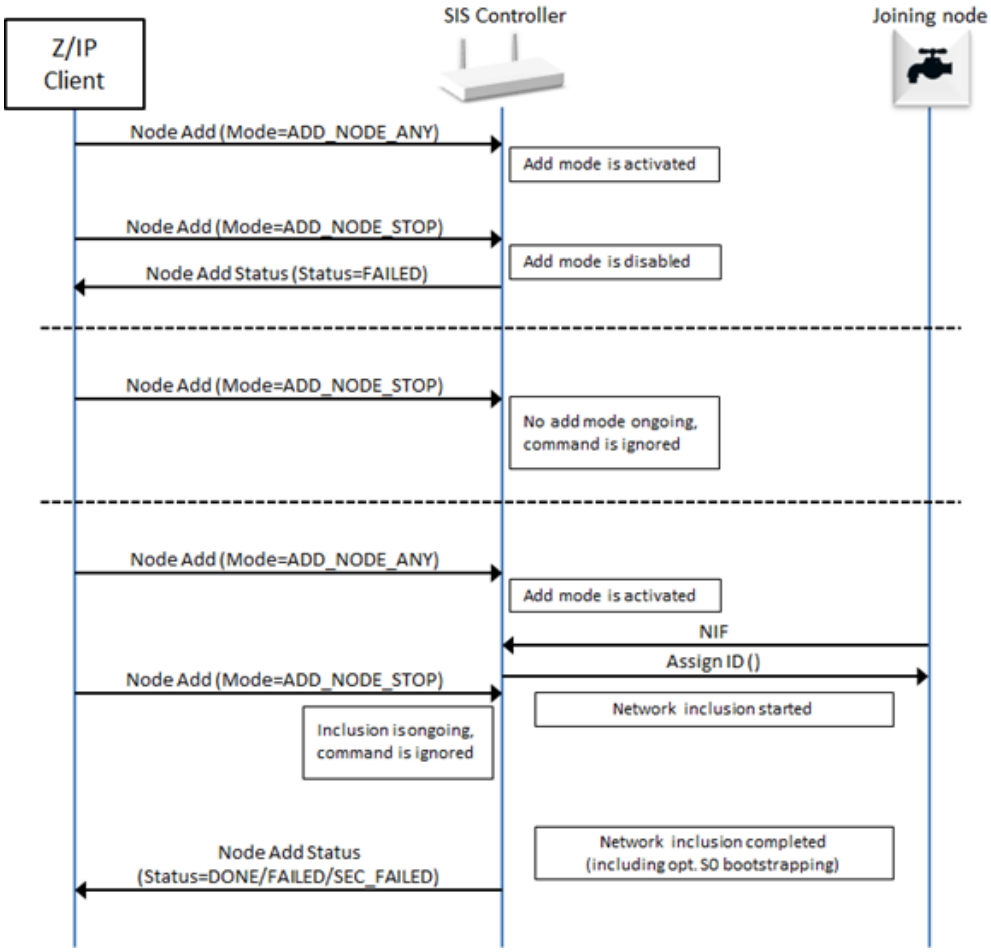


Figure 5.14: Z/IP Client requesting a node to interrupt Add Mode

5.2.5.11 Network management inclusion command class, version 2

5.2.5.11.1 Compatibility considerations

The Network Management Inclusion Command Class, version 2 is backwards compatible with Network Management Inclusion Command Class, version 1. A node supporting Network Management Inclusion
CC:0034.02.00.21.001 Command Class, version 2 MUST also support Network Management Inclusion Command Class, version 1.

All commands not mentioned in this version remain unchanged from version 1.

The following commands are introduced to support the multiple security keys and DSK functionalities of the Security 2 Command Class:

- Node Add Keys Report Command
- Node Add Keys Set Command
- Node Add DSK Report Command
- Node Add DSK Set Command

The following command has been extended to support the new S2/inclusion controller bootstrapping process:

- Node Add Command
- Node Add Status Command
- Failed Node Replace Command
- Failed Node Replace Status Command

Use-cases and frames flows for the new functionalities of this Command Class are shown in *Use cases and frame flows*.

5.2.5.11.2 Node add command

This command is used to add nodes to the Z-Wave network.

CC:0034.02.01.11.005 The Node Add Status Command MUST be returned in response to this command unless it is to be ignored.

CC:0034.02.01.11.002 This command MUST NOT be issued via multicast addressing.

CC:0034.02.01.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.82: Node Add Command V2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD (0x01)							
Seq No							
Reserved							
Mode							
tx Options							

Fields not described in this version remain unchanged from version 1.

Mode (1 byte)

This field is use to indicate to the receiving node which mode to use for the inclusion of a new node.
CC:0034.02.01.11.004 This field MUST comply with Table 5.83.

Table 5.83: Encoding of Node Add :: Mode parameter

Value	Identifier	Description	Version
0x01	NODE_ADD_ANY	Add any type of node to the network and allow Security 0 bootstrapping	1
0x05	NODE_ADD_STOP	Stop Add Mode. The command MAY be ignored if Add Mode was not activated. The command MAY be ignored if network inclusion or security bootstrapping is ongoing	1
0x07	NODE_ADD_ANY_S2	Not possible to transmit data because the Z-Wave network is busy (jammed).	2

Examples of Add Mode activation and deactivation are also given in *Z/IP Client requesting a node to interrupt Add Mode*.

5.2.5.11.3 Node add status command

This command is used to report the result of a node inclusion.

Table 5.84: Node Add Status Command V2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD_STATUS (0x02)							
Seq No							
Status							
Reserved							
New NodeID							
Node Info Length							
List.	Z-Wave Protocol Specific Part						
Opt. Func.	Z-Wave Protocol Specific Part						
Basic Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							
Granted Keys							
KEX Fail Type							

*) Command classes may be extended –> spanning two bytes for one command class

Fields not described in this version remain unchanged from version 1.

Command Class (N bytes)

Refer to *Node info cached report command* and Table 5.27.

CC:0034.02.02.11.001 The Security Command Class Mark (0xF100) MUST indicate command classes supported using the highest listed Security Key in the Granted Key field value.

Granted Keys (8 bits)

This field is used to indicate which network keys were granted during bootstrapping.

CC:0034.02.02.11.002 This field MUST be treated as a bitmask and comply with Table 5.85

Table 5.85: Node Add Status::Granted keys encoding

Bit	Description
0	Indicates the Unauthenticated Security Class Key
1	Indicates the Authenticated Security Class Key
2	Indicates the Access Control Security Class Key
7	Indicates the Security 0 Network Key

KEX Fail Type (8 bits)

This field is used to indicate which error occurred in case S2 bootstrapping was not successful.

This field MUST comply with Table 5.86.

CC:0034.02.02.11.003

Table 5.86: Node Add Status::Kex Fail Type encoding

Value	KEX Fail Type Identifier	Description
0x00	•	Bootstrapping was successful
0x01	KEX_FAIL_KEX_KEY	Key failure indicating that no match exists between requested/granted keys in the network.
0x02	KEX_FAIL_KEX_SCHEME	Scheme failure indicating that no scheme is supported by controller or joining node specified an invalid scheme.
0x03	KEX_FAIL_KEX_CURVES	Curve failure indicating that no curve is supported by controller or joining node specified an invalid curve.
0x05	KEX_FAIL_DECRYPT	Node failed to decrypt received frame.
0x06	KEX_FAIL_CANCEL	User has cancelled the S2 bootstrapping.
0x07	KEX_FAIL_AUTH	The Echo KEX Set/Report frame did not match the earlier exchanged frame.
0x08	KEX_FAIL_KEY_GET	The joining node has requested a key, which was not granted by the including node at an earlier stage.
0x09	KEX_FAIL_KEY_VERIFY	Including node failed to decrypt and hence verify the received frame encrypted with exchanged key.
0x0A	KEX_FAIL_KEY_REPORT	The including node has transmitted a frame containing a different key than what is currently being exchanged.

5.2.5.11.4 Node add keys report command

This command is used to inform which S2 keys have been requested during S2 bootstrapping.

CC:0034.02.11.11.001 The Node Add Keys Set Command MUST be returned in response to this command.

CC:0034.02.11.11.002 This command MUST NOT be issued via multicast addressing.

CC:0034.02.11.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.87: Node Add Keys Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD_KEYS_REPORT (0x11)							
Seq No							
Reserved							Request CSA
Requested Keys							

Seq No (1 byte)

Refer to *Sequence number management*.

Request CSA (1 bit)

This field is used to indicate if the joining node is requesting CSA (Client-Side Authentication, refer to Section 4).

CC:0034.02.11.11.004 The value 1 MUST indicate that the node requests CSA.

The value 0 MUST indicate that the node does not request CSA.

Requested Keys (1 bytes)

This field is used to advertise the requested keys by the joining node.

CC:0034.02.11.11.005 This field MUST be treated as a bitmask and comply with Table 5.85

CC:0034.02.12.11.004 This field MUST be set to 0x00 if the Accept field is set to 0.

Accept (1 bit)

This field is used to indicate if the S2 bootstrapping process is accepted by the user and must continue.

CC:0034.02.12.11.005 The value 0 MUST indicate that the S2 bootstrapping is not accepted and MUST be interrupted. The value 1 MUST indicate that the S2 bootstrapping is accepted and MUST continue.

5.2.5.11.5 Node add DSK report command

This command is used to report the DSK of the node being S2 bootstrapped and indicates whether an input is needed for node authentication.

CC:0034.02.13.11.001 The Node Add DSK Set Command MUST be returned in response to this command.

CC:0034.02.13.11.002 This command MUST NOT be issued via multicast addressing.

CC:0034.02.13.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.88: Node Add DSK Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD_DSK_REPORT (0x13)							
Seq No							
Reserved				Input DSK Length			
DSK 1							
...							
DSK 16							

Seq No (8 bits)

Refer to *Sequence number management*.

Input DSK Length (4 bits)

CC:0034.02.13.11.004 This field is used to indicate how many DSK bytes MUST be input as a minimum to authenticate the node being included.

CC:0034.02.13.11.005 The value 0 MUST indicate that no user input is necessary (e.g. Unauthenticated Security Class or CSA has been granted).

DSK (16 bytes)

This field is used to transmit the DSK of the node being S2 bootstrapped. Refer to Section 4.

5.2.5.11.6 Node add DSK set command

This command is used to indicate the S2 bootstrapping controller if the DSK is accepted and report the user input when needed.

Table 5.89: Node Add DSK Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD_DSK_SET (0x14)							
Seq No							
Accept	Reserved			Input DSK Length			
DSK 1							
...							
DSK N							

Seq No (1 byte)

Refer to *Sequence number management*.

Input DSK Length (4 bits)

This field indicates the length in bytes of the DSK input by the user.

CC:0034.02.14.11.001 This field MUST be set to the same or a higher value than the "Input DSK Length" field value received in the Node Add DSK Report Command that caused this command to be returned.

CC:0034.02.14.11.002 The value 0 MUST indicate that no user input has been done (e.g. Unauthenticated Security Class, CSA has been granted or user refused to input DSK).

Input DSK (N bytes)

CC:0034.02.14.11.003 This field indicates the DSK input by the user. A receiving node (Z/IP gateway) MUST overwrite the part of the DSK with the Input DSK contained in this frame

CC:0034.02.14.11.004 The length of this field in bytes MUST be according to the Input DSK Length field value. If the Input DSK Length is set to 0, this field MUST be omitted.

Accept (1 bit)

This field is used to indicate if the DSK Report is accepted by the user and if S2 bootstrapping must continue.

CC:0034.02.14.11.005 The value 0 MUST indicate that the DSK Report is not accepted and S2 bootstrapping MUST be interrupted.

The value 1 MUST indicate that the DSK Report is accepted and S2 bootstrapping MUST continue.

5.2.5.11.7 Failed node replace command

This command is used to replace a non-responding node with a new one in having the same NodeID.

CC:0034.02.09.11.005 The Failed Node Replace Status Command MUST be returned in response to this command unless it is to be ignored.

CC:0034.02.09.11.002 This command MUST NOT be issued via multicast addressing.

CC:0034.02.09.11.003 A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.90: Failed Node Replace Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = FAILED_NODE_REPLACE (0x09)							
Seq No							
NodeID							
tx Options							
...							
Mode							

Fields not described in this version remain unchanged from version 1.

Mode (1 byte)

This field indicates the type of operation for the failed replace process.

CC:0034.02.09.11.004 This field MUST comply with Table 5.91

Table 5.91: Failed Node Replace::Mode encoding

Value	Identifier	Description	Version
0x01	START_FAILED_NODE_REPLACE	Initiate a failed node replace process.	1
0x05	STOP_FAILED_NODE_REPLACE	Cancel a failed node replace process. The command MAY be ignored if no replaced failed process is active. The command MAY be ignored if network inclusion is ongoing	1
0x07	START_FAILED_NODE_REPLACE_S2	Initiate a failed node replace process and allow S2 bootstrapping for the new node	2

CC:0034.02.09.13.001

5.2.5.11.8 Failed node replace status command

This command is used to indicate the progress of the Replace Failed Node Command.

Table 5.92: Failed Node Replace Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = FAILED_NODE_REPLACE_STATUS (0x0A)							
Seq No							
Status							
Node ID							
Granted Keys							
KEX Fail Type							

Fields not described in this version remain unchanged from version 1.

Granted Keys (8 bits)

This field is used to indicate which network keys were granted during bootstrapping.

CC:0034.02.0A.11.001 This field MUST be treated as a bitmask and comply with Table 5.85

KEX Fail Type (8 bits)

Refer to *Node add status command* and Table 5.86

5.2.5.11.9 Use cases and frame flows

5.2.5.11.10 Z/IP Client with SIS or Primary controller including an S2 node

The frame flow for an S2 capable node inclusion using the Network Management Inclusion Command Class is shown in Figure 5.15.

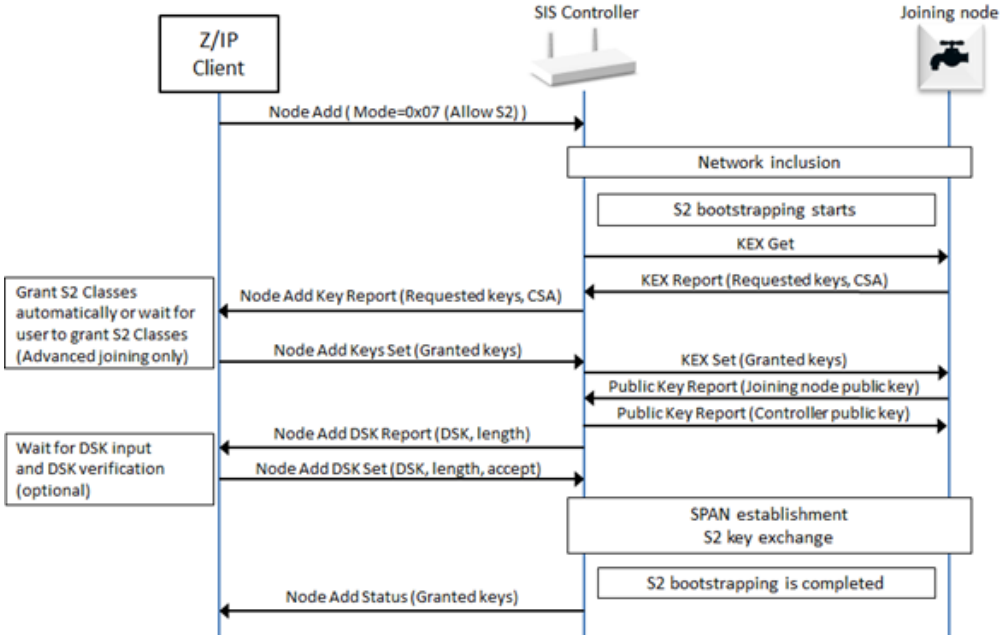


Figure 5.15: Node inclusion with a SIS/Primary controller

5.2.5.11.11 Z/IP Client with an S2 inclusion controller including an S2 node

When performing S2 bootstrapping, the unsolicited destination of the Z/IP Gateway will receive a unsolicited Node Add S2 Keys Report from the Z/IP Gateway. The Z/IP Client at this point has two options:

CC:0034.02.00.11.001

1. Automatically grant requested S2 Classes without presenting a user dialog in the S2 Keys Report step. In this case, the Z/IP Client MUST present a user dialog in next step before sending the DSK Set

CC:0034.02.00.11.002
CC:0034.02.00.13.001

2. Using advanced joining where the user MUST confirm the specific keys being requested in a dialog, before continuing to next step. In this case, the Z/IP MAY present a user dialog in next step before sending the DSK Set, if required by the S2 Classes being granted.

This is done without the SIS having entered Add Node mode. From this point on the S2 inclusion frame flow is same as when including through the SIS.

The frame flow for the node inclusion when an S2 capable inclusion controller has been used for including a new S2 capable node is shown in Figure 5.16.

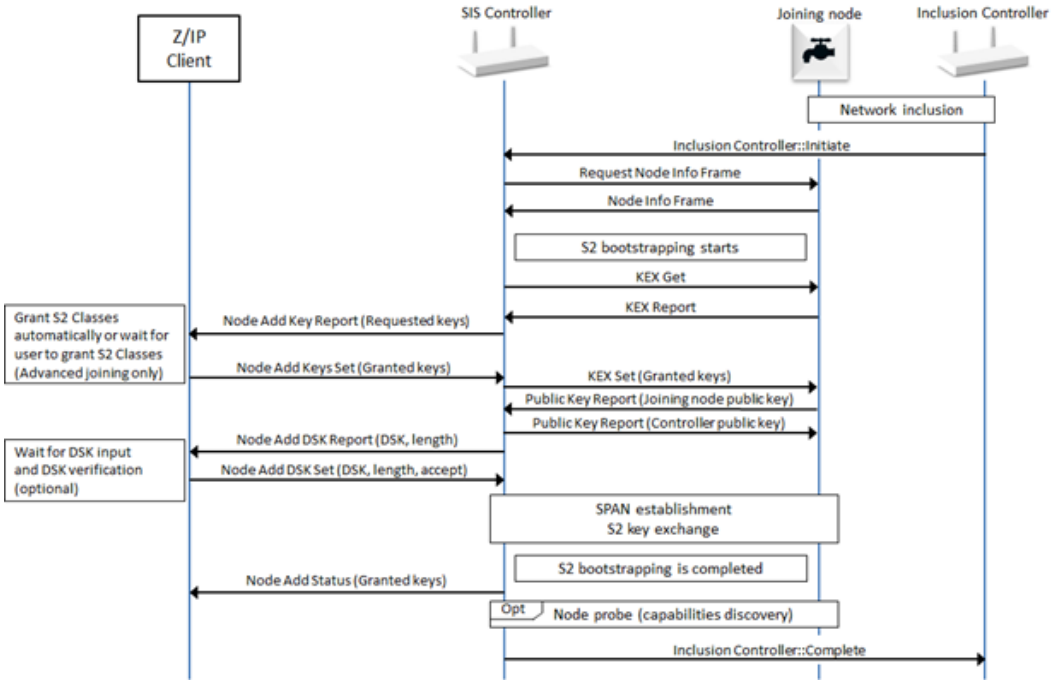


Figure 5.16: Node inclusion with an S2 inclusion controller

5.2.5.11.12 Z/IP client with an S2 inclusion controller including an S0 node

CC:0034.02.00.13.002 The unsolicited destination of the Z/IP Gateway will receive an unsolicited Node Add Status Report from the Z/IP Gateway. The unsolicited destination Z/IP Client MAY show a dialog informing that a node was included by an inclusion controller.

The frame flow for an S0 capable node inclusion using an S2 capable inclusion controller including is shown in Figure 5.17.

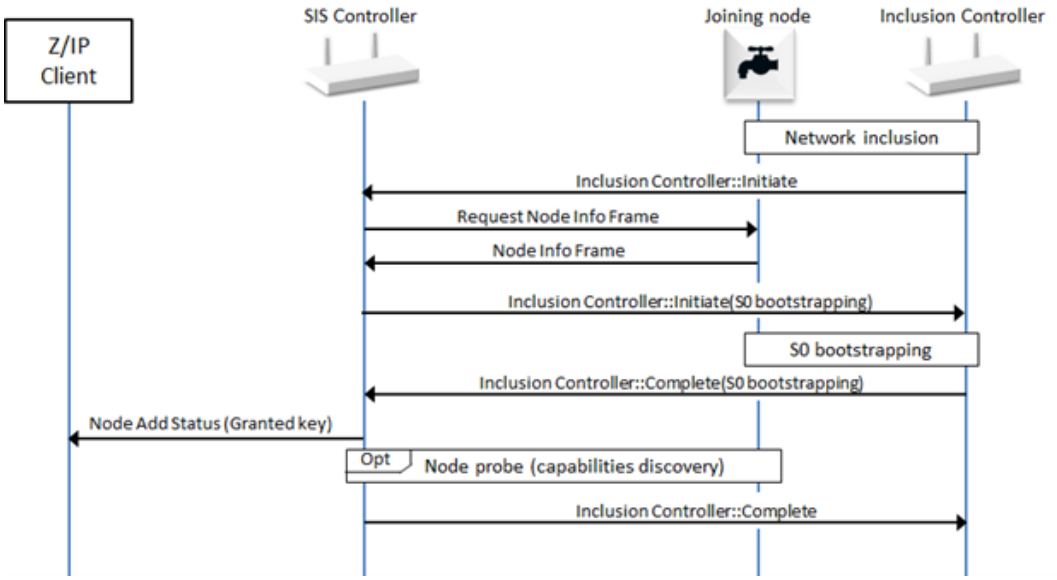


Figure 5.17: S0 node inclusion with an S2 inclusion controller

5.2.5.12 Network management inclusion command class, version 3

5.2.5.12.1 Compatibility considerations

The Network Management Inclusion Command Class, version 3 is backwards compatible with the Network Management Inclusion Command Class, version 2.

CC:0034.03.00.21.001 A node supporting the Network Management Inclusion Command Class, version 2 MUST also support the Network Management Inclusion Command Class, version 2.

CC:0034.03.00.21.002 All commands and fields not mentioned in this version MUST remain unchanged from version 2.

The following command has been extended to support the report of a Smart Start node:

- Node Add Status Command

The following commands are introduced in order to support the Smart Start functionality:

- Included Node Information Frame Report Command
- Smart Start Join Started Command

CC:0034.03.00.21.003 Frame flows for the new functionalities of this Command Class are shown in "ref"*network_protocol_command_classes-command_class_definitions-network_management_command_classes-network_management_*
A Z/IP Gateway MUST comply with *Network management inclusion command class, version 3*.

CC:0034.03.00.21.004 A supporting node MUST issue the Node Add Status Command, Included Node Information Frame Report Command and the Smart Start Join Started Command to the first and the second unsolicited destinations.

5.2.5.12.2 Command class dependencies

CC:0034.03.00.21.005 A node supporting the Network Management Inclusion Command Class, version 3 MUST also support the Node Provisioning Command Class, version 1.

5.2.5.12.3 Node add status command

This command is used to report the result of the Node Add Command or report that a new node was included.

Table 5.93: Node Add Status Command V3

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_ADD_STATUS (0x02)							
Seq No							
Status							
Reserved							
New NodeID							
Node Info Length							
List.	Z-Wave Protocol Specific Part						
Opt. Func.	Z-Wave Protocol Specific Part						
Basic Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							
Granted Keys							
KEX Fail Type							
Reserved				DSK Length			
DSK 1							
...							
DSK L							

CC:0034.03.02.11.001 Fields not described below MUST remain unchanged from version 2.

Status (8 bits)

CC:0034.03.02.11.002 This field indicates the outcome of the add mode and MUST comply with Table 5.94.

Table 5.94: Node Add Status::Status parameter encoding V3

Value	Status identifier	Description	Version
0x06	ADD_NODE_STATUS_DONE	The new node has been included in the network. If the new node and controller support S0 or S2, it indicates that the network inclusion and security bootstrapping were completed successfully. (This includes the case where the node was granted no S2 key)	1
0x07	ADD_NODE_STATUS_FAILED	The process failed, no new node was added in the network. Version 3: This status is also used if the node failed a smart start inclusion and has been removed. In this case, it may attempt the inclusion again.	1
0x09	ADD_NODE_STATUS_SECURITY_FAILED	Node has been included but the security bootstrapping failed	1

DSK Length (5 bits)

CC:0034.03.02.11.003 This field MUST indicate the length of the DSK field in bytes.

CC:0034.03.02.11.004 This field MUST be set to 0 if the added node does not support the S2 Command Class.

CC:0034.03.02.11.005 This field MUST be set to 16 if the added node supports the S2 Command Class.

DSK (L bytes)

CC:0034.03.02.11.006

This field MUST advertise the DSK of the node that has been added to the network.

CC:0034.03.02.11.007 The length of this field (in bytes) MUST be according to the DSK Length field value. This field MUST be omitted if the DSK Length field is set to 0.

5.2.5.12.4 Included node information frame report command

CC:0034.03.19.11.006 This command MUST be sent to the (first and second) unsolicited destinations when an Included NIF (INIF) is received and the following conditions are fulfilled:

- The advertised NHID matches an entry in the provisioning list
- The advertised HomeID is different than the current network HomeID.

CC:0034.03.19.11.007 A node issuing this command MUST subsequently issue a Node Provisioning Report Command for the matched entry in the provisioning list.

With the two commands, a Z/IP client can use the relevant information to guide the user on how to perform a reset operation on the device.

Table 5.95: Included Node Information Frame Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = INCLUDED_NIF_REPORT (0x19)							
Seq No							
Reserved			DSK Length				
DSK 1							
...							
DSK N							

Seq No (1 byte)

Refer to *Sequence number management*.

Reserved

CC:0034.03.19.11.002 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

DSK Length (5 bits)

CC:0034.03.19.11.008 This field MUST indicate the length of the DSK field in bytes.

CC:0034.03.19.11.009 This field MUST be set to 16.

DSK (N bytes)

CC:0034.03.19.11.00A This field MUST advertise the DSK of the provisioning list entry that has been matched from the NHID in the received INIF.

CC:0034.03.19.11.00B The length of this field (in bytes) MUST be according to the DSK Length field value.

5.2.5.12.5 Smart start join started command

CC:0034.03.15.11.001 This command MUST be sent to the (first and second) unsolicited destinations when a Smart Start inclusion starts.

CC:0034.03.15.11.002 The Add Node Status Command MUST be issued after the Smart Start inclusion and S2 bootstrapping attempts took place.

Table 5.96: Smart Start Join Started Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = SMART_START_JOIN_STARTED_REPORT (0x15)							
Seq No							
Reserved				DSK Length			
DSK 1							
...							
DSK N							

Seq No (1 byte)

Refer to *Sequence number management*.

DSK Length (5 bits)

CC:0034.03.15.11.003 This field MUST indicate the length of the DSK field in bytes.

CC:0034.03.15.11.004 This field MUST be set to 16.

DSK (N bytes)

This field is used to advertise the DSK for the Provisioning List entry which starts the Smart Start inclusion process.

CC:0034.03.15.11.005 The length of this field (in bytes) MUST be according to the DSK Length field value.

5.2.5.12.6 Usage and frame flows

5.2.5.12.7 Z/IP Gateway adding a smart start node that is on the provisioning list

The frame flow for a Smart Start inclusion of a node previously added on the provisioning list is shown in Figure 5.18.

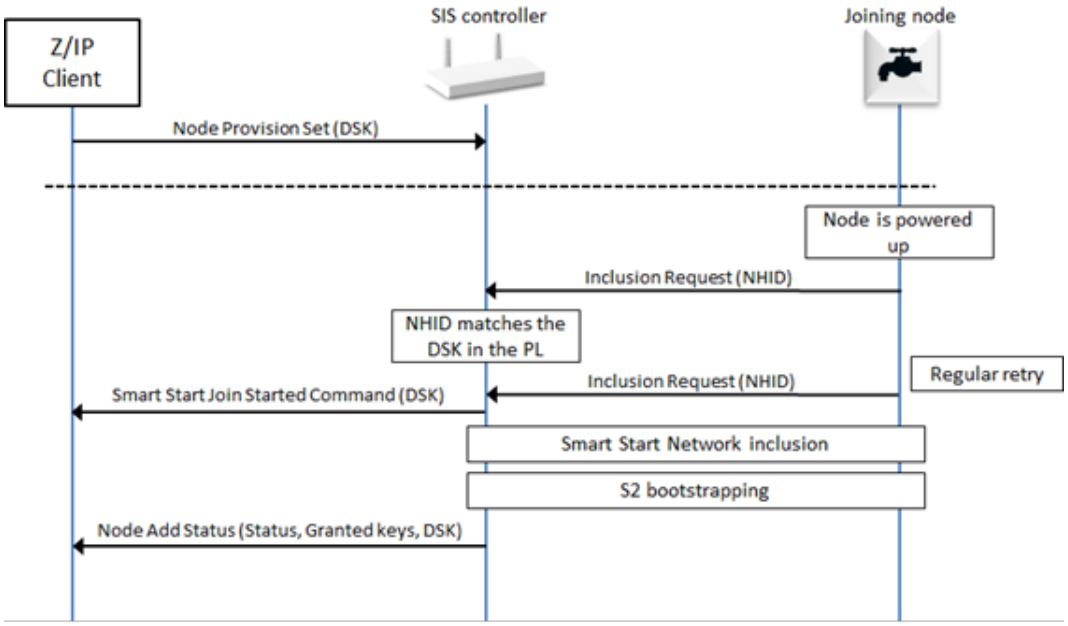


Figure 5.18: Smart Start inclusion

5.2.5.12.8 Z/IP Gateway adding a smart start node that is subsequently added on the provisioning list

The frame flow for a Smart Start inclusion of a node subsequently added on the provisioning list is shown in Figure 5.19.

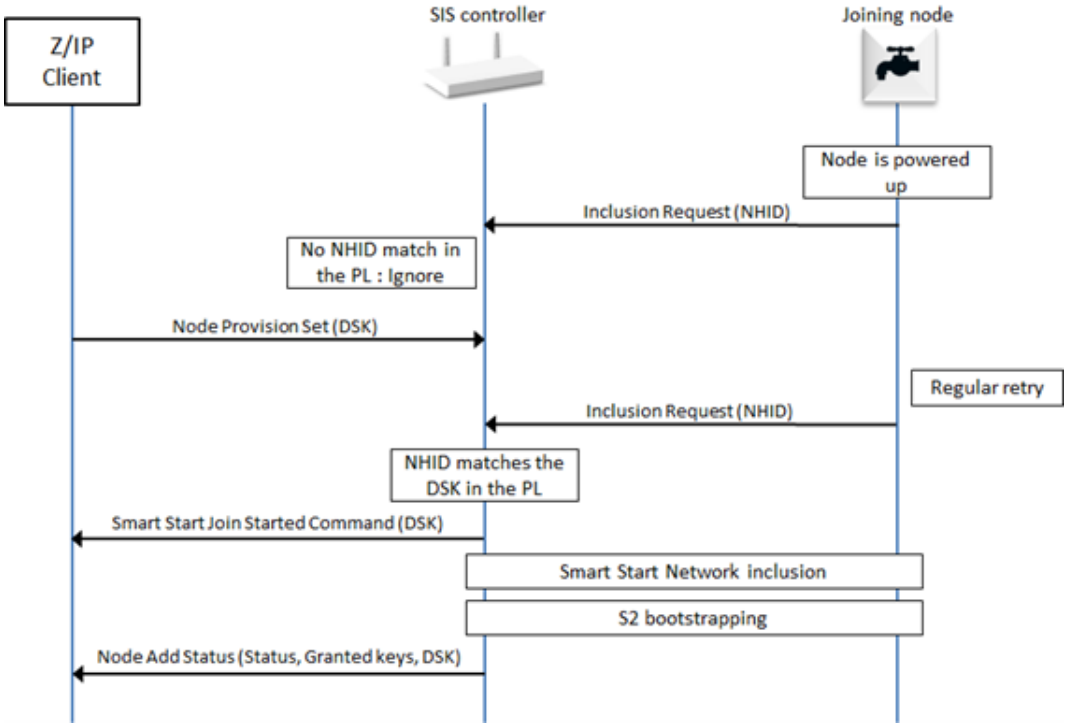


Figure 5.19: Smart Start inclusion (2)

5.2.5.12.9 Z/IP Gateway receiving an INIF from a node (the provisioning list) included in another network

The frame flow for a Smart Start inclusion of a node included in another network is shown in Figure 5.20.

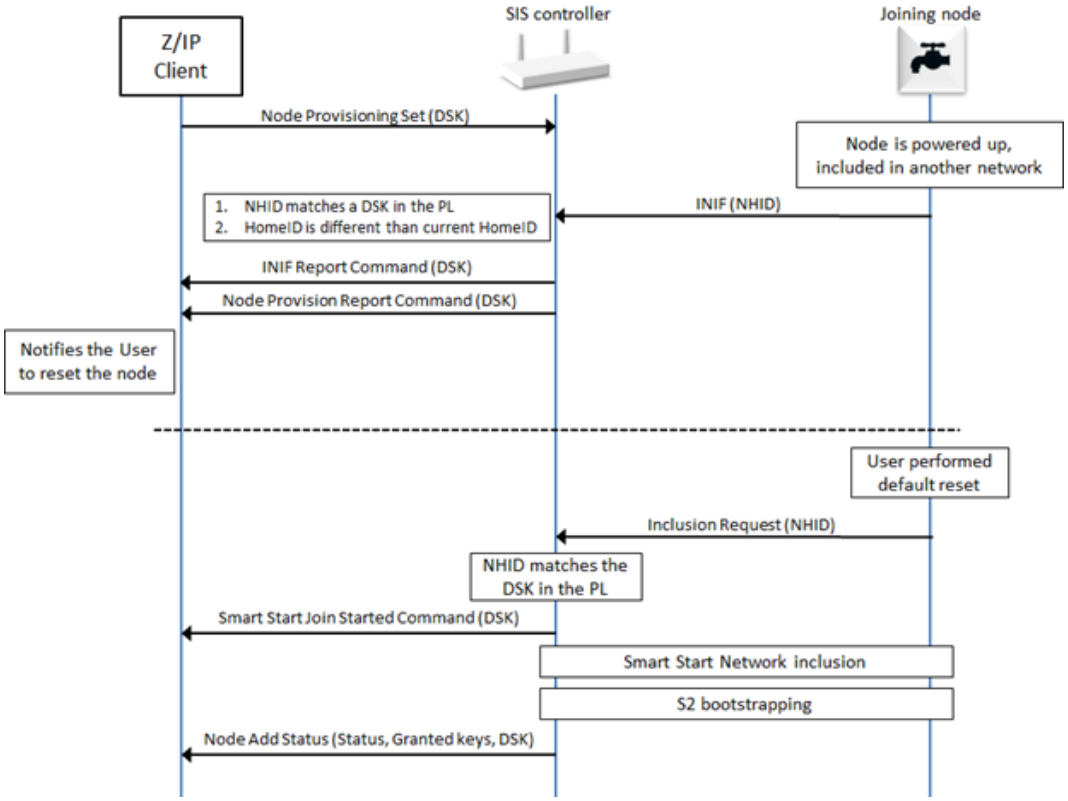


Figure 5.20: Smart Start inclusion (3)

5.2.5.12.10 Z/IP Gateway including an S2 only node that is on the provisioning list

The frame flow for an S2 only node (non-Smart Start) inclusion is shown in Figure 5.21. The Z/IP Client can decide to automatically grant the requested S2 keys or ask the user for confirmation.

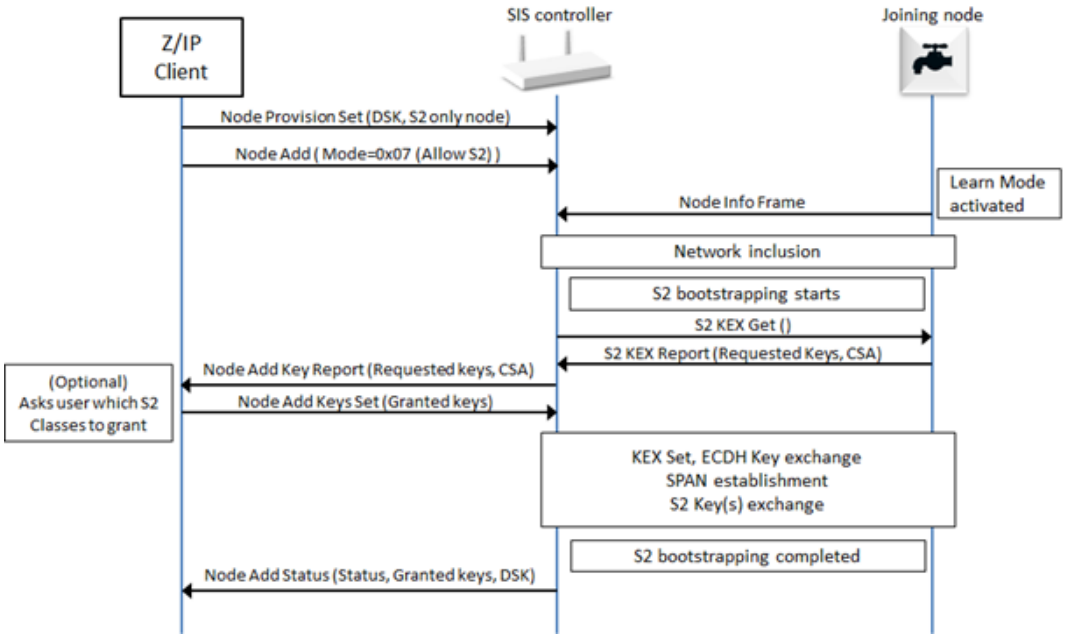


Figure 5.21: S2 Only Node inclusion with user interaction

5.2.5.13 Network management inclusion command class, version 4

5.2.5.13.1 Compatibility considerations

The Network Management Inclusion Command Class, version 4 is backwards compatible with the Network Management Inclusion Command Class, version 3.

A node supporting the Network Management Inclusion Command Class, version 4 MUST also support the Network Management Inclusion Command Class, version 3.

All commands and fields not mentioned in this version MUST remain unchanged from version 3.

This version of the Network Management Inclusion Command Class introduces support for the Z-Wave Long Range protocol. The following commands are updated:

- Node Remove Status Command
- Failed Node Remove Command
- Failed Node Remove Status Command

The following command is added:

- Extended Node Add Status Command

5.2.5.13.2 Command class dependencies

A node supporting the Network Management Inclusion Command Class, version 4 MUST also support the Node Provisioning Command Class, version 1.

A node supporting the Network Management Inclusion Command Class, version 4 MUST also support the Node Provisioning Bootstrapping Mode TLV (type=0x36) with value set to 0x02 (Z-Wave Long Range SmartStart Bootstrapping)

Node remove status command

This command is used to advertise the status of a node removal attempt.

Table 5.97: Node Remove Status Command V4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = NODE_REMOVE_STATUS (0x04)							
Seq No							
Status							
NodeID							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

Fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field is used to advertise the NodeID that was attempted to be removed from the network.

This field SHOULD be set to 0x00 if no attempt has been made.

This field MUST be set to 0xFF if the removed NodeID is greater than 255.

Extended NodeID (2 bytes)

This field is used to advertise the NodeID that was attempted to be removed from the network.

This field MUST be set to the actual NodeID that was attempted to be removed from the network.

This field SHOULD be set to 0x00 if no attempt has been made.

5.2.5.13.3 Failed node remove command

This command is used to remove a non-responding node.

A non-responding node is put onto the failed NodeID list by a controller when detected. In case the node responds again at a later stage, it is removed from the failed NodeID list. A node MUST be on the failed NodeID list and as an extra precaution also fail to respond before it is removed. Responding nodes MUST NOT be removed.

The Failed Node Remove Status Command MUST be returned in response to this command when the removal attempt has been made.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.98: Failed Node Remove Command V4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = FAILED_NODE_REMOVE (0x07)							
Seq No							
NodeID							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

All fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field is used to specify the NodeID of the failing node which MUST be removed.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the *Extended NodeID* field.

Extended NodeID (2 bytes)

This field is used to specify the NodeID of the failing node which MUST be removed.

If the *NodeID* field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the *NodeID* field is set to 0xFF, this field MUST indicate the NodeID of the failing node which MUST be removed.

5.2.5.13.4 Failed node remove status command

This command is used to report the results of a failed node removal attempt.

Table 5.99: Failed Node Remove Status Command V4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = FAILED_NODE_REMOVE_STATUS (0x08)							
Seq No							
NodeID							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

All fields not described below MUST remain unchanged from version 3.

NodeID (1 byte)

This field is used to specify the NodeID of the failing node which was attempted to be removed.

If this field is set to 0xFF, it MUST indicate that the NodeID is indicated in the *Extended NodeID* field.

Extended NodeID (2 bytes)

This field is used to specify the NodeID of the failing node which was attempted to be removed.

If the *NodeID* field is in the range 0x00..0xFE, this field MUST be set to the same value as the *NodeID* field by a sending node.

If the *NodeID* field is set to 0xFF, this field MUST indicate the NodeID of the failing node which was attempted to be removed.

5.2.5.13.5 Extended node add status command

This command is used to report the result of a node inclusion.

Table 5.100: Extended Node Add Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION (0x34)							
Command = EXTENDED_NODE_ADD_STATUS (0x16)							
Seq No							
Status							
Assigned NodeID (MSB)							
Assigned NodeID (LSB)							
Node Info Length							
List.	Z-Wave Protocol Specific Part						
Opt. Func.	Z-Wave Protocol Specific Part						
Basic Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							
Granted Keys							
KEX Fail Type							

Fields not described below MUST be identical to the *Node add status command*

Assigned NodeID (2 bytes)

This field MUST indicate the assigned NodeID to the newly added node. This field's value is valid only if the Status field is set to ADD_NODE_STATUS_DONE.

This field MUST be set to 0x00 if no new NodeID was assigned to an included node.

Status (1 byte)

This field is used to indicate the outcome of the SmartStart inclusion and MUST be encoded according to Table 5.103.

Table 5.101: Extended Node Add Status::Status parameter encoding

Value	Status identifier	Description
0x06	ADD_NODE_STATUS_DONE	The inclusion was successful and the node is ready to operate
0x07	ADD_NODE_STATUS_FAILED	The inclusion had an error, the joining node should have reset and no new node is part of the network.

5.2.5.13.6 Usage and frame flows

5.2.5.13.7 Z/IP Gateway including a SmartStart with extended nodeID

An example of a frame flow for a Z/IP Gateway including a node using an Extended NodeID is show in Figure 5.22.

If an entry is set to be bootstrapped using Z-Wave Long Range SmartStart, a Z/IP Gateway MUST issue the Extended Node Add Status Command when the inclusion is complete.

Node Add Status Command MUST be used only for Z-Wave S2 or Z-Wave SmartStart inclusions

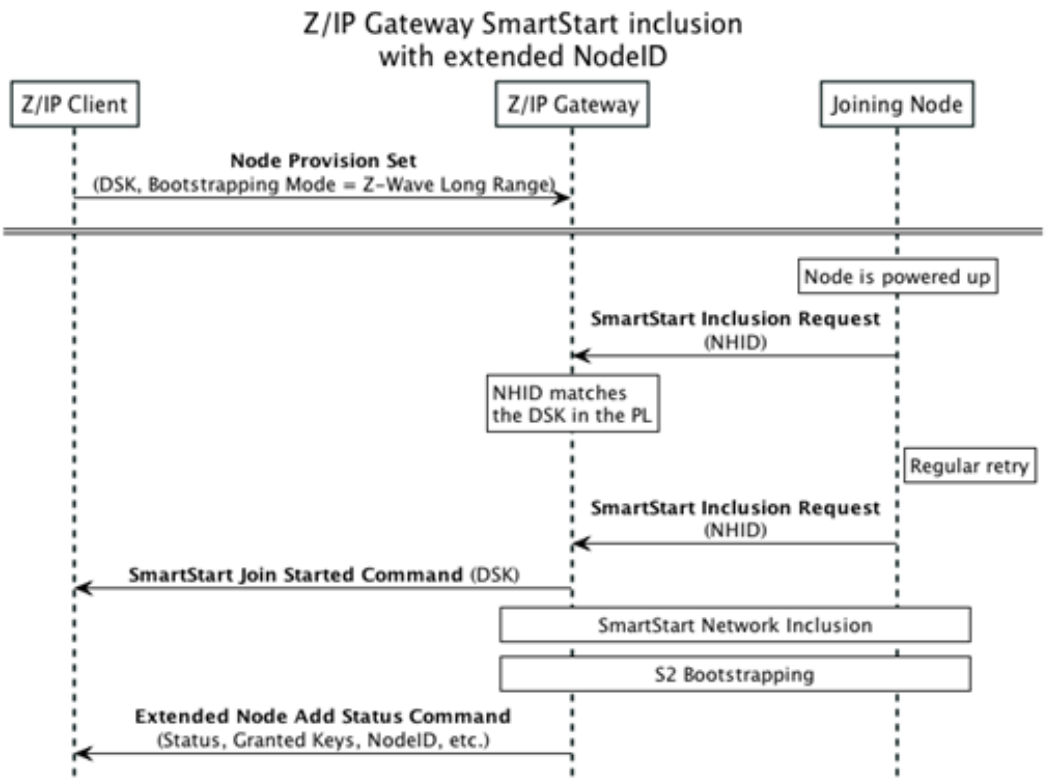


Figure 5.22: Extended NodeID SmartStart inclusion

5.2.5.14 Network management primary command class, version 1 [OBSOLETED]

CC:0054.01.00.11.001

Warning: THIS COMMAND CLASS HAS BEEN OBSOLETED New implementations MUST NOT support this Command Class.

The Network Management Primary Command Class provides functions to pass on the primary role to another controller.

5.2.5.14.1 Controller change command

This command is used to add a controller node to the network and assign the primary controller role to the included controller.

This command has the same functionality as Node Add with the exception that the new controller will become the primary controller and the controller adding the node will become secondary.

CC:0054.01.01.11.001 The Controller Change Status Command MUST be returned in response to this command.

CC:0054.01.01.11.002 This command MUST NOT be issued via multicast addressing.

A receiving node MUST ignore this command if it is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods

Table 5.102: Controller Change Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PRIMARY (0x54)							
Command = CONTROLLER_CHANGE (0x01)							
Seq No							
Reserved							
Mode							
tx Options							

Seq No (1 byte)

Refer to *Sequence number management*.

Reserved

CC:0054.01.01.11.004 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Mode (1 byte)

This field is use to indicate to the receiving node if the controller change mode must be activated or
CC:0054.01.01.11.005 de-activated. This field MUST comply with Table 5.103.

Table 5.103: Controller Change::Mode encoding

Value	Mode identifier	Description
0x02	CONTROLLER_CHANGE_START	Start the process of creating a new primary controller for the network
0x05	CONTROLLER_CHANGE_STOP	Stop the controller change and report a failure

tx Options (1 byte)

CC:0054.01.01.11.006 The tx Options field allows a controlling node to specify if transmissions MUST use special properties.. This field MUST be treated as a bitmask and MUST comply with Table 5.104

Table 5.104: Controller Change::Tx Options encoding

Value	Mode identifier	Description
0x00	NULL	Transmit at normal power level without any transmit options.
0x02	TRANSMIT_OPTION_LOW_POWER	Transmit at low output power level (1/3 of normal RF range)
0x20	TRANSMIT_OPTION_EXPLORE	Resolve new routes via explorer discovery if existing routes fail

5.2.5.14.2 Controller change status command

This command is used to advertise the outcome of the Controller Change attempt.

Table 5.105: Controller Change Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_PRIMARY (0x54)							
Command = CONTROLLER_CHANGE_STATUS (0x02)							
Seq No							
Status							
Reserved							
New NodeID							
Node Info Length							
List.	Z-Wave Protocol Specific Part						
Opt. Func.	Z-Wave Protocol Specific Part						
Basic Device Class							
Generic Device Class							
Specific Device Class							
Command Class 1 *)							
...							
Command Class N *)							

*) Command classes may be extended –> spanning two bytes for one command class

For fields' description, refer to *Node add status command*.

5.2.5.15 Network management installation and maintenance command class, version 1

The Network Management Installation and Maintenance Command Class is used to access statistical data. Data relating to the transmission of an actual frame may be obtained via the Z/IP Packet Installation and Maintenance Header Extension.

- – **All Transmissions / Route Information:**
  - – **Packet Error Count (PEC)** – Also sometimes referred to as PER. The number of unsuccessful transmissions experienced by the device.
  - – **Transmission Counter (TC)** – Number of frames sent by the specified device.
  - – **Neighbors (NB)** – Information on known neighbors for a specified device.
  - – **Network Management - Priority Route Set**
  - – **Network Management - Priority Route Get**
  - – **Network Management - Priority Route Report**

5.2.5.15.1 Priority route set

This command is used to set the network route to use when sending commands to the specified NodeID.

CC:0067.01.01.12.001 The use of this command is NOT RECOMMENDED.

Table 5.106: Priority Route Set

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = PRIORITY_ROUTE_SET (0x01)							
NodeID							
Repeater 1 [First repeater]							
Repeater 2							
Repeater 3							
Repeater 4 [Last repeater]							
Speed							

NodeID (1 byte)

CC:0067.01.01.11.001 This field is used to specify the destination NodeID for which a last working route MUST be set.

Repeater (4 bytes)

This field is used to specify repeaters for the route. Each byte represents a NodeID and the first field (Repeater 1) is the first repeater of the route.

CC:0067.01.01.11.002 The value 0x00 MUST indicate that the byte does not represent a repeater. If the route is shorter than four repeaters, unused repeaters fields MUST be set to 0x00. If Repeater 1 is set to 0x00, it means that the Last Working Route is direct (nodes are within direct reach).

Speed (1 byte)

CC:0067.01.01.11.003 This field is used to indicate which speed MUST be used for the route. This field MUST comply with Table 5.107.

Table 5.107: IME Speed Encoding

Value	Speed
0x01	9.6 kbit/sec
0x02	40 kbit/sec
0x03	100 kbit/sec

5.2.5.15.2 Priority route get

This command is used to query the current network route from a node for a given destination.

CC:0067.01.02.11.001 The Priority Route Report MUST be returned in response to this command.

CC:0067.01.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:0067.01.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.108: Priority Route Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = PRIORITY_ROUTE_GET (0x02)							
NodeID							

NodeID (1 byte)

This field is used to specify the NodeID destination for which the current network route is requested.

5.2.5.15.3 Priority route report

This command is used to advertise the current network route in use for an actual destination NodeID.

Table 5.109: Priority Route Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = PRIORITY_ROUTE_REPORT (0x03)							
NodeID							
Type							
Repeater 1 - 1 [First repeater]							
Repeater 2 – 1							
Repeater 3 – 1							
Repeater 4 - 1 [Last repeater]							
Speed -1							

Type (1 byte)

CC:0067.01.03.11.001 This field is used to indicate the route type. It MUST comply with Table 5.110. A node MUST return the route with the highest priority value if several routes are available at the node.

Table 5.110: Route Type Encoding

Value	Identifier	Description	Priority
0x00	–	There is no route defined for the target NodeID. In this case, the Repeater and Speed fields MUST be set to 0x00 and ignored by a receiving node.	4 (lowest)
0x01	ZW_PRIORITY_ROUTE_ZW_LWR	The returned route is a last working route. The Last Working route is the last successful route used between the sender and receiver.	2
0x02	ZW_PRIORITY_ROUTE_ZW_NLWR	The returned route is a next to last working route. It is a route which was Last Working Route and has been replaced by a new route.	3
0x10	ZW_PRIORITY_ROUTE_APP_PR	The returned has been determined by the application	1 (highest)

CC:0067.01.03.11.002

Repeater (4 bytes)

Refer to *Priority route set*.

Speed (1 byte)

Refer to *Priority route set*.

5.2.5.15.4 Statistics get

This command is used to query Installation and Maintenance statistics from a node.

CC:0067.01.04.11.001 The Statistics Report MUST be returned in response to this command.

CC:0067.01.04.11.002 This command MUST NOT be issued via multicast addressing.

CC:0067.01.04.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.111: Statistics Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = STATISTICS_GET (0x04)							
NodeID							

NodeID (1 byte)

This field is used to specify the NodeID for which statistics are requested.

5.2.5.15.5 Statistics report

This command is used to report Installation and Maintenance statistics recorded by a node.

Table 5.112: Statistics Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = STATISTICS_REPORT (0x05)							
NodeID							
Statistics – Type 1							
Statistics – Length 1							
Statistics – Value 1							
...							
Statistics – Type N							
Statistics – Length N							
Statistics – Value N							

NodeID (1 byte)

CC:0067.01.04.11.004 This field MUST carry the same value as received in the Statistics Get Command.

Statistics (N bytes)

CC:0067.01.04.11.005 The statistics field MUST be formatted as cascaded Type-Length-Value (TLV) structures. The Z/IP CC:0067.01.04.13.001 Gateway MAY send any combination of TLV structures. Valid types are shown in Table 5.113.

Table 5.113: Statistics Get::Type encoding

Name	Statistics - Type	Statistics - Length (Bytes)
Route Changes (RC)	0	1
Transmission Count (TC)	1	1
Neighbors (NB)	2	n
Packet Error Count (PEC)	3	1
Sum of transmission times (TS)	4	4
Sum of transmission times squared (TS2)	5	4

CC:0067.01.04.11.006 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

5.2.5.15.6 Route changes (RC)

Table 5.114: Route Changes

7	6	5	4	3	2	1	0
Statistics - Type = 0x00							
Statistics – Length = 1							
Statistics – Value = Route Changes							

Route Changes (1 byte)

The RC field is used to advertise the number of routing attempts needed to reach a destination. The number is a combination of Last Working Route (LWR) changes and Jitter measurements during transmission attempts between the Z/IP Gateway and the Z-Wave device.

RC is incremented automatically by the Z/IP Gateway when either of the below conditions are true:

- – Last Working Route changed from the transmission of one command to the next
- – T_n - T_{n-1} > 150ms where T_n and T_{n-1} = *the time needed to complete a transmission of a command*
  - – IF 2 channel and FLIRS node, RC: $T_n = T_n$ mod 1100
  - – IF 3 channel and FLIRS node, RC cannot increment based on time calculation

5.2.5.15.7 Transmission Count (TC)

Table 5.115: Transmission Count

7	6	5	4	3	2	1	0
Statistics - Type = 0x01							
Statistics – Length = 1							
Statistics – Value = Transmission Count							

Transmission Count (1 byte)

Total number of transmissions sent by all Z/IP Clients through the Z/IP GW to the specified Z-Wave destination node.

5.2.5.15.8 Neighbors (NB)

Table 5.116: Neighbors

7	6	5	4	3	2	1	0
Statistics - Type = 0x02							
Statistics – Length = N * 2							
Statistics – Value = NodeID 1							
Statistics – Value = Repeater 1	Reserved		Statistics – Value = Speed 1				
...							
Statistics – Value = NodeID N							
Statistics – Value = Repeater N	Reserved		Statistics – Value = Speed N				

NodeID (N * 1 byte)

The NodeID of the actual neighbor.

Speed (N * 4 bits)

Table 5.117: Statistics Report::Speed Encoding

Value	Speed
0x01	9.6 kbit/sec
0x02	40 kbit/sec
0x04	100 kbit/sec

CC:0067.01.04.11.007 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Repeater (N * 1 bit)

If this bit is set then the node is a repeater.

5.2.5.15.9 Packet error count (PEC)

Table 5.118: Packet Error Count

7	6	5	4	3	2	1	0
Statistics - Type = 0x03							
Statistics – Length = 1							
Statistics – Value = Packet Error Count							

Packet Error Count (1 byte)

CC:0067.01.04.11.008 Also sometimes referred to as PER. PEC is measured by the Gateway. The PEC value MUST be incremented each time the Gateway detects a failing transmission for each specific Z-Wave destination node.

Sum of transmission times (TS)

Table 5.119: Sum of Transmission Times

7	6	5	4	3	2	1	0
Statistics - Type = 0x04							
Statistics – Length = 4							
Statistics – Value = Sum of transmission times 1 (MSB)							
Statistics – Value = Sum of transmission times 2							
Statistics – Value = Sum of transmission times 3							
Statistics – Value = Sum of transmission times 4 (LSB)							

Sum of transmission times (4 bytes)

The sum of all transmission times. This may be used to calculate the average transmission time. The time is given as a 32-bit unsigned integer MSB in milliseconds.

$$\langle T \rangle = \frac{1}{N} \sum_{i}^{N} T_i$$

Where N is the number of transmissions.

5.2.5.15.10 Sum of transmission times squared (TS2)

Table 5.120: Sum of Transmission Times Squared

7	6	5	4	3	2	1	0
Statistics - Type = 0x05							
Statistics – Length = 4							
Statistics – Value = Sum of transmission times squared 1 (MSB)							
Statistics – Value = Sum of transmission times squared 2							
Statistics – Value = Sum of transmission times squared 3							
Statistics – Value = Sum of transmission times squared 4 (LSB)							

Sum of transmission times squared (4 bytes)

The sum of the square of all transmission times. This may be used to calculate the variance of the transmission time. The time is given as a 32 bit unsigned integer MSB in milliseconds^2.

The Variance may be calculated as follows:

$$\langle T^2 \rangle = \frac{1}{N} \sum_{i}^{N} T_i^2$$

(König-Huygens theorem)

Where N is the number of transmissions.

A high variance is a sign of a bad link.

5.2.5.15.11 Statistics clear

This command is used to clear all statistic registers maintained by the node.

Table 5.121: Statistics Clear

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = STATISTICS_CLEAR (0x06)							

A receiving node MUST set all counters to 0.

5.2.5.15.12 Use cases

5.2.5.15.13 Intranode network management: TV OSD system controlling lamps

Intranode network management is the process close to Z-Wave API programming. No messages ever leave the device. Messages only flow between different software modules.

Use Case: TV OSD System (island mode)

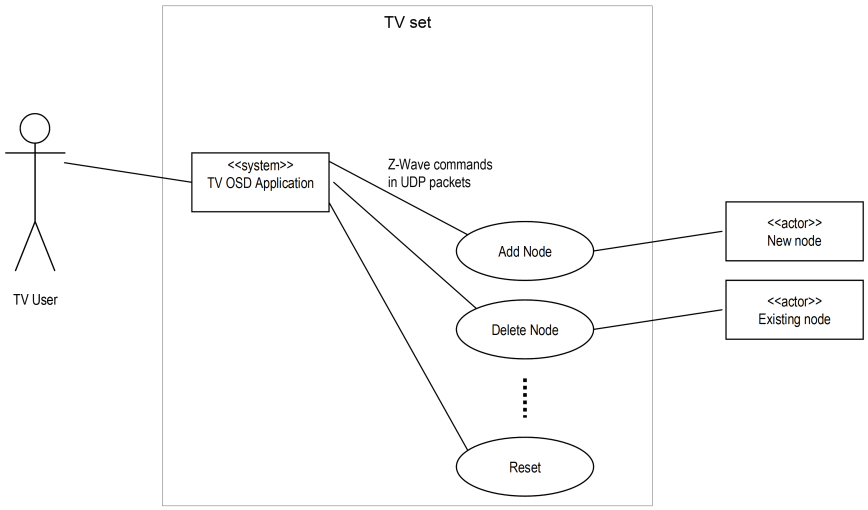


Figure 5.23: TV OSD System controlling lamps

Using UDP/IP for carrying the messages allows for a simple integration interface between applications designed by different partners.

5.2.5.15.14 Intranet network management: remote controlling a primary controller

Intranet network management extends the use of command messages to separate physical devices. Messages flow between software modules but the modules reside in separate physical entities having individual IP addresses – or at least separate NodeIDs.

Use Case: Managing a primary static controller from a remote control

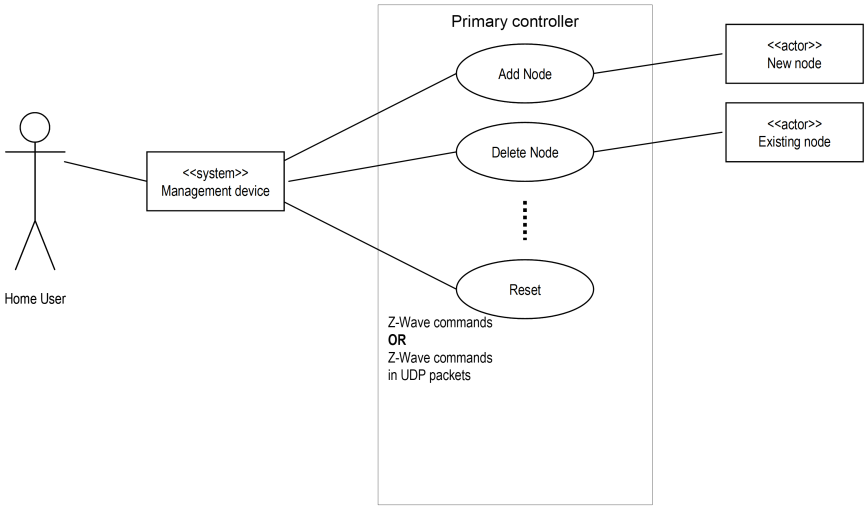


Figure 5.24: Managing a primary static controller from a remote control

Network management via messages allows for sophisticated interfaces to the primary controller of

a network. Controllers with SUC/SIS capability may also leverage from the Network Management command classes.

5.2.5.15.15 Internet network management #1: call-center support for TV OSD user

Internet network management uses the same command messages. Messages flow between software modules but the modules reside in separate physical entities in a non-trusted environment such as the Internet. Remote access technologies should be used to protect the communication.

In this use case a TV user may call the service provider for support in adding a new lamp to the network.

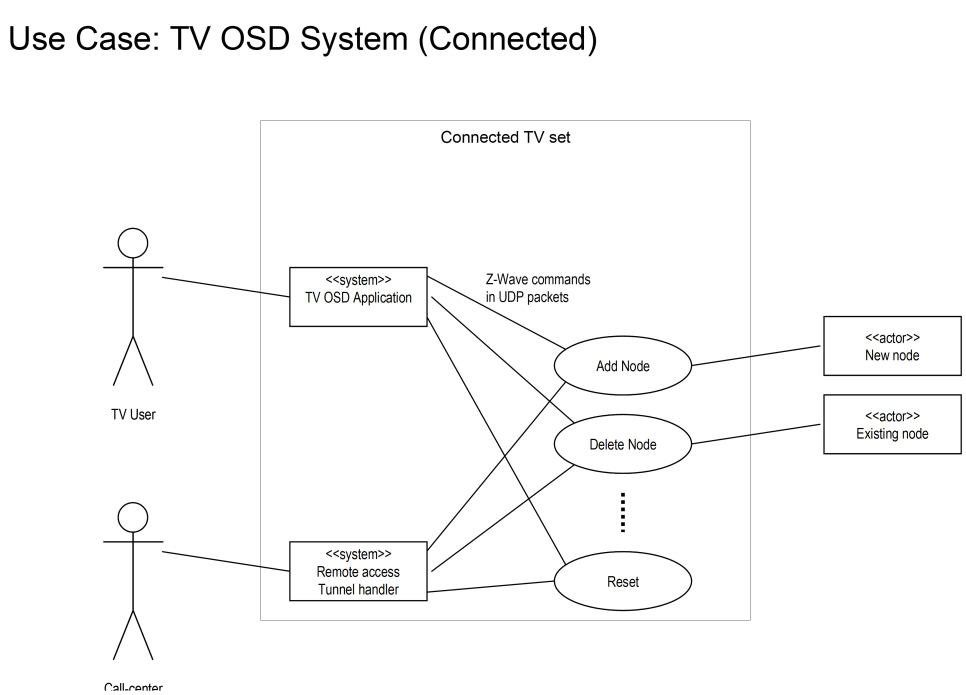


Figure 5.25: TV OSD System

Internet network management #2: remote management of Z/IP network

In this use case a user may use an IP based home control management system running in the LAN for setting up the Z/IP network. The user may use normal UDP transport in the LAN environment. Due to the critical nature of the network management command classes the user however should use remote access protection technologies over LAN as well as over Internet. The benefit of designing a home control system using remote access protection by default is that it may be moved from a location in the LAN to any place in the Internet and work completely unaffected.

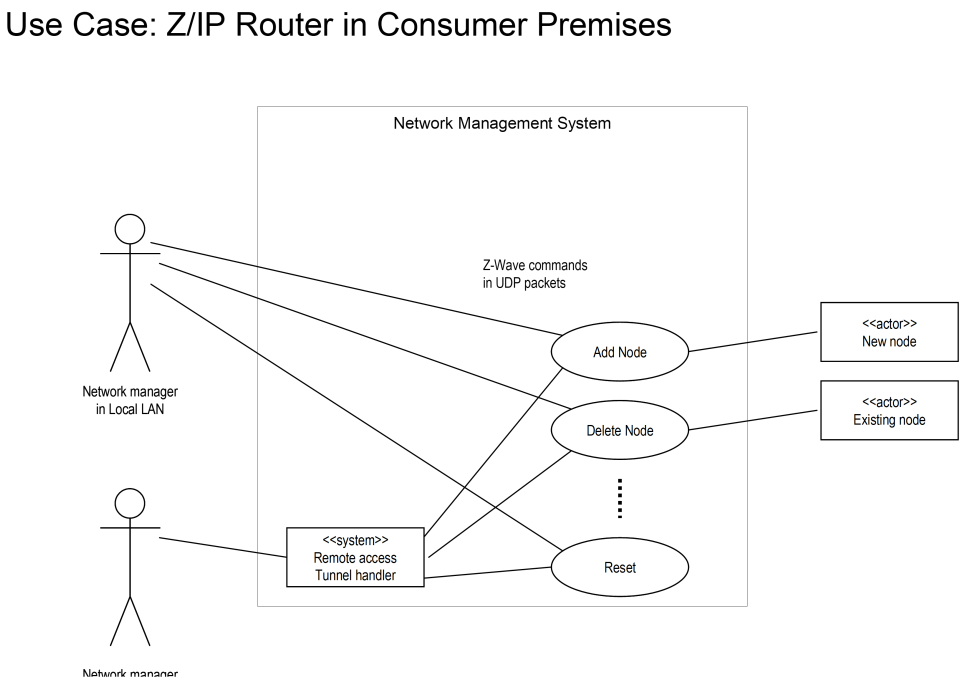


Figure 5.26: Z/IP Router in consumer premises

5.2.5.15.16 Traffic flow: gathering node information

The following sequence diagram introduces a new concept of gathering Node Information.

The node list provides an overview of the nodes in the network; as good as the Z/IP gateway can provide this information. Using that node list, the requesting host may request information on individual nodes from the Z/IP Gateway. The "Node Info Cached Get" command reports all supported and controlled classes.

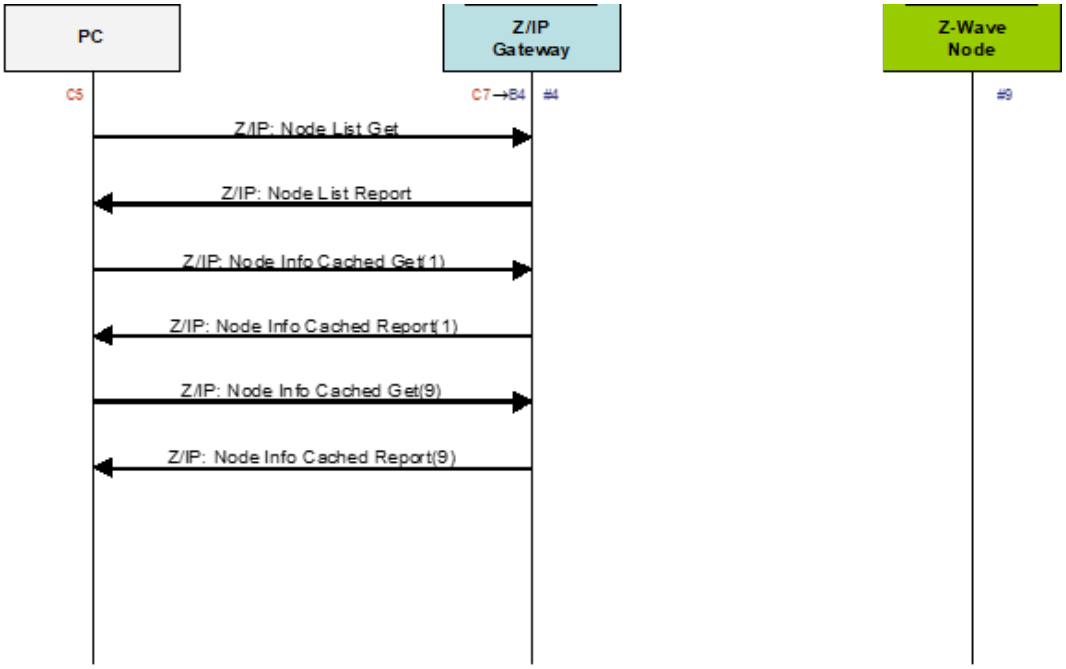


Figure 5.27: Gathering node information

5.2.5.16 Network management installation and maintenance command class, version 2

5.2.5.16.1 Compatibility considerations

The Network Management Installation and Maintenance Command Class, version 2 is backwards compatible with the Network Management Installation and Maintenance Command Class, version 1.

CC:0067.02.01.21.002 All commands and fields not mentioned in this version MUST remain unchanged from version 1.

The following commands have been added to allow a supporting node to report the RSSI it measured in each channel of the network:

- RSSI Get Command
- RSSI Report Command

5.2.5.16.2 RSSI get command

This command is used to query the measured RSSI on the Z-Wave network from a node.

CC:0067.02.07.11.001 The RSSI Report Command MUST be returned in response to this command.

CC:0067.02.07.11.002 This command MUST NOT be issued via multicast addressing.

CC:0067.02.07.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.122: RSSI Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = RSSI_GET (0x07)							

5.2.5.16.3 RSSI report command

This command is used to advertise the measured RSSI on the Z-Wave network for each used channel.

Table 5.123: RSSI Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = RSSI_REPORT (0x08)							
Channel 1 RSSI							
Channel 2 RSSI							
Channel 3 RSSI							

Channel 1 RSSI (8 bits)

CC:0067.02.08.11.001 This field MUST carry the measured RSSI value on channel 1.

CC:0067.02.08.11.002 This field MUST be encoded as using signed representation in the dBm unit and according to Table 5.124.

Channel 2 RSSI (8 bits)

CC:0067.02.08.11.003 This field MUST carry the measured RSSI value on channel 2.

CC:0067.02.08.11.004 This field MUST be encoded as using signed representation in the dBm unit and according to Table 5.124.

Channel 3 RSSI (8 bits)

CC:0067.02.08.11.005 This field MUST carry the measured RSSI value on channel 3, if applicable.

CC:0067.02.08.11.006 This field MUST be encoded as using signed representation in the dBm unit and according to Table 5.124.

Table 5.124: RSSI Encoding

Value(signed)	Description
127 (0x7F)	RSSI_NOT_AVAILABLE. This value is returned for unused channels or if no RSSI measurement is available.
126 (0x7E)	RSSI_MAX_POWER_SATURATED This value is returned if the measured RSSI is above the maximum power.
125 (0x7D)	RSSI_BELOW_SENSITIVITY. This value is returned if the measured RSSI is below the receiver's sensitivity.
-32..-128 (0xE0..0x80)	These values represent the actual RSSI measurement value from respectively -32 dBm to -128 dBm

CC:0067.02.08.11.007 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

5.2.5.17 Network management installation and maintenance command class, version 3

5.2.5.17.1 Compatibility considerations

The Network Management Installation and Maintenance Command Class, version 3 is backwards compatible with the Network Management Installation and Maintenance Command Class, version 2.

All commands and fields, that are not mentioned in this version MUST remain unchanged from version 2.

The following command has been added to notify the Z/IP Client application the occurrence of S2 Nonce Resynchronization event, (including extended NodeID support) :

- S2 Resynchronization Event Command

The following commands are introduced to support extended NodeIDs:

- Extended Statistics Get
- Extended Statistics Report

5.2.5.17.2 S2 resynchronization event command

This command is used to notify the Z/IP Client application the occurrence of S2 Nonce Resynchronization event. This will allow the client application to recover faster from a synchronization error that may cause message delay and loss.

The Z/IP Gateway MUST send this command to the Unsolicited destinations when the Gateway received Nonce Report with SOS flag equal to 1 while Verify Delivery to the sending node is inactive.

The Z/IP Gateway MUST NOT send this command in the following cases:

- The Z/IP Gateway received the Nonce Report with SOS flag equals to 1 and Verify Delivery to sending node is active.
- The Z/IP Gateway received the Nonce Report with SOS flag equals to 1 and the ZIP NAK is sent to the Z/IP Client application.
- The Z/IP Gateway received the Nonce Report with SOS flag equals to 1 when the sending node cannot decrypt the frame sent from the Gateway.

- The Z/IP Gateway received Nonce Get (i.e., node coming out of power reset.

Table 5.125: S2 Resynchronization Event Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = S2_RESYNCHRONIZATION_EVENT (0x09)							
NodeID							
Reason							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

NodeID (8 bits)

The NodeID field contains the NodeID of the peer node triggering a nonce resynchronization.

This field MUST be set to 0xFF if the NodeID triggering a nonce resynchronization is greater than 255.

Reason (8 bits)

The reason field contains the detailed reason for the Resynchronization event. This field MUST encoded as described in Table 5.126. This field MUST use signed encoding.

Table 5.126: S2 Resynchronization Event Reason Encoding

Value	Description
0	SOS_EVENT_REASON_UNANSWERED A Nonce Report with SOS equals to 1 was received at an unexpected time and no response was sent. Application may use this information to abort Supervision Report timeout if the remote NodeID matches. The Nonce Report was unanswered because the retransmission was performed while the S2 layer was idle or transmitting to another NodeID. In this case, a frame to NodeID was most likely lost. If the ZIP Client had only one frame outstanding with NodeID, it can safely be assumed that the frame was lost. Note: Supervision Encapsulation should be used to acknowledge outstanding frames.
-127..-1	*Reserved*
1..127	*Reserved*

Extended NodeID (2 bytes)

This field is used to advertise the NodeID of the peer node triggering a nonce resynchronization.

This field MUST be set to the actual NodeID that was triggered a Nonce resynchronization.

5.2.5.17.3 Extended statistics get

This command is used to query Installation and Maintenance statistics from a node.

The Extended Statistics Report MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.127: Extended Statistics Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = EXTENDED_STATISTICS_GET (0x0B)							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

Extended NodeID (2 bytes)

This field is used to specify the NodeID for which statistics are requested. A receiving node MUST return a report for the NodeID indicated in this field.

5.2.5.17.4 Extended statistics report

Table 5.128: Extended Statistics Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = EXTENDED_STATISTICS_REPORT (0x0C)							
Extended NodeID (MSB)							
Extended NodeID (LSB)							
Statistics – Type 1							
Statistics – Length 1							
Statistics – Value 1							
...							
Statistics – Type N							
Statistics – Length N							
Statistics – Value N							

NodeID (2 bytes)

This field is used to advertise the NodeID for which statistics are advertised. No TLV MUST be appended for non-existing NodeIDs or if no statistics have been saved for the advertised NodeID.

Statistics (N bytes)

The statistics field MUST be formatted as cascaded Type-Length-Value (TLV) structures. The Z/IP Gateway MAY send any combination of TLV structures. Valid types are shown in Table 5.113 and described in section *Statistics report*.

5.2.5.18 Network management installation and maintenance command class, version 4

5.2.5.18.1 Compatibility considerations

The Network Management Installation and Maintenance Command Class, version 4 is backwards compatible with the Network Management Installation and Maintenance Command Class, version 3.

All commands and fields not mentioned in this version MUST remain unchanged from version 3.

The following commands have been added to allow a supporting node to configure the channel to use for Z-Wave Long Range:

- Z-Wave Long Range Channel Configuration Set
- Z-Wave Long Range Channel Configuration Get
- Z-Wave Long Range Channel Configuration Report

The following commands have been extended to allow a supporting node to report the RSSI it measured in the Z-Wave Long Range channel of the network:

- RSSI Report Command

5.2.5.18.2 RSSI report command

This command is used to advertise the measured RSSI on the Z-Wave network for each used channel.

Table 5.129: RSSI Report Command V4

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = RSSI_REPORT (0x08)							
Channel 1 RSSI							
Channel 2 RSSI							
Channel 3 RSSI							
Z-Wave Long Range Primary Channel RSSI							
Z-Wave Long Range Secondary Channel RSSI							

Fields not described below MUST remain unchanged from version 3.

Z-Wave Long Range Primary Channel RSSI (8 bits)

This field MUST carry the measured RSSI value on the Z-Wave Long Range Primary Channel.

This field MUST be encoded as using signed representation in the dBm unit and according to Table 5.124.

Z-Wave Long Range Secondary Channel RSSI (8 bits)

This field MUST carry the measured RSSI value on the Z-Wave Long Range Secondary Channel.

This field MUST be encoded as using signed representation in the dBm unit and according to Table 5.124.

5.2.5.18.3 Z-Wave long range channel configuration set

This command is used to configure which channel to use for the Z-Wave Long Range protocol.

Table 5.130: Z-Wave Long Range Channel Configuration Set

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = ZWAVE_LR_CHANNEL_CONFIGURATION_SET (0x0A)							
Z-Wave Long Range Channel							

Z-Wave Long Range Channel (8 bits)

This field is used to specify which channel to use for Z-Wave Long Range.

The value 0x01 MUST indicate to use the Primary Z-Wave Long Range Channel.

The value 0x02 MUST indicate to use the Secondary Z-Wave Long Range Channel.

All other values are reserved.

5.2.5.18.4 Z-Wave long range channel configuration get

This command is used to request the currently configured Z-Wave Long Range Channel.

The Z-Wave Long Range Channel Configuration Report MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.131: Z-Wave Long Range Channel Configuration Get

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = ZWAVE_LR_CHANNEL_CONFIGURATION_GET (0x0D)							

5.2.5.18.5 Z-Wave long range channel configuration report

This command is used to advertise the configured Z-Wave Long Range Channel.

Table 5.132: Z-Wave Long Range Channel Configuration Report

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NETWORK_MANAGEMENT_INSTALLATION_MAINTENANCE (0x67)							
Command = ZWAVE_LR_CHANNEL_CONFIGURATION_REPORT (0x0E)							
Z-Wave Long Range Channel							

Z-Wave Long Range Channel (8 bits)

This field is used to advertise the channel to use for Z-Wave Long Range.

The value 0x01 MUST indicate to use the Primary Z-Wave Long Range Channel.

The value 0x02 MUST indicate to use the Secondary Z-Wave Long Range Channel.

All other values are reserved.

A supporting node SHOULD use the Primary Z-Wave Long Range Channel by default.

5.2.6 No Operation Command Class, version 1

The No Operation Command Class is used to check if a node is reachable by sending a Command less frame to the specified destination. Feature used by the Z-Wave protocol in many situations e.g. checking that an excluded node is non-responding. This Command can also be used on application level e.g. checking if a SUC/SIS is reachable from a new node in the network. This command class contains no command identifier and data.

Notice: It is not necessary to announce the No Operation Command Class in the NIF.

Table 5.133: No Operation Command Class, version 1

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NO_OPERATION (0x00)							

5.2.7 Node Provisioning Command Class, version 1

The Node Provisioning Command Class is used to manage a list of unique nodes (Node Provisioning List) in a Smart Start enabled controller or gateway.

5.2.7.1 Terminology

Smart start allows a controller to include new nodes in a network (or keep them out) without user interaction.

A Smart Start enabled controller or gateway maintains a **Node Provisioning** List or **Provisioning List** (PL). The Provisioning List is a list of unique nodes and their additional associated meta data necessary for performing their network inclusion and security bootstrapping.

A **Provisioning List entry** represents a node and its associated data. Provisioning List entries may also be used for ignoring nodes.

A Z/IP Client or controller can read and edit the Provisioning List entries of a Z/IP Gateway or controller using this Command Class.

5.2.7.2 Compatibility considerations

CC:0078.01.00.22.001 This Command Class MAY be carried in Z/IP Packets or in Z-Wave frames. However, this Command Class SHOULD only be used in Z/IP Packets.

CC:0078.01.00.21.001 A node supporting this Command Class MUST support at least 232 entries in its Node Provisioning List.

5.2.7.3 Security considerations

This Command Class allows a controlling node to include new nodes in the Z-Wave network and grant them all the security keys.

CC:0078.01.00.41.001 A node supporting this Command Class MUST NOT support it in a Z-Wave network if its highest Security Class is lower than S2 Access Control.

5.2.7.4 Node Provisioning Set Command

This command is used to create or update an entry in the node provisioning list of a supporting node.

Table 5.134: Node Provisioning Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_PROVISIONING (0x78)							
Command = NODE_PROVISIONING_SET (0x01)							
Seq No							
Reserved				DSK Length			
DSK 1							
...							
DSK N							
Meta Data Extension 1 (Optional)							
...							
Meta Data Extension M (Optional)							

Seq No (8 bits) Refer to *Sequence number management*.

Reserved

CC:0078.01.01.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

DSK Length (5 bits)

CC:0078.01.01.11.002 This field MUST indicate the length of the DSK field in bytes.

CC:0078.01.01.11.003 This field MUST be set to 16.

DSK (N bytes)

This field is used to advertise the DSK for the entry being added or updated.

CC:0078.01.01.11.004 A receiving node MUST add a new entry in the provisioning list if it does not have any entry with the advertised DSK value.

CC:0078.01.01.11.00D A receiving node MUST ignore a command attempting to create a new entry if the Provisioning List is full.

CC:0078.01.01.11.005 A receiving node MUST update the corresponding entry in the provisioning list if it already has an entry with the advertised DSK value.

CC:0078.01.01.11.006 The length of this field (in bytes) MUST be according to the DSK Length field value.

Meta Data Extension (M bytes)

This field is used to carry additional metadata associated to the node.

CC:0078.01.01.13.001 This field MAY contain zero, one or several extensions.

CC:0078.01.01.11.007 Each extension MUST comply with *Meta Data Extension Format* and [28].

CC:0078.01.01.11.009 If the Bootstrapping mode Type (0x36) is omitted from this command, the Bootstrapping mode value 1 (Smart Start Mode) MUST be assumed by the receiving node when creating a new entry.

CC:0078.01.01.11.00B If the SmartStart Inclusion Setting Type (0x34) is omitted from this command, the Inclusion setting value 0 (Pending) MUST be assumed by the receiving node when creating a new entry that has a SmartStart Bootstrapping mode.

CC:0078.01.01.11.00A The Network Status Type (0x37) MUST NOT be carried in this command. The Network Status Type (0x37) MUST be ignored if received in this command.

5.2.7.5 Node Provisioning Delete Command

This command is used to delete one or all entries in the node provisioning list of a supporting node. Already included nodes will stay in the Z-Wave network even if no more corresponding node provisioning list entry is kept by the controller.

Table 5.135: Node Provisioning Delete Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_PROVISIONING (0x78)							
Command = NODE_PROVISIONING_DELETE (0x02)							
Seq No							
Reserved				DSK Length			
DSK 1							
...							
DSK N							

Seq No (8 bits)

Refer to *Sequence number management*.

DSK Length (5 bits)

CC:0078.01.02.11.001 This field MUST indicate the length of the DSK field in bytes.

CC:0078.01.02.11.002 This field MUST be set to 0 or 16.

CC:0078.01.02.11.003 The value 0 MUST indicate that the receiving node MUST delete all entries in its Node Provisioning List.

CC:0078.01.02.11.004 The value 16 MUST indicate that the receiving node MUST delete the entry in its Node Provisioning List that match the advertised value in the DSK field.

DSK (N bytes)

This field is used to advertise the DSK for the entry being deleted.

CC:0078.01.02.11.005 A receiving node MUST delete the corresponding entry from the Node Provisioning List if it has an entry with the advertised DSK value.

CC:0078.01.02.11.006 A receiving node MUST ignore this command if it has no entry with the advertised DSK value.

CC:0078.01.02.11.007 The length of this field (in bytes) MUST be according to the DSK Length field value.

This field MUST be omitted if the DSK Length field is set to 0.

5.2.7.6 Node Provisioning Get Command

This command is used to request the metadata information associated to an entry in the node Provisioning List of the receiving node.

CC:0078.01.05.11.001 The Node Provisioning Report Command MUST be returned in response to this command.

CC:0078.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:0078.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.136: Node Provisioning Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_PROVISIONING (0x78)							
Command = NODE_PROVISIONING_GET (0x05)							
Seq No							
Reserved			DSK Length				
DSK 1							
...							
DSK N							

Seq No. (8 bits)

Refer to *Sequence number management*.

DSK Length (5 bits)

CC:0078.01.05.11.004 This field MUST indicate the length of the DSK field in bytes.

CC:0078.01.05.11.005 This field MUST be set to 16.

DSK (N bytes)

This field is used to advertise the DSK for the entry being requested.

CC:0078.01.05.11.006 A receiving node MUST return the corresponding DSK entry if it has an entry matching the requested DSK in its Provisioning List.

CC:0078.01.05.11.007 A receiving node MUST return a report containing no DSK (DSK Length set to 0) if the requested DSK value is not in its Provisioning List.

CC:0078.01.05.11.008 The length of this field (in bytes) MUST be according to the DSK Length field value.

5.2.7.7 Node Provisioning Report Command

This command is used to advertise the contents of an entry in the node Provisioning List of the sending node.

Table 5.137: Node Provisioning Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_PROVISIONING (0x78)							
Command = NODE_PROVISIONING_REPORT (0x06)							
Seq No							
Reserved			DSK Length				
DSK 1 (Optional)							
...							
DSK N (Optional)							
Meta Data Extension 1 (Optional)							
...							
Meta Data Extension M (Optional)							

Seq No (8 bits) Refer to *Sequence number management*.

DSK Length (5 bits)

CC:0078.01.06.11.001 This field MUST indicate the length of the DSK field in bytes.

CC:0078.01.06.11.002 This field MUST be set to 0 or 16

CC:0078.01.06.11.003 The value 0 MUST indicate that the requested DSK is not present in the Provisioning List.

DSK (N bytes)

CC:0078.01.06.11.004 This field is used to advertise the DSK for the Provisioning List entry being advertised.

CC:0078.01.06.11.005 The length of this field (in bytes) MUST be according to the DSK Length field value. This field MUST be omitted if the DSK Length field is set to 0.

Meta Data Extension (M bytes)

This field is used to carry additional metadata associated to the Provisioning List entry.

CC:0078.01.06.13.001 This field MAY contain several extensions.

CC:0078.01.06.11.006 Each extension Type, Length and Value MUST comply with *Meta Data Extension Format* and [28].

CC:0078.01.06.13.002 A supporting node MAY set the critical flag to 0 even when advertising critical extensions.

CC:0078.01.06.11.007 If the DSK Length field is set to 0, this field MUST be omitted.

If the DSK Length field is not set to 0:

- CC:0078.01.06.11.008 A sending node MUST advertise the SmartStart Inclusion Setting extension (type 0x34)
- CC:0078.01.06.11.009 A sending node MUST advertise the Bootstrapping mode extension (type 0x36)
- CC:0078.01.06.11.00B A sending node MUST advertise the Network Status extension (type 0x37)
- CC:0078.01.06.11.00A A sending node MUST advertise all other extension data kept in the Provisioning List

5.2.7.8 Node Provisioning List Iteration Get Command

This command is used to read the entire the provisioning list of a supporting node.

CC:0078.01.03.11.001 The Node Provisioning List Iteration Report Command MUST be returned in response to this command unless it is to be ignored.

CC:0078.01.03.11.002 This command MUST NOT be issued via multicast addressing.

CC:0078.01.03.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

CC:0078.01.03.11.004 A sending node MUST follow the frame flow in Section 5.2.7.11.1.

Table 5.138: Node Provisioning List Iteration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_PROVISIONING (0x78)							
Command = NODE_PROVISIONING_LIST_ITERATION_GET (0x03)							
Seq No							
Remaining Counter							

Seq No (8 bits) Refer to *Sequence number management*.

Remaining Counter (8 bits)

This field is used to iterate over the Provisioning List. The field indicates the remaining amount of
CC:0078.01.03.11.005 entries in the Provisioning List. This field MUST be in the range 0x01..0xFF.

CC:0078.01.03.11.006 This field MUST be set to 0xFF to start a new iteration. A supporting node MUST return the first entry and the actual amount of remaining entries in the returned report, i.e. If the Provisioning list has 3 elements the first response Remaining Count field MUST be set to 2.

CC:0078.01.03.11.007 A sending node MUST subsequently set this field to the returned value ”Remaining Count” value received in the returned Report if the ”Remaining Count” value is higher than 0x00. A supporting node MUST ignore this field if it is not set to the expected next iteration value.

CC:0078.01.03.11.008 This command MUST be ignored by a supporting node if this field is set to a value lower than 0xFF and no iteration has been started.

Refer to Section 5.2.7.11.1.

5.2.7.9 Node Provisioning List Iteration Report Command

This command is used to advertise the contents of an entry in the Provisioning List of the sending node.

Table 5.139: Node Provisioning List Iteration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_NODE_PROVISIONING (0x78)							
Command = NODE_PROVISIONING_LIST_ITERATION_REPORT (0x04)							
Seq No							
Remaining Count							
Reserved				DSK Length N			
DSK 1 (Optional)							
...							
DSK N (Optional)							
Meta Data Extension 1 (Optional)							
...							
Meta Data Extension M (Optional)							

Seq No (8 bits) Refer to *Sequence number management*.

Remaining Count (8 bits)

The field MUST indicate the remaining amount of entries in the Provisioning List iteration. This field
CC:0078.01.04.11.001 MUST be in the range 0x00..0xFE.

DSK Length (5 bits)

CC:0078.01.04.11.002 This field MUST indicate the length of the DSK field in bytes.

CC:0078.01.04.11.003 This field MUST be set to 0 or 16.

CC:0078.01.04.11.004 The value 0 MUST indicate that the Provisioning List of the sending node is empty or the Provisioning List Entry has been deleted after the start of the iteration.

CC:0078.01.04.11.005 The value 16 MUST indicate that the sending node advertises the DSK of a Provisioning List entry.

DSK (N bytes)

This field is used to advertise the DSK for the Provisioning List entry being advertised.

CC:0078.01.04.11.006 The length of this field (in bytes) MUST be according to the DSK Length field value. This field MUST be omitted if the DSK Length field is set to 0.

Meta Data Extension (M bytes)

This field is used to carry additional metadata associated to the Provisioning List entry.

CC:0078.01.04.13.001 This field MAY contain several extensions.

CC:0078.01.04.11.007 Each extension Type, Length and Value MUST comply with *Meta Data Extension Format* and [28].

CC:0078.01.04.13.002 A supporting node MAY set the critical flag to 0 even when advertising critical extensions.

CC:0078.01.04.11.008 If the DSK Length field is set to 0, this field MUST be omitted.

If the DSK Length field is not set to 0:

CC:0078.01.04.11.009 • A sending node MUST advertise the SmartStart Inclusion Setting extension (type 0x34)

CC:0078.01.04.11.00A • A sending node MUST advertise the Bootstrapping mode extension (type 0x36)

CC:0078.01.04.11.00C • A sending node MUST advertise the Network Status extension (type 0x37)

CC:0078.01.04.11.00B • A sending node MUST advertise all other extension data kept in the Provisioning List

5.2.7.10 Meta Data Extension Format

CC:0078.01.00.11.001 Each Meta Data extension MUST be parsed according to the following format:

Table 5.140: Meta Data Extension Format

7	6	5	4	3	2	1	0
Meta Data Type							critical
Length							
Value 1 (Optional)							
...							
Value L (Optional)							

Meta Data Type (7 bits)

This field is used to advertise the type of the data contained in the corresponding extension.

CC:0078.01.00.11.002 For the list of defined valid extensions, refer to [28]. Values not defined in [28] are reserved and MUST NOT be used by a sending node.

Critical (1 bit)

This field is used to advertise the criticality of the extension.

CC:0078.01.00.11.003 A supporting node MUST discard and ignore the entire command if this flag is set to '1' and the Meta Data Type field advertises a value that the node does not support.

CC:0078.01.00.12.001 A controlling node which controls only (i.e. does not support this Command Class) SHOULD keep the Provisioning List entry in its record even if this flag is set to '1' and the node does not know what the extension means.

CC:0078.01.00.11.004 If this flag is set to '0' and the Meta Data Type field advertises a value that the receiving node does not support, the actual extension MUST be ignored and left out the provisioning list entry.

CC:0078.01.00.11.005 In this case, a receiving node MUST continue processing of the encapsulation command after the discarded extension.

Length (8 bits)

CC:0078.01.00.11.006 This field MUST indicate the length of the corresponding Value field in bytes.

Value (L bytes)

CC:0078.01.00.11.007 This field MUST indicate the value of the Meta Data type being advertised in the extension.

CC:0078.01.00.11.008 The length of this field (in bytes) MUST be according to the corresponding Length field value .This field MUST be omitted if the corresponding Length field is set to 0.

CC:0078.01.00.11.009 The encoding of this field MUST be interpreted with the Meta Data Type field as defined in [28].

5.2.7.11 Usage and Frame Flows

5.2.7.11.1 Z/IP Client requesting the entire Node Provisioning list.

The frame flow for Z/IP client requesting the entire Provisioning List of a supporting node is shown in Figure 5.28.

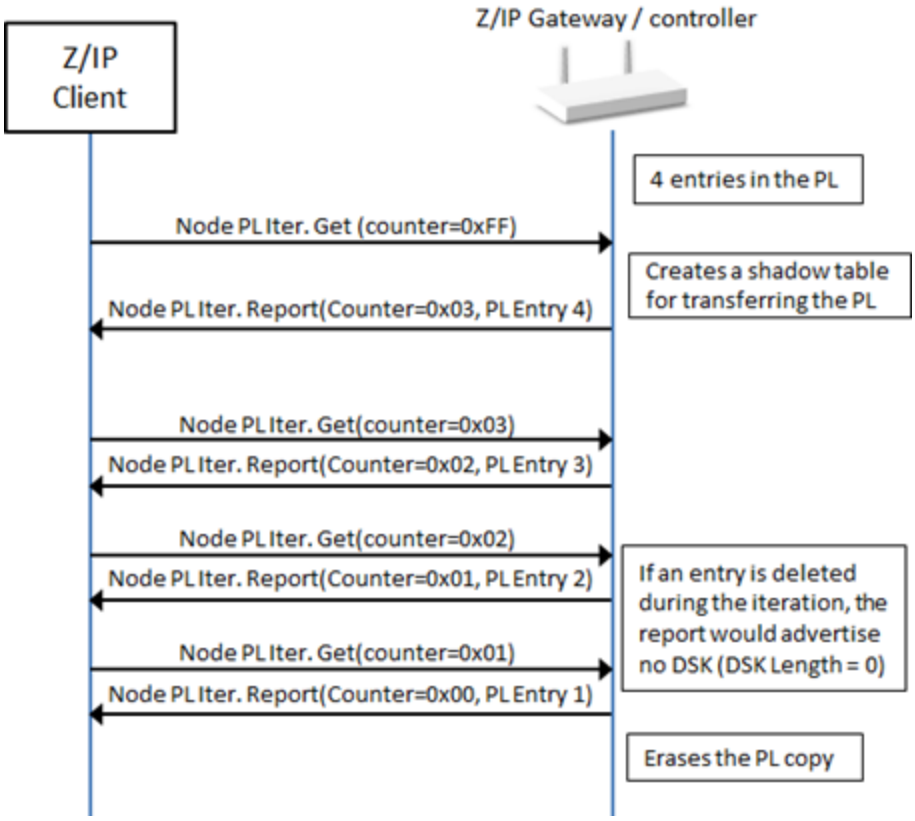


Figure 5.28: Reading the entire Node Provisioning List

5.2.8 Powerlevel Command Class, version 1

The Powerlevel Command Class defines RF transmit power controlling Commands useful when installing or testing a network. The Commands makes it possible for supporting controllers to set/get the RF transmit power level of a node and test specific links between nodes with a specific RF transmit power level.

NOTE: This Command Class is only used in an installation or test situation.

5.2.8.1 Powerlevel Set Command

This command is used to set the power level indicator value, which should be used by the node when transmitting RF, and the timeout for this power level indicator value before returning the power level defined by the application.

Table 5.141: Powerlevel Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_POWERLEVEL (0x73)							
Command = POWERLEVEL_SET (0x01)							
Power Level							
Timeout							

Power level (8 bits)

CC:0073.01.01.11.001 This field indicates the power level value that the receiving node MUST set. However, a supporting node MAY decide not to change its actual Tx configuration. In any case, the value received in this Command MUST be returned in a *Powerlevel Report Command* in response to a Powerlevel Get Command as if the power setting was accepted for the indicated duration.

CC:0073.01.01.11.002 This field MUST be encoded according to Table 5.142.

Table 5.142: Powerlevel Set::Power level encoding

Value	Description
0x00	NormalPower
0x01	minus1dBm
0x02	minus2dBm
0x03	minus3dBm
0x04	minus4dBm
0x05	minus5dBm
0x06	minus6dBm
0x07	minus7dBm
0x08	minus8dBm
0x09	minus9dBm

CC:0073.01.01.11.003 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Timeout value is ignored if Power level is set to normalPower.

Timeout (8 bits)

The time in seconds the node should keep the Power level before resetting to normalPower level. It is fundamental, that the timeout IS implemented and followed by the application, for keeping the network consistent. Valid values are 1-255 resulting in timeouts from 1 second to 255 seconds.

5.2.8.2 Powerlevel Get Command

This command is used to request the current power level value.

CC:0073.01.02.11.001 The Powerlevel Report Command MUST be returned in response to this command.

CC:0073.01.02.11.002 This command MUST NOT be issued via multicast addressing.

CC:0073.01.02.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.143: Powerlevel Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_POWERLEVEL (0x73)							
Command = POWERLEVEL_GET (0x02)							

5.2.8.3 Powerlevel Report Command

This command is used to advertise the current power level.

Table 5.144: Powerlevel Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_POWERLEVEL (0x73)							
Command = POWERLEVEL_REPORT (0x03)							
Power Level							
Timeout							

Power level (8 bits)

This value is the current power level indicator value in effect on the node.

CC:0073.01.03.11.001 This field MUST be encoded according to Table 5.142.

If the returned value is normalPower, the timeout value is ignored.

Timeout (8 bits)

The time in seconds the node has back at Power level before resetting to normal Power level.

5.2.8.4 Powerlevel Test Node Set Command

This command is used to instruct the destination node to transmit a number of test frames to the specified NodeID with the RF power level specified. After the test frame transmissions the RF power level is reset to normal and the result (number of acknowledged test frames) is saved for subsequent read-back. The result of the test may be requested with a Powerlevel Test Node Get Command.

CC:0073.01.04.12.001 A receiving node SHOULD return an unsolicited Powerlevel Test Node Report Command when it completed the Powerlevel test initiated by this command.

Table 5.145: Powerlevel Test Node Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_POWERLEVEL (0x73)							
Command = POWERLEVEL_TEST_NODE_SET (0x04)							
Test NodeID							
Power Level							
Test Frame Count (MSB)							
Test Frame Count (LSB)							

Test NodeID (8 bits)

The test NodeID that should receive the test frames.

A power level test will not work with a test NodeID which is either a sleeping or FLiRS node. A
CC:0073.01.04.12.002 controller SHOULD NOT initiate a powerlevel test towards sleeping or FLiRS nodes.

Power level (8 bits)

The power level indicator value to use in the test frame transmission.

CC:0073.01.04.11.001 This field MUST be encoded according to Table 5.142.

Test frame count (16 bits)

The Test frame count field contains the number of test frames to transmit to the Test NodeID. The first byte is the most significant byte. Valid Test frame count range is 1..65535.

5.2.8.5 Powerlevel Test Node Get Command

This command is used to request the result of the latest Powerlevel Test.

CC:0073.01.05.11.001 The Powerlevel Test Node Report Command MUST be returned in response to this command.

CC:0073.01.05.11.002 This command MUST NOT be issued via multicast addressing.

CC:0073.01.05.11.003 A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.146: Powerlevel Test Node Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_POWERLEVEL (0x73)							
Command = POWERLEVEL_TEST_NODE_GET (0x05)							

5.2.8.6 Powerlevel Test Node Report Command

This command is used to report the latest result of a test frame transmission started by the Powerlevel Test Node Set Command.

Table 5.147: Powerlevel Test Node Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_POWERLEVEL (0x73)							
Command = POWERLEVEL_TEST_NODE_REPORT (0x06)							
Test NodeID							
Status of operation							
Test frame acknowledged count (MSB)							
Test frame acknowledged count (LSB)							

Test NodeID (8 bits)

This field advertises the NodeID of the node, which is or has been under test.

CC:0073.01.06.11.001 If a test has been performed, this field MUST reflect the NodeID used in the last test initiated with the Powerlevel Test Node Set Command.

CC:0073.01.06.11.002 If no test has been performed, this field MUST be set to 0. In this case, the Status of operation and
CC:0073.01.06.13.001 Test frame acknowledged count fields MAY be ignored.

CC:0073.01.06.13.002 It is OPTIONAL to save the last Powerlevel test result in the NVM. If a node saves the last Powerlevel
CC:0073.01.06.13.003 test result in the volatile memory, it MAY set this field to 0 after going to sleep or losing power.

Status of operation (8 bits)

This field indicates the result of the last test initiated with the Powerlevel Test Node Set Command.
CC:0073.01.06.11.003 It MUST be encoded according to Table 5.148.

Table 5.148: Powerlevel Test Node Report::Status of Operation Encoding

Value	Description
0x00	Test Failed No frame was returned during the test
0x01	Test Success At least 1 frame was returned during the test
0x02	Test in Progress The test is still ongoing

CC:0073.01.06.11.004 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0073.01.06.13.004 A controller MAY return "Test Failed" for a non-existing NodeID without carrying the test. However,
CC:0073.01.06.11.005 it MUST return "Test in Progress" if carrying the test even if it knows that the test will fail.

Test frame acknowledged count (16 bits)

This field indicates the number of test frames transmitted, which the Test NodeID has acknowledged. The first byte is the most significant byte.

5.2.9 Z/IP Command Class, version 1 [OBSOLETED]

Warning: THIS COMMAND CLASS VERSION HAS BEEN OBSOLETED

CC:0023.01.00.11.001 New implementations MUST use the Z/IP Command Class Version 2.

5.2.10 Z/IP Command Class, version 2

The Z/IP Command Class is a special Command Class intended for encapsulation of Z-Wave commands in IP packets.

Z/IP Packets may be exchanged between IP hosts running over physical layers such as Ethernet or WiFi.

5.2.10.1 Security considerations

This Command Class is used to encapsulate Z-Wave Commands in an IP network. A Z/IP Gateway will forward some of the encapsulated Z-Wave commands into the Z-Wave Network.

The Z-Wave nodes may use encryption to protect the integrity of the Z-Wave network (refer to Security 0 and Security 2 Command Classes). IP nodes should always assume to be in a hostile network and support an encryption mechanism, such as DTLS.

If an IP node implements IP LAN security (DTLS or equivalent):

CC:0023.02.00.41.001
- Z/IP Packet received via secure IP channel MUST be accepted and a Z/IP Gateway MUST forward the optional Z-Wave Command in the Z-Wave network.
- Z/IP Packet received via non-secure IP channel:

CC:0023.02.00.41.002 - Z/IP Discovery Command Class MUST be accepted. - All other encapsulated Command Classes CC:0023.02.00.41.003 MUST be discarded.

If an IP node does not implement IP LAN security (DTLS or equivalent):

CC:0023.02.00.41.004
- All Z/IP Packet received via non-secure IP channel MUST be accepted.

CC:0023.02.00.43.001 If the Z/IP Command Class interface is used to communicate between applications within the same machine, the use of DTLS is OPTIONAL.

CC:0023.02.00.41.005 If the Z/IP Command Class interface is used to communicate between applications across an IP network, the use of DTLS is REQUIRED.

5.2.10.2 Interoperability considerations

CC:0023.02.00.32.001 Any Z-Wave Command Class SHOULD be sent encapsulated in a Z/IP Packet if a transmission takes place in an IP network.

CC:0023.02.00.31.001 Commands part of this Command Class MUST NOT be encapsulated in a Z/IP Packet Command.

5.2.10.3 Z/IP Packet Command

IP->UDP:4123->Z/IP Packet Command->Optional Z-Wave command

IP->UDP:41230->DTLS->Z/IP Packet Command->Optional Z-Wave command

CC:0023.02.02.11.001 A Z/IP Packet Command MUST be carried in a UDP packet, using destination port 4123 when no LAN security is used.

CC:0023.02.02.11.002 A Z/IP Packet Command MUST be carried in a UDP packet, using destination port 41230 when DLTS is used.

CC:0023.02.02.11.003 A node returning an answer to a Z/IP Packet (Ack/NAck Response) MUST swap the UDP source and destination ports.

The Z/IP Packet may carry a Z-Wave command or it may be used to communicate positive or negative acknowledgement for the delivery of another Z/IP Packet.

CC:0023.02.02.11.004 The Z/IP Packet MUST NOT be used for transmission between native Z-Wave nodes. The Z/IP Packet is intended for transport of encapsulated Z-Wave commands inside IP packets in an IP environment.

For that reason, normal Z-Wave MAC layer frame size limitations do not apply to this command, however Z-Wave MAC Layer frame size and Z-Wave Transport Service Command Class size limitations apply to the Z-Wave Command field.

Table 5.149: Z/IP Packet Command

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ZIP (0x23)</td></tr>
<tr><td colspan="8">Command = COMMAND_ZIP_PACKET (0x02)</td></tr>
<tr><td rowspan="2">Ack Request</td><td rowspan="2">Ack Response</td><td rowspan="2">NAck Response</td><td colspan="3">(NAck flags)</td><td colspan="2" rowspan="2">Reserved</td></tr>
<tr><td>Waiting</td><td>Queue Full</td><td>Option Error</td></tr>
<tr><td>Header ext. included</td><td>Z-Wave Cmd included</td><td>More Information</td><td>Secure Origin</td><td colspan="4">Reserved</td></tr>
<tr><td colspan="8">Seq No</td></tr>
<tr><td>Res</td><td colspan="7">Source End Point</td></tr>
<tr><td>Bit address</td><td colspan="7">Destination End Point</td></tr>
<tr><td colspan="8">Header extension 1 (Optional)</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">Header extension N (Optional)</td></tr>
<tr><td colspan="8">Z-Wave Command 1 (Optional)</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">Z-Wave Command M (Optional)</td></tr>
</table>

CC:0023.02.02.11.005 A receiving node MUST inspect the header flags in order to determine the offset to use for accessing the optional fields. If the packet contains invalid data, e.g. the ACK_RES and NACK_RES bits are both set to 1 or if the length of the extended header does not add up, a receiving node MUST ignore the packet.

Ack Request (1 bit)

CC:0023.02.02.11.006 This flag signals that the receiving node MUST return an Ack or NAck message in response to the actual Z/IP Packet.

CC:0023.02.02.11.007 This field MUST be encoded according to Table 5.150.

Table 5.150: Z/IP Packet::Ack Request Flag encoding

Value	Description
'1'	Return Ack or NAck
'0'	No confirmation needed

CC:0023.02.02.11.008 If this flag is set to 1, the Z/IP Packet MUST contain a Z-Wave Command. A receiving node MUST discard the packet if this flag is set to 1 but no Z-Wave Command is included.

This field is intended for delivery acknowledgement for Z-Wave Commands encapsulated in Z/IP packets.

CC:0023.02.02.12.001 Z-Wave link-level acknowledgement SHOULD always be used between Z-Wave nodes when Z-Wave is used as link layer.

CC:0023.02.02.11.009 A Z/IP Gateway MUST return a "NAck+Waiting" indication no later than 200ms after receiving an Ack Request if the Z-Wave Command is still being processed or pending delivery.

CC:0023.02.02.13.001 A sending node that has requested an Ack and has waited for more than 300ms without receiving an Ack or NAck indication MAY conclude that the Z-Wave Command is lost and retransmit the Z/IP Packet.

CC:0023.02.02.11.00A In case of successful delivery to a Z-Wave node, a Z/IP Packet with Ack Response indication MUST be returned by the Z/IP Gateway upon reception of the Z-Wave Ack.

CC:0023.02.02.11.00B In case of successful delivery to a Z/IP node, the Z/IP node itself MUST return a Z/IP Packet with Ack Response indication.

Ack Response (1 bit)

CC:0023.02.02.11.00C This flag MUST be used to indicate that the destination has received the Z-Wave Command encapsulated in a preceding Z/IP packet.

CC:0023.02.02.11.00D This field MUST NOT be interpreted as a confirmation that the destination has accepted the application command carried in the Z-Wave Command field.

CC:0023.02.02.11.00E This field MUST be encoded according to Table 5.151.

Table 5.151: Z/IP Packet::Ack Response Flag encoding

value	Description
'1'	This Z/IP Packet acknowledges a preceding packet that requested an Ack Response
'0'	This Z/IP Packet does not acknowledge a preceding packet that requested an Ack Response. CC:0023.02.02.11.00F A receiving node MUST inspect the NAck Response field

CC:0023.02.02.11.010 If this field is set to 1, the *Seq No* field value MUST be the same as the Z/IP packet being acknowledged.

NAck Response (1 bit)

CC:0023.02.02.11.011 This flag MUST be used to indicate that the destination has not (yet) received the Z-Wave Command encapsulated in a preceding Z/IP Packet.

CC:0023.02.02.11.012 This field MAY be set to 1 by intermediate nodes such as a Z/IP Gateway. This field MUST be encoded according to Table 5.152.

Table 5.152: Z/IP Packet::NAck Response Flag encoding

value	Description
'1'	This Z/IP Packet negatively acknowledges a preceding packet that requested an Ack Response. (i.e. the Z-Wave Command was not delivered to the destination) CC:0023.02.02.11.013 A receiving node MUST inspect the NAck flags fields.
'0'	This field and the *NAck flags* fields may be ignored.

CC:0023.02.02.11.014 If this field is set to 1, the *Seq No* field value MUST be the same as the Z/IP packet being negatively acknowledged.

If this field is set to 1 but none of the *NAck flags* field is set to 1, the Z-Wave Command was lost but no specific reason is provided.

(NAck flags) Waiting (1 bit)

CC:0023.02.02.12.002 This flag is a companion flag to the *NAck Response* flag. It SHOULD be inspected only if the *NAck Response* flag field is set to 1 and SHOULD be ignored otherwise.

CC:0023.02.02.11.015 This flag MUST be ignored if the *Queue Full* flag is set to 1.

CC:0023.02.02.11.016 This flag MUST be used to indicate that the destination may have a long response time. i.e. the Z-Wave Command has not timed out yet and is pending delivery. This field MUST be encoded according to Table 5.153.

Table 5.153: Z/IP Packet::Waiting Flag encoding

value	Description
'1'	Waiting: the preceding Z/IP Packet encapsulated Z-Wave Command is not yet delivered to the destination and delivery will be attempted later on
'0'	Not waiting: the preceding Z/IP Packet encapsulated Z-Wave Command will not be delivered later on.

CC:0023.02.02.13.002 An "Expected Delay" Option MAY be returned by a Z/IP Gateway, indicating how long it should take
CC:0023.02.02.12.003 before the final delivery acknowledgment status is known, refer to Section 5.2.13.2.1. A sending node
CC:0023.02.02.11.017

SHOULD use this information to provide better user responsiveness. A default value of 90 seconds MUST be used by the sending node if no "Expected delay" Option is provided.

A Z/IP "NAck+Waiting" indication is returned for every packet that is queued up. If a sending node triggers to queue up three Z-Wave Commands, it will receive a "NAck+Waiting" indication after each packet. It may be desirable to queue up three configuration commands if the intention is to perform a few configuration changes and allow a battery node to return to sleep.

CC:0023.02.02.12.004 If a sending node wants to transfer larger amounts of data or commands, e.g. probing capabilities or downloading a new firmware image, it is RECOMMENDED to send a single Z/IP Packet using the More Information field to make the destination node stay awake. When a Z/IP Ack Response indication is returned to the sending node, it can start transferring packets at a higher rate.

CC:0023.02.02.11.018 A Z/IP Gateway MUST return a "NAck+Waiting" indication no later than 200ms after receiving an Ack Request indication if the Z-Wave Command is still pending delivery.

CC:0023.02.02.11.019 If a message has been delayed for more than 60 seconds, an intermediate receiver, such as a Z/IP Gateway, MUST transmit a new "NAck+Waiting" indication every 60 seconds to let the sending node know that it is still operational.

CC:0023.02.02.13.003 A sending node waiting for more than 90 seconds after receiving a "NAck+Waiting" indication MAY conclude that the Z-Wave Command is lost and retransmit a new Z/IP Packet.

CC:0023.02.02.11.01A A Z/IP Gateway issuing a "NAck+Waiting" indication MUST subsequently issue an Ack Response indication when the Z-Wave Command has been delivered.

CC:0023.02.02.11.01B A Z/IP Gateway MUST return a Z/IP *NAck Response* indication if the Z-Wave Command delivery is aborted or not successful.

(NAck flag) Queue Full (1 bit)

CC:0023.02.02.12.005 This flag is a companion flag to the NAck Response flag. It SHOULD be inspected only if the NAck Response flag is set to 1 and SHOULD be ignored otherwise.

CC:0023.02.02.11.01C This flag MUST be used by a Z/IP Gateway for packets targeting battery nodes, in a busy network, during bulk data transfers or route re-discovery.

CC:0023.02.02.13.004 This flag MAY also be returned for always listening Z-Wave destinations.

CC:0023.02.02.11.01D A sending node MUST wait for at least 10 seconds before re-transmitting a new Z/IP Packet.

CC:0023.02.02.11.01E This flag MUST be returned by a Z/IP Gateway if there is no more room in the queue system used for delivering Commands into the Z-Wave network. This field MUST be encoded according to Table 5.154.

Table 5.154: Z/IP Packet::Queue Full Flag encoding

value	Description
'1'	Queue is full: the preceding Z/IP Packet Command is discarded and will not be delivered to the destination
'0'	Queue OK

(NAck flag) Option Error (1 bit)

CC:0023.02.02.12.006 This flag is a companion flag to the *NAck Response* flag. It should only be inspected if the *NAck Response* flag is set to 1 and SHOULD be ignored otherwise.

CC:0023.02.02.11.01F This flag MUST be set to 1 if a critical option is not recognized by the receiving node and the entire Z/IP Packet was discarded.

CC:0023.02.02.11.020 This flag MUST NOT be set to 1 if an elective option is not recognized by the receiving node. Elective options MUST be silently ignored by a receiving node.

Table 5.155: Z/IP Packet::Option Error Flag encoding

value	Description
'1'	Option Error: A critical extension was not understood and the entire Z/IP Packet was discarded
'0'	(no error)

CC:0023.02.02.12.007 A node setting this flag to 1 SHOULD include the offending Option in the "NAck+OptionError" indication returned to the originating node.

CC:0023.02.02.11.021 A node receiving a "NAck+OptionError" indication MUST NOT process the Z/IP Packet Options in the Header Extension field as it is only included for debugging purposes.

Header extension Included (1 bit)

This flag is used to indicate that a *Header Extension* field is included in the Z/IP Packet. Refer to the *Header Extension* field description below.

CC:0023.02.02.11.022 This flag MUST be encoded according to Table 5.156.

Table 5.156: Z/IP Packet::Header Extension Included Flag encoding

value	Description
'1'	Header Extension field MUST be included in the Z/IP Packet
'0'	Header Extension field MUST NOT be included in the Z/IP Packet

Reserved

CC:0023.02.02.11.023 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Z-Wave Command Included (1 bit)

This flag is used to indicate that a *Z-Wave Command* field is included in the Z/IP packet. Refer to the *Z-Wave Command* field description below.

CC:0023.02.02.11.024 This flag MUST be encoded according to Table 5.157.

Table 5.157: Z/IP Packet::Z-Wave Command Included Flag encoding

value	Description
'1'	Z-Wave Command field MUST be included in the Z/IP Packet
'0'	Z-Wave Command field MUST NOT be included in the Z/IP Packet

CC:0023.02.02.11.025 If a Z/IP Packet is received with payload length = 0 and the "Z-Wave command included" bit set to 1, a receiving node MUST treat the Z/IP Packet as if the "Z-Wave command included" bit was set to 0.

CC:0023.02.02.11.04A If receiving a Z/IP Command with the Ack Request field set to 1 and no *Z-Wave Command included*, a Z/IP Gateway MUST issue a NOP frame to the destination and return a Z/IP Packet with Ack Response if the destination acknowledged the NOP frames.

More Information (1 bit)

This flag is used to indicate the Z/IP Gateway that it should prevent a sleeping node from returning to sleep during the next minute.

CC:0023.02.02.11.026 This flag MUST indicate that more Z/IP Packets with Z-Wave Commands will be subsequently transmitted. A sending node knowing that it will be sending more commands to the destination node
CC:0023.02.02.13.005 MAY set this flag to 1.

CC:0023.02.02.11.027 This flag MUST be encoded according to Table 5.158.

Table 5.158: Z/IP Packet::More Information Flag encoding

value	Description
'1'	The Z/IP Gateway should keep the sleeping node awake
'0'	The Z/IP Gateway should put the sleeping node to sleep

Secure Origin (1 bit)

This field indicates if the Z-Wave Command is to be treated securely.

CC:0023.02.02.11.028 The value 1 MUST indicate that the Z-Wave Command MUST be treated securely (i.e. it was or will be sent using encryption in the Z-Wave network).

CC:0023.02.02.11.029 The value 0 MUST indicate that the Z-Wave Command MUST NOT be treated securely. (i.e. it was or will be sent non-securely in the Z-Wave network)

CC:0023.02.02.11.02A A Z/IP Gateway forwarding the contents of an encrypted Z-Wave frame MUST set the Secure Origin flag to '1'.

CC:0023.02.02.11.02B A Z/IP Gateway forwarding the contents of a non-encrypted Z-Wave frame MUST set the Secure Origin flag to '0'.

CC:0023.02.02.11.02C A Z/IP Gateway MUST inspect the Secure Origin flag when forwarding a Z-Wave Command contained in a Z/IP Packet from an IP network to a Z-Wave network.

CC:0023.02.02.11.02D A Z/IP Gateway MUST NOT use secure communication via Z-Wave if this flag is set to '0' and MUST use secure communication via Z-Wave if this flag is set to '1'.

Seq No (8 bits)

This field is used to identify Z/IP Packet duplicates or retransmissions.

CC:0023.02.02.11.02E This field MUST carry a unique sequence number. Each sequence number MUST be generated from an 8-bit counter that is incremented by 1 whenever a new sequence number is generated. When a node powers up, the sequence counter MUST be initialized to a random value.

CC:0023.02.02.13.006 The counter MAY be shared with other Command Classes.

CC:0023.02.02.11.02F Retransmitted Z/IP packets MUST carry the same value as the original Z/IP Packet. A Z/IP Ack or NAck packet MUST carry the same Seq No value as the Z/IP packet being acknowledged.

CC:0023.02.02.11.030 Multiple Z/IP Packets may be received in case of link-layer retransmissions. Z/IP Packet duplicates MUST be ignored by a receiving node.

Source End Point (7 bits)

This field is used to indicate the originating end point from which the Z-Wave Command was sent.

CC:0023.02.02.11.031 This field MUST be in the range 0..127.

The Source End Point value 0 represents the Root Device. Refer to the Multi Channel Command Class for more details.

Bit address (1 bit)

This field is used to advertise if the *destination End Point* field is specified as a bit mask.

CC:0023.02.02.11.032 The value 0 MUST indicate that the Destination End Point field is specified as a single End Point.

CC:0023.02.02.11.033 The value 1 MUST indicate that the Destination End Point field is specified as a bit mask. Only destination end points 1..7 are bit addressable.

CC:0023.02.02.11.034 Bit addressing MUST NOT be used if the encapsulated command is a request (requiring a reply from the destination).

Destination End Point (7 bits)

This field is used to indicate the destination End Point of the actual Z-Wave Command.

CC:0023.02.02.11.035

If the *Bit address* field is set to 0, this field MUST carry a single End Point identifier value in the range 0..127.

CC:0023.02.02.11.036 The value 0 MUST represent the Root Device. Values in the range 1..127 MUST represent an actual End Point.

CC:0023.02.02.11.037 If the Bit address field is set to 1, this field MUST use the following encoding:

- Bit 0 in the Destination End Point indicates if End Point 1 is a destination
- Bit 1 in the Destination End Point indicates if End Point 2 is a destination
- …

CC:0023.02.02.11.038 The bit value 0 MUST be used to advertise that the corresponding End Point is not a destination.

The bit value 1 MUST be used to advertise that the corresponding End Point is a destination.

Header Extension (variable)

CC:0023.02.02.11.039 This field is used for advertising additional Z/IP Packet Options that are necessary in certain cases. A Z/IP node MUST support and parse this field.

CC:0023.02.02.11.03A This field MUST be omitted if the *Header extension Included* field is set to 0.

CC:0023.02.02.11.03B If the *Header Extension Included* field is set to 1, this field MUST be formatted as follows:

Table 5.159: Header Extension Field

7	6	5	4	3	2	1	0
Header Extension Length							
Z/IP Packet Option 1, 1							
…							
Z/IP Packet Option P, 1							
…							
Z/IP Packet Option 1, N							
…							
Z/IP Packet Option P, N							

Header Extension Length (1 byte)

CC:0023.02.02.11.03C This field MUST indicate the combined length in bytes of the Z/IP Header Extension Length and all the Z/IP Packet Options included in the Z/IP Header Extension.

CC:0023.02.02.11.03D This field MUST be in the range 1..255. The length of the Header Extension field cannot exceed 255 bytes.

Z/IP Packet Option (variable)

CC:0023.02.02.11.03E Each Z/IP Packet Option MUST be treated parsed as a block using the following format:

Table 5.160: Z/IP Packet Option (variable) Field

7	6	5	4	3	2	1	0
Critical	Option Type						
Option Length							
Option Data 1 (Optional)							
…							
Option Data L (Optional)							

CC:0023.02.02.11.03F A receiving node MUST accept receiving supported options in any order.

(Z/IP Packet Option) Critical (1 bit)

This field is used to indicate if the whole Z/IP Packet Command must be ignored if the option is not recognized by the receiving node.

CC:0023.02.02.11.040

The value 0 MUST indicate that the option is elective and a receiving node MUST ignore this option only and continue processing the frame if the option is not recognized.

CC:0023.02.02.11.041 The value 1 MUST indicate that the option is critical and a receiving node MUST discard the entire Z/IP Packet Command and return a Z/IP Packet Command with the Option Error flag set to 1 if the option is not recognized.

CC:0023.02.02.11.042 An option MUST be considered as recognized even if:

- The Option Length field is set to a greater value than expected
- Reserved fields in the Option Data field are not set to 0.

CC:0023.02.02.11.043 An option MUST NOT be considered as recognized when:

- The Option Type field is set to an unknown value.
- A field value in the Option Data field value which is out of expected range or seems to be using reserved values.

(Z/IP Packet Option) Option Type (7 bits)

This field is used to indicate which format to use for parsing the corresponding *Option Data* field. The list of defined Option Type is specified in Section 5.2.13.2.

CC:0023.02.02.11.044 A receiving node MUST accept supported Z/IP Packet Options in any order.

(Z/IP Packet Option) Option Length (8 bits)

CC:0023.02.02.11.045 This field MUST indicate the length of the corresponding Option Data field in bytes.

(Z/IP Packet Option) Option Data (L bytes)

CC:0023.02.02.11.046 This field is used to carry the actual Option data. It MUST be parsed and interpreted using the corresponding *Option Type* field value.

CC:0023.02.02.11.047 The size of this field in bytes MUST be according the corresponding Option Length field. This field MUST be omitted if the corresponding Option Length field is set to 0.

Z-Wave Command (M bytes)

CC:0023.02.02.11.048 This field carries a complete Z-Wave command. This field MUST be formatted according to the corresponding command class as defined in Section 4, Section 3, Section 2 and in this specification.

CC:0023.02.02.12.008 A sending Z/IP client SHOULD be aware that this command will be transmitted over a Z-Wave network and therefore respect the Z-Wave Command length limitations. A Z/IP Client SHOULD limit the length of this field to 45 bytes for non-S2 destination nodes and 117 bytes for S2 supporting nodes.

CC:0023.02.02.11.049 This field MUST be omitted if the *Z-Wave Cmd Included* field is set to 0.

5.2.11 Z/IP Command Class, version 3

The Z/IP Packet Command Class, version 3 adds the support of the Encapsulation Format Info Option to the Z/IP Packet Option types.

5.2.11.1 Compatibility considerations

Z/IP Packet Command Class, version 3 is backwards compatible with the Z/IP Packet Command Class, version 2.

CC:0023.03.00.21.001 A device supporting Z/IP Packet Command Class, version 3 MUST support Z/IP Packet Command Class, version 2.

CC:0023.03.00.21.002 All fields and commands not described in this version MUST remain unchanged from version 2.

CC:0023.03.00.21.003 A node supporting the Z/IP Packet Command Class, version 3 MUST support the Encapsulation Format Information Option and respect the requirements specified in *Encapsulation Format Information Option*.

5.2.11.2 Z/IP Packet Command

CC:0023.03.02.11.001 The frame structure MUST remain unchanged from version 2.

Secure Origin (1 bit)

This field is superseded by the Encapsulation Format Information Option.

CC:0023.03.02.11.002 A version 3 sending node MUST use the Encapsulation Format Information Option with a version 3 receiving node.

CC:0023.03.02.11.003 This field MUST be ignored by a receiving node if an Encapsulation Formation Information Z/IP Option is included in the Z/IP Packet.

5.2.12 Z/IP Command Class, version 4

The Z/IP Packet Command Class, version 4 adds the support of a Keep Alive command in order to prevent the closure of a Z/IP DTLS session and introduces new Z/IP Packet Options.

5.2.12.1 Compatibility considerations

Z/IP Packet Command Class, version 4 is backwards compatible with the Z/IP Packet Command Class, version 3.

CC:0023.04.00.21.001 A device supporting Z/IP Packet Command Class, version 4 MUST support Z/IP Packet Command Class, version 3.

CC:0023.04.00.21.002 All fields and commands not described in this version MUST remain unchanged from version 3.

CC:0023.04.00.21.003 A node supporting the Z/IP Packet Command Class, version 4 MUST respect the requirements specified in *Z-Wave Multicast Addressing Option*.

The Z/IP Keep Alive Command is introduced in this version in order to prevent a DTLS session to time out between a Z/IP Client and server. The default DTLS timeout configured in Z/IP deployments is 60 seconds, i.e. a peer will close the DTLS connection if no command is sent or received in 60 seconds.

The Installation and Maintenance Report Option is extended with new TLVs and a new option is added to indicate a receiving client the addressing method that has been used on the Z-Wave network.

5.2.12.2 Z/IP Keep Alive Command

This command is used to as a Keep Alive probe for a DTLS session between two IP nodes.

CC:0023.04.03.21.001 This command SHOULD be issued at a minimum interval of 25 seconds and at a maximum interval of 55 seconds after the last sent or received command in order to prevent session closure.

Table 5.161: Z/IP Keep Alive Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP (0x23)							
Command = COMMAND_ZIP_KEEP_ALIVE (0x03)							
Ack Request	Ack Response	Reserved					

Reserved

CC:0023.04.03.11.001 This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Ack Request (1 bit)

This field is used by a sending node to request an acknowledgement for this command from a receiving node.

CC:0023.04.03.11.002 The value 1 MUST indicate that an acknowledgement is requested.

The value 0 MUST indicate that an acknowledgement is not requested.

CC:0023.04.03.11.003 If this flag is set to 1, a receiving node MUST return a Z/IP Keep Alive Command with the *Ack Response* flag set to 1.

CC:0023.04.03.11.004 This field MUST NOT be set to 1 if the Ack Response field is set to 1.

Ack Response (1 bit)

This field is used by a node to acknowledge that it received a Z/IP Keep Alive Command with the *Ack Request* flag set to 1.

CC:0023.04.03.11.005 The value 1 MUST indicate that this command is an acknowledgement for a received Z/IP Keep Alive Command.

The value 0 MUST indicate that this command is not an acknowledgement

CC:0023.04.03.11.006 This field MUST NOT be set to 1 if the Ack Request field is set to 1.

5.2.13 Z/IP Command Class, version 5

5.2.13.1 Compatibility considerations

Z/IP Packet Command Class, version 5 is backwards compatible with the Z/IP Packet Command Class, version 4.

A device supporting Z/IP Packet Command Class, version 5 MUST support Z/IP Packet Command Class, version 4.

All fields and commands not described in this version MUST remain unchanged from version 4.

5.2.13.2 List of defined Z/IP Packet Options

The list of defined Z/IP Packet Option Types is listed in Table 52 and each type is defined in the following subsections.

CC:0023.00.02.11.001 A sending node using a given option MUST support as a minimum the version indicated in Table 52 for the Z/IP Command Class.

Table 5.162: Z/IP Packet Option types

Option Type	Type value	Class	Version
Expected delay	1	Elective	2
Installation and Maintenance Get	2	Elective	2
Installation and Maintenance Report	3	Elective	2
Encapsulation Format Information	4	Critical	3
Z-Wave Multicast Addressing	5	Elective	4

CC:0023.00.02.11.002 All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

CC:0023.00.02.11.003 A receiving node MUST accept supported options in any order.

5.2.13.2.1 Expected Delay Option

This option is used to advertise an expected delay when issuing a Z/IP Packet command with "NAck+Waiting" indication.

Table 5.163: Expected Delay Option

7	6	5	4	3	2	1	0
Critical = 0	Option Type = ZIP_OPTION_EXPECTED_DELAY = 1						
Option Length = 3							
Seconds 1 (MSB)							
Seconds 2							
Seconds 3 (LSB)							

Critical (1 bit)

CC:0023.00.02.11.004 The Critical field MUST be set to 0.

Option Type (7 bits)

CC:0023.00.02.11.005 The Option Type field MUST be set to 0x01 to indicate the Expected Delay Option.

Option Length (8 bits)

CC:0023.00.02.11.006 The Option Length field MUST indicate the length of the Seconds field.

Seconds (24 bits)

CC:0023.00.02.11.007

The Seconds field MUST indicate the expected time in seconds before issuing a new Z/IP Packet Command with a new status.

5.2.13.2.2 Installation and Maintenance Get Option

This option is used to request a receiving node to return a Z/IP Packet Command containing the Installation and maintenance Report Option.

CC:0023.00.02.11.008 In order to trigger an Installation and Maintenance Report to be returned, a sending node MUST:

- Add this Option in the Z/IP Packet Command
- Add a Z-Wave Command in the Z/IP Packet Command (NOP Command Class MAY be used if the sending node does not have any other Z-Wave Command to transmit)
- Set the Ack Request flag to 1 in the Z/IP Packet Command

CC:0023.00.02.11.009 A receiving node MUST:

- Return a Z/IP Packet Command containing the Installation and Maintenance Report Z/IP Option after the transmission of the contained Z-Wave Command to the destination.
- If returned, the Installation and Maintenance Report Z/IP Option MUST advertise the statistics associated to the Z-Wave Command transmission.

CC:0023.00.02.13.001 A receiving MAY ignore the request for an Installation and Maintenance Report if it does not support this Option.

Table 5.164: Installation and Maintenance Get Option

7	6	5	4	3	2	1	0
Critical = 0	Option Type = INSTALLATION_MAINTENANCE_GET = 2						
Option Length = 0							

5.2.13.2.3 Installation and Maintenance Report Option

This option is used to advertise Z-Wave transmission data about the communication between the Z/IP Gateway and a Z-Wave device in the network.

The Installation and Maintenance Report Option is used for data relating to the transmission of an actual frame. Statistical data may be accessed via the Network Management Installation and Maintenance Command Class.

Table 5.165: Installation and Maintenance Report Option

7	6	5	4	3	2	1	0
Critical = 0	Option Type = INSTALLATION_MAINTENANCE_REPORT = 3						
Option Length							
IME - Type 1							
IME - Length 1							
IME - Value 1, 1							
...							
IME - Value L, 1							
...							
IME - Type N							
IME - Length N							
IME - Value 1, N							
...							
IME - Value L, N							

Critical (1 bit)

CC:0023.00.02.11.00A The Critical field MUST be set to 0.

Option Length (1 byte)

CC:0023.00.02.11.00B This field MUST indicate the combined length (in bytes) of the following IME-TLV fields.

IME - Type / Length / Value (TLV) (variable)

CC:0023.00.02.11.00C This field is used to carry values advertising Z-Wave transmission statistics. Each TLV block MUST be encoded according to one of the Types defined in Table 53 and in the following subsections.

CC:0023.00.02.11.00D A sending node using a given TLV MUST support as a minimum the version indicated in Table 53 for the Z/IP Command Class.

CC:0023.00.02.13.002 The Z/IP Gateway MAY send any combination of the IME TLVs when using this Z/IP Packet Option.

Table 5.166: Z/IP Packet::IME-Type/Length/Value encoding

IME - Type	Name	IME - Length	Version
0x00	Route Changed	1 byte	2
0x01	Transmission Time (TT)	2 bytes	2
0x02	Last Working Route (LWR)	5 bytes	2
0x03	Incoming RSSI	5 bytes	4
0x04	ACK channel	1 byte	4
0x05	Transmit channel	1 byte	4
0x06	Routing scheme	1 byte	4
0x07	Routing attempts	1 byte	4
0x08	Last failed link	2 bytes	4
0x09	Tx Power	2 bytes	5
0x0A	Measured Noise Floor	2 bytes	5
0x0B	Outgoing RSSI	5 bytes	5

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

5.2.13.2.4 Route Changed (3 bytes)

Table 5.167: Route Changed

7	6	5	4	3	2	1	0
IME - Type = 0x00							
IME - Length = 1							
IME - Value = Route Changed							

Route Changed (8 bits)

This field is used to indicate if the last working route was changed for the current transmission.

CC:0023.00.02.11.00E If the last working route was changed, this field MUST be set to 0x01.

If the last working route was not changed, this field MUST be set to 0x00.

5.2.13.2.5 Transmission Time (4 bytes)

Table 5.168: Transmission Time

7	6	5	4	3	2	1	0
IME - Type = 0x01							
IME - Length = 2							
IME - Value = Transmission Time 1 (MSB)							
IME - Value = Transmission Time 2 (LSB)							

Transmission Time (16 bits)

This field is used to indicate the time it took to send the command until the reception of an Ack.
CC:0023.00.02.11.00F The value MUST be encoded using unsigned representation and MUST be specified using the ms (milliseconds) unit.

5.2.13.2.6 Last Working Route (7 bytes)

Table 5.169: Last Working Route

7	6	5	4	3	2	1	0
IME - Type = 0x02							
IME - Length = 5							
IME - Value = Repeater 1							
IME - Value = Repeater 2							
IME - Value = Repeater 3							
IME - Value = Repeater 4							
IME - Value = Speed							

This TLV is used to advertise the last used Working Route. If multiple Last Working Routes exist,
CC:0023.00.02.11.010 this MUST be the one used to transmit the frame.

Repeater 1-4 (4 bytes)

This field contains the NodeID of the repeaters used for the last working route.

CC:0023.00.02.11.011 The value 0 MUST indicate that the actual repeater was not used.

Values in the range 1..232 MUST indicate an actual NodeID used as repeater.

CC:0023.00.02.11.012 The first Repeater byte set to 0 MUST indicate that no more repeaters were used for the transmission. If the first Repeater byte is set to 0, it means that the Last Working Route (LWR) is a direct transmission.

Speed (8 bits)

CC:0023.00.02.11.013 This field is used to advertise the transmission speed used to reach the destination node. This field MUST be encoded according to Table 5.170.

Table 5.170: IME Speed Encoding

Value	Protocol	Speed
0x01	Z-Wave	9.6 kbit/sec
0x02	Z-Wave	40 kbit/sec
0x03	Z-Wave	100 kbit/sec
0x04	Z-Wave Long Range	100 kbit/sec

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Incoming RSSI (7 bytes)

Table 5.171: Incoming RSSI

7	6	5	4	3	2	1	0
IME - Type = 0x03							
IME - Length = 5							
IME - Value = RSSI hop 1							
IME - Value = RSSI hop 2							
IME - Value = RSSI hop 3							
IME - Value = RSSI hop 4							
IME - Value = RSSI hop 5							

The IME - Values advertise the RSSI value measured in the incoming direction (back towards the source of the message).

RSSI Hop (5 bytes)

CC:0023.00.02.11.014 The RSSI values MUST be encoded as using signed representation in the dBm unit and according to Table 5.172.

Table 5.172: RSSI Encoding

Value (signed)	Description
0x7F (127)	RSSI_NOT_AVAILABLE This value is returned for unused hops or if no RSSI measurement is available.
0x7E (126)	RSSI_MAX_POWER_SATURATED This value is returned if the measured RSSI is above the maximum power.
0x7D (125)	RSSI_BELOW_SENSITIVITY This value is returned if the measured RSSI is below the receiver's sensitivity
0x7E…0xE1 (124..-31)	Reserved
0xE0 (-32)	-32 dBm
0xDF (-33)	-33 dBm
…	…
0x80 (-128)	-128 dBm

5.2.13.2.7 ACK channel (3 bytes)

Table 5.173: ACK Channel

7	6	5	4	3	2	1	0
IME - Type = 0x04							
IME - Length = 1							
IME - Value = ACK channel							

ACK channel (8 bits)

This value reports the RF channel on which the ACK for this frame was received.

5.2.13.2.8 Transmit channel (3 bytes)

Table 5.174: Transmit Channel

7	6	5	4	3	2	1	0
IME - Type = 0x05							
IME - Length = 1							
IME - Value = Transmit channel							

Transmit channel (8 bits)

This value reports the RF channel on which the Z-Wave Command was transmitted.

5.2.13.2.9 Routing scheme (3 bytes)

Table 5.175: Routing Scheme

7	6	5	4	3	2	1	0
IME - Type = 0x06							
IME - Length = 1							
IME - Value = Routing scheme							

Routing scheme (8 bits)

This value reports the routing scheme that was used to find the successful route for delivering the Z-Wave Command.

CC:0023.00.02.11.015 The Routing scheme value MUST encoded according to Table 5.176.

Table 5.176: Routing Scheme IME::Routing Scheme encoding

Value	Description
0x00	Idle
0x01	Direct transmission (no routing)
0x02	Application static route
0x03	Last working route
0x04	Next to last working route
0x05	Return route or controller auto route
0x06	Direct resort
0x07	Explorer frame

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

5.2.13.2.10 Routing attempts (3 bytes)

Table 5.177: Routing Attempts

7	6	5	4	3	2	1	0
IME - Type = 0x07							
IME - Length = 1							
IME - Value = Routing attempts							

Routing attempts (8 bits)

This TLV reports the number of routing attempts that were performed before successfully delivering the Z-Wave Command.

5.2.13.2.11 Failed link (4 bytes)

Table 5.178: Failed Link

7	6	5	4	3	2	1	0
IME - Type = 0x08							
IME - Length = 2							
IME - Value = Failed Link Neighbor NodeID 1							
IME - Value = Failed Link Neighbor NodeID 2							

Failed Link Neighbor NodeID (16 bits)

This TLV is used if the transmission of the Z-Wave Command failed. The value reports the neighbor NodeIDs of the failing link in the last attempted route.

CC:0023.00.02.12.001 If the last node failed, Failed Link Neighbor NodeID 2 SHOULD be set to 0x00.

5.2.13.2.12 Tx Power (2 bytes)

This TLV is used to report the Tx Power (in dBm) used during the transmissions with the remote node.

Table 5.179: Tx Power

7	6	5	4	3	2	1	0
IME - Type = 0x09							
IME - Length = 2							
IME - Value = Local Node Tx Power							
IME - Value = Remote Node Tx Power							

Local Node Tx Power (8 bits)

This field is used to indicate the Tx Power (in dBm) used for the transmission from the local node.

This field MUST be encoded as using signed representation in the dBm unit.

The value 0x7F (127) MUST indicate that the measurement is not available.

All other values MUST represent the actual Tx Power setting, in dBm.

Remote Node Tx Power (8 bits)

This field is used to indicate the Tx Power (in dBm) used by the remote node for the transmission.

This field MUST be encoded as using signed representation in the dBm unit.

The value 0x7F (127) MUST indicate that the measurement is not available.

All other values MUST represent the actual Tx Power setting, in dBm.

5.2.13.2.13 Measured Noise Floor (2 bytes)

This TLV is used to report the Measured Noise Floor (in dBm) used during the transmissions with the remote node.

Table 5.180: Measured Noise Floor

7	6	5	4	3	2	1	0
IME - Type = 0x0A							
IME - Length = 2							
IME - Value = Local Node Measured Noise Floor							
IME - Value = Remote Node Measured Noise Floor							

Local Node Measured Noise Floor (8 bits)

This field is used to indicate the Measured Noise Floor (in dBm) by the local node during the transmission.

This field MUST be encoded as using signed representation and MUST be encoded according to Table 5.181.

Table 5.181: Noise Floor Encoding

Value (signed)	Description
0x7F (127)	NOT_AVAILABLE. This value is returned for unused hops or if no RSSI measurement is available.
0x7E (126)	MAX_POWER_SATURATED This value is returned if the measured RSSI is above the maximum power.
0x7D (125)	BELOW_SENSITIVITY. This value is returned if the measurement is below the receiver's sensitivity.
0x7D (125).. 0x80 (-128)	This value represents the measurement in dBm

Remote Node Measured Noise Floor (8 bits)

This field is used to indicate the Measured Noise Floor (in dBm) by the remote node during the transmission.

This field MUST be encoded as using signed representation and MUST be encoded according to Table 5.181.

5.2.13.2.14 Outgoing RSSI (7 bytes)

Table 5.182: Outgoing RSSI

7	6	5	4	3	2	1	0
IME - Type = 0x0B							
IME - Length = 5							
IME - Value = Outgoing RSSI hop 1							
IME - Value = Outgoing RSSI hop 2							
IME - Value = Outgoing RSSI hop 3							
IME - Value = Outgoing RSSI hop 4							
IME - Value = Outgoing RSSI hop 5							

The IME - Values advertise the RSSI value measured in the outgoing direction (from our local node to the remote node).

Outgoing RSSI Hop (5 bytes)

The RSSI values MUST be encoded as using signed representation in the dBm unit and according to Table 5.172.

5.2.13.2.15 Encapsulation Format Information Option

The Encapsulation Format Information Option is used to carry information about the Z-Wave encapsulations that were or must be used to communicate between the Z-Wave node and the sending host (e.g. a Z/IP Gateway).

The purpose of this Option is to preserve the encapsulation between a Z-Wave node and a host (e.g. Z/IP Gateway).

CC:0023.00.02.11.016 A Z/IP Gateway MUST use the encapsulation indicated in the Encapsulation Format Information Option when transmitting Z/IP Commands over in a Z-Wave Network.

CC:0023.00.02.11.017 A Z/IP Gateway receiving a Z-Wave Command that must be forwarded over an IP network MUST indicate in the Encapsulation Format Information Option what the Z-Wave encapsulation was.

CC:0023.00.02.11.018 If a Z/IP client receives this Option and the Z-Wave Command requires to return a response, the Z/IP client MUST apply the encapsulation indicated by the Option when sending a reply.

CC:0023.00.02.13.003 A Z/IP client MAY use this Option to dictate the encapsulation format when sending unsolicited messages.

Table 5.183: Encapsulation Format Information Option

7	6	5	4	3	2	1	0
Critical = 1	Option Type = ENCAPSULATION_FORMAT_INFO = 4						
Option Length = 2							
Security 2 Security Class							
Reserved							CRC16

Critical (1 bit)

CC:0023.00.02.11.019 This field indicates that the whole frame MUST be discarded if the extension is not supported.

This field MUST be set to 1 when using the Encapsulation Format Info Option.

Option Type (7 bits)

CC:0023.00.02.11.01A The Type field MUST be set to 4 for the Encapsulation Format Information Option.

Option Length (8 bits)

CC:0023.00.02.11.01B The Option Length field MUST indicate the length of the Option Data fields, which is currently defined as 2 bytes long.

Security 2 Security Class (1 byte)

CC:0023.00.02.11.01C This Security 2 Security Class field indicates which Security 2 Security Class MUST be used for communication with the target node.

CC:0023.00.02.11.01D A receiving node MUST replace previous information about a node's secure capabilities with the information contained in this field and attempt subsequent communication with the target node using the highest security key contained in this command.

CC:0023.00.02.11.01E This field MUST be encoded as a bit field and according to Table 5.184.

Table 5.184: Security Class Field Encoding

Bit set to 1	Security 2 - Security Class
None	NON_SECURE
0	S2_UNAUTHENTICATED
1	S2_AUTHENTICATED
2	S2_ACCESS_CONTROL
7	S0

CRC16 (1 bit)

The CRC16 field indicates whether communication with the target node use CRC16 encapsulation or not.

CC:0023.00.02.11.01F The value 1 MUST indicate that CRC16 encapsulation is used and MUST be used for subsequent communication with the Z-Wave node.

CC:0023.00.02.11.020 The value 0 MUST indicate that CRC16 encapsulation is not used and MUST NOT be used for subsequent communication with the Z-Wave node.

CC:0023.00.02.11.021 The CRC16 field MUST NOT be set to 1 if the Security 2 Security Class field is different than "NON_SECURE"

5.2.13.2.16 Z-Wave Multicast Addressing Option

This option is used to advertise if Multicast Addressing has been used by the sending node in the Z-Wave network.

CC:0023.00.02.12.002 A Z/IP Client SHOULD NOT use this option when sending a Z/IP Packet Command.

CC:0023.00.02.11.022 A Z/IP Gateway supporting Z/IP Command version 4 or newer MUST use this option in a Z/IP Packet Command if forwarding a command that has been received using Multicast addressing from a Z-Wave node.

The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.185: Z-Wave Multicast Addressing Option

7	6	5	4	3	2	1	0
Critical = 0	Option Type = ZWAVE_MULTICAST_ADDRESSING = 5						
Option Length = 0							

Critical (1 bit)

CC:0023.00.02.11.023 The Critical field MUST be set to 0.

Option Type (7 bits)

CC:0023.00.02.11.024 The Option Type field MUST be set to 0x05 to indicate the Z-Wave Multicast Addressing Option.

Option Length (8 bits)

CC:0023.00.02.11.025 The Option Length field MUST indicate the length of the Option Data field.

CC:0023.00.02.11.026 No Option Data is currently defined for this option; this field MUST be set to 0 and the Option Data field MUST be omitted.

5.2.14 Z/IP 6LoWPAN Command Class, version 1

The Z/IP 6LoWPAN Command Class supports the transmission of IPv6 Packets over Z-Wave networks.

The Z/IP 6LoWPAN Command Class, version 1 is defined by RFC 7428.

5.2.15 Z/IP Gateway Command Class, version 1

The Z/IP gateway Command Class is used for configuration and management of a Z/IP gateway, e.g. to enable portal communication.

5.2.15.1 Interoperability considerations

The Z/IP Gateway Command Class is intended for use together with the Z/IP Portal Command Class to provide a streamlined workflow for preparing and performing installation of Z/IP Gateways in consumer premises. Section 5.2.18.1.1 presents the concepts of tunnel creation, maintenance and bootstrapping of a Z/IP Gateway. A Z/IP Gateway may operate in a standalone environment where it is only accessed locally or it may create a tunnel to a portal provider to allow remote access. Commands defined in this Command Class MUST be encapsulated in Z/IP Packets.

5.2.15.2 Gateway Mode Set Command

Any host may send the Gateway Mode Set command during initial configuration of the gateway. Most likely, a service provider or an OEM will use the command in a central facility when preparing deployment at customer premises.

Table 5.186: Gateway Mode Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = GATEWAY_MODE_SET (0x01)							
Mode							

Mode (1 byte)

This field sets the communication mode of the Z/IP Gateway

Table 5.187: Gateway Mode Set::Mode encoding

Value	Mode
0x01	Stand-alone (default)
0x02	Portal

If Mode is set to ”Stand-alone”, the Z/IP Gateway MUST NOT do any attempts to create secure tunnels to other peers in the LAN or in the Internet.

The default mode SHOULD be ”Stand-alone”. By default, peer profiles SHOULD NOT be defined.

A Mode value set to ”Portal” MUST be ignored if the actual gateway does not support the Z/IP Portal Command Class, If Mode is set to ”Portal”, the Z/IP Gateway MUST use the peer profile defined with the Gateway Peer Set command to create a secure connection to the portal server.

Once the Z/IP Gateway has been configured for portal connection creation, the Z/IP Gateway SHOULD be locked for unauthorized access by issuing a Gateway Lock Set; refer to Section 5.2.15.8.

5.2.15.3 Gateway Mode Get Command

The Gateway Mode Get command is used to request the current Z/IP Gateway operational mode.

The Gateway Mode Report Command MUST be returned in response to this command except if the Z/IP Gateway is locked with the Gateway Lock Set command and the Hide parameter of the Gateway Lock Set command was enabled.

In that case, the Gateway Mode Get command MUST be silently ignored.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.188: Gateway Mode Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = GATEWAY_MODE_GET (0x02)							

5.2.15.4 Gateway Mode Report Command

This command is used to advertise the mode.

Table 5.189: Gateway Mode Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = GATEWAY_MODE_REPORT (0x03)							
Mode							

Mode (1 byte)

This field indicates the communication mode of the Z/IP Gateway. Refer to Section 5.2.15.2 and Table 5.187 for details.

5.2.15.5 Gateway Peer Set Command

The Peer Set Command is used to define one or more peers to which the Z/IP Gateway connects. The peer may be a portal server or one or more Z/IP Gateways.

A Peer Set command MUST always carry the peer identity as an IPv6 address and an IP port number. The command SHOULD also specify the symbolic peer name as a FQDN.

If the Gateway Mode is set to ”Portal”, there MUST NOT be defined more than one Peer profile.

If the Gateway Mode is set to ”Stand-alone”, there MUST NOT be defined any peer profiles.

Table 5.190: Gateway Mode Peer Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = GATEWAY_PEER_SET (0x04)							
Peer Profile							
IPv6 Address 1							
...							
IPv6 Address 16							
Port 1							
Port 2							
Reserved		Peer Name Length					
Peer Name 1 (UTF-8) (Optional)							
... (Optional)							
Peer Name N (UTF-8) (Optional)							

Peer Profile (8 bits)

This field identifies the actual peer profile.

The value 0 (zero) is reserved for future use.

The first peer profile MUST be number 1.

IPv6 Address

Full IPv6 address with no compression. The address SHOULD be in the ULA IPv6 prefix or in a globally routable IPv6 prefix. The address MAY be an IPv4-mapped IPv6 address.

The field MUST NOT carry a link-local IPv6 address.

The IPv6 address MAY be specified as ::/128 (all zeros), i.e. the unspecified address. If setting the IPv6 address field to the unspecified IPv6 address, the Peer Name field MUST be set to a DNS-resolvable FQDN.

Port (16 bits)

This field MUST carry the port number that the peer is listening on. The peer SHOULD use port number 44123 [19].

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Peer Name Length (6 bits)

May be any value from 0 to 63. The value indicates the number of Peer Name bytes following this field. The number of readable characters may be less since some UTF-8 characters are represented by two or more bytes.

Peer Name (N bytes) (optional)

This field is only present if the Peer Name Length field has a value greater than zero.

The Peer Name field MUST be formatted as a UTF-8 based FQDN string such as ”example.com”.

Only if that fails, the Z/IP Gateway SHOULD try connecting to the peer using the Peer Name and the Port.

A Z/IP Gateway SHOULD try connecting to the peer using the IPv6 address and the Port.

5.2.15.6 Gateway Peer Get Command

The Gateway Peer Get Command is used to request active peer profiles.

The Gateway Peer Report Command MUST be returned in response to this command except if the Z/IP Gateway is locked with the Gateway Lock Set command and the Hide parameter of the Gateway Lock Set command was enabled.

In that case, the Gateway Peer Get command MUST be silently ignored.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.191: Gateway Mode Peer Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = GATEWAY_PEER_GET (0x05)							
Peer Profile							

Peer Profile (8 bits)

This field identifies the actual peer profile.

A requesting host SHOULD start specifying the Peer Profile value 1 (one). This will cause the Z/IP Gateway to indicate the number of actual peers in the returned Gateway Peer Report command.

5.2.15.7 Gateway Peer Report Command

The Gateway Peer Report Command is used to report details of a peer profile.

A Gateway Peer Report command MUST always carry the peer address as an IPv6 address and MUST include the peer resource name if it was previously specified.

Table 5.192: Gateway Mode Peer Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = GATEWAY_PEER_REPORT (0x06)							
Peer Profile							
Peer Count							
IPv6 Address 1							
...							
IPv6 Address 16							
Port 1							
Port 2							
Reserved		Peer Name Length					
Peer Name 1 (UTF-8) (Optional)							
... (Optional)							
Peer Name N (UTF-8) (Optional)							

Peer Profile

This identifier is used to identify the actual peer profile.

The value 0 (zero) is reserved for future use.

Peer Count (8 bits)

This field indicates the number of peer profiles currently defined.

If the Peer Count field has the value 0, all other fields of the Gateway Peer Report MUST be 0.

IPv6 Address

This field MUST carry a full IPv6 address with no compression.

Port (16 bits)

This field MUST carry the port number that the peer is listening on. The peer SHOULD use port number 44123 [19].

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Peer Name Length (6 bits)

May be any value from 0 to 63. The value indicates the number of Peer Name bytes following this field. The number of readable characters may be less since some UTF-8 characters are represented by two or more bytes.

Peer Name (N bytes) (optional)

This field is only present if the Peer Name Length field has a value greater than zero.

The Peer Name field MUST be formatted as a UTF-8 based FQDN string such as ”example.com”.

If the Peer Count value is zero, the Resource Name string MUST be unspecified (zero-length).

5.2.15.8 Gateway Lock Set Command

The Lock Set command MUST lock down access to configuration parameters in the Z/IP Gateway relating to secure connections and portal login. Once the Z/IP Gateway has been locked, it MUST NOT be possible to unlock the device. Two exceptions apply:

- A factory default reset MUST unlock the Z/IP Gateway and revert settings to default.
- An unlock command received via an authenticated secure connection to the portal MUST unlock the Z/IP Gateway.

Table 5.193: Gateway Lock Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = GATEWAY_LOCK_SET (0x07)							
Reserved						Show	Lock

Lock (1 bit)

This field controls if Z/IP Gateway configuration parameters may be changed by the customer.

The value 0 MUST indicate that the parameters are unlocked and can be changed by the customer.

The value 1 MUST indicate that the parameters are locked and cannot be changed by the customer. The Z/IP gateway MUST accept to receive the Lock=1 flag from any connection.

The Z/IP gateway MUST NOT accept to receive the Lock=0 flag from any connection; except for an authenticated secure connection to the portal.

To prevent users and trojan viruses from creating tunnels to rogue portals, the Z/IP Gateway SHOULD automatically lock access to secure tunnel configuration parameters 24 hours after a factory default reset.

Show (1 byte)

This field controls if Z/IP Gateway configuration parameters may be read by the customer after the Z/IP Gateway has been locked.

The value 0 MUST indicate that parameters are not available to the customer.

The value 1 MUST indicate that parameters are available to the customer.

If the Show parameter is '0' the Z/IP Gateway MUST NOT respond to any queries for Z/IP Gateway parameters.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

5.2.15.9 Unsolicited Destination Set Command

The Unsolicited Destination Set Command is used to configure the destination information that the Z/IP Gateway must use for incoming unsolicited frames.

Table 5.194: Unsolicited Destination Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = UNSOLICITED_DESTINATION_SET (0x08)							
Unsolicited IPv6 Destination 1							
...							
Unsolicited IPv6 Destination 16							
Unsolicited Destination Port 1							
Unsolicited Destination Port 2							

Unsolicited IPv6 Destination (16 bytes)

Unsolicited Z-Wave frames received from any Z-Wave node MUST be forwarded to the Unsolicited IPv6 Destination address.

Unsolicited Destination Port (2 bytes)

Unsolicited Z-Wave frames received from any Z-Wave node MUST be forwarded to the Unsolicited IPv6 Destination Port. Byte 1 is the Most Significant byte.

The Unsolicited IPv6 Destination Port SHOULD be port 4123.

IPv6 enabled Z-Wave nodes MAY send Z-Wave commands encapsulated in Z/IP Packets to the Unsolicited IPv6 Destination address. The Z/IP Gateway MUST translate the destination port of Z/IP Packets destined for the Unsolicited IPv6 Destination address from port 4123 to the port number defined for the Unsolicited Destination Port.

5.2.15.10 Unsolicited Destination Get Command

The Unsolicited Destination Get Command is used to request the destination information that the Z/IP Gateway uses for incoming unsolicited frames.

The Unsolicited Destination Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.195: Unsolicited Destination Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = UNSOLICITED_DESTINATION_GET (0x09)							

5.2.15.11 Unsolicited Destination Report Command

The Unsolicited Destination Report Command is used to report the destination information that the Z/IP Gateway uses for incoming unsolicited frames.

The command format is outlined below:

Table 5.196: Unsolicited Destination Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = UNSOLICITED_DESTINATION_REPORT (0x0A)							
Unsolicited IPv6 Destination 1							
...							
Unsolicited IPv6 Destination 16							
Unsolicited Destination Port 1							
Unsolicited Destination Port 2							

Unsolicited IPv6 Destination (16 bytes) Refer to Section 5.2.15.9. Unsolicited Destination Port (2 bytes) Refer to Section 5.2.15.9.

5.2.15.12 Application Node Info Set Command

The Application Node Info Set Command is used to set the application specific part of the Node Information that a Z/IP Gateway returns when queried by a Z-Wave node.

Table 5.197: Application Node Info Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = COMMAND_APPLICATION_NODE_INFO_SET (0x0B)							
Command Class 1 *)							
...							
Command Class N *)							

*) Command classes may be extended spanning two bytes for one command class

Command Class (N bytes)

See description *Node info cached report command* and Table 5.27.

5.2.15.13 Application Node Info Get Command

The Application Node Info Get Command is used to request the Node Information that a Z/IP Gateway returns when queried by a Z-Wave node.

The Application Node Info Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.

A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Table 5.198: Application Node Info Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = COMMAND_APPLICATION_NODE_INFO_GET (0x0C)							

5.2.15.14 Application Node Info Report Command

The Application Node Info Report Command is used to report the Node Information that a Z/IP Gateway returns when queried by a Z-Wave node. Only the application specific part is returned.

Table 5.199: Application Node Info Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_GATEWAY (0x5F)							
Command = COMMAND_APPLICATION_NODE_INFO_GET (0x0C)							
Command Class 1 *)							
...							
Command Class N *)							

*) Command classes may be extended spanning two bytes for one command class

Command Class (N bytes)

Refer to Section 5.2.15.12.

5.2.16 Z/IP ND Command Class, version 1

Z/IP ND Command Class builds on the same principles as IPv6 ND RFC 4861, RFC 3122 and is inspired by the frame formats. Z/IP ND does however not implement the full range of functions defined for IPv6 ND.

5.2.16.1 Interoperability considerations

Z/IP ND commands allow a Z/IP Gateway to translate between an IPv6 address and a Z-Wave NodeID (Link-Layer address) when requested by an IP host located in a Z-Wave HAN or anywhere else in an IPv6 environment. The Z/IP ND Commands are not intended for classic Z-Wave applications. Z/IP ND messages MUST always be carried in UDP datagrams without Z/IP Packet encapsulation.

5.2.16.2 Security considerations

The commands defined in this Command Class MUST always be accepted by a receiving node, regardless of whether IP security (such as DTLS) was used for the transmission.

5.2.16.3 Z/IP Node Solicitation Command

The Z/IP Node Solicitation Command is used to resolve an IPv6 address of a Z-Wave node to the NodeID (Link-Layer address) of that node in its actual Z-Wave HAN / IP subnet.

Several IPv6 addresses MAY be resolved to the same NodeID.

The Zip Node Solicitation MUST be transmitted in unicast to the Z/IP Gateway of the actual Z/IP HAN. A Z/IP Gateway MUST NOT respond to Zip Node Solicitation commands received via multicast.

A Zip Node Advertisement MUST be returned in response to the Zip Node Solicitation.

Table 5.200: Z/IP Node Solicitation Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_ND (0x58)							
Command = ZIP_NODE_SOLICITATION (0x03)							
Reserved							
NodeID = 0							
IPv6 Address 1							
...							
IPv6 Address 16							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

NodeID (8 bits)

The NodeID field is not used in the Zip Node Solicitation. The field MUST be set to zero by a transmitting host and ignored by a receiving host.

IPv6 Address (16 bytes)

The IP address of the target Z-Wave node. It MUST NOT be a multicast address.

5.2.16.4 Z/IP Inverse Node Solicitation Command

The Z/IP Inverse Node Solicitation Command is used to resolve a NodeID (link-layer address) of a Z-Wave node to an IPv6 address of that node in its actual Z-Wave HAN / IP subnet.

The Zip Inverse Node Solicitation MUST be transmitted in unicast to the Z/IP Gateway of the actual Z/IP HAN. A Z/IP Gateway MUST NOT respond to Zip Inverse Node Solicitation commands received via multicast.

A *Z/IP Node Advertisement Command* MUST be returned in response to the Zip Inverse Node Solicitation.

Table 5.201: Z/IP Inverse Node Solicitation Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_ND (0x58)							
Command = ZIP_INV_NODE_SOLICITATION (0x04)							
Reserved				Local	Reserved		
NodeID							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Local (1 bit)

The flag indicates that the requester would like to receive the site-local address (a.k.a. ULA) even if a global address exists. The flag is typically used by a configuration tool when creating an association between HAN nodes within the same site. Using ULA addresses for intra-HAN association serves to decouple long-term associations in the home from frequently changing global prefixes.

NodeID (8 bits)

The NodeID (Link-Layer Address) that is to be resolved to an IPv6 address.

5.2.16.5 Z/IP Node Advertisement Command

The Z/IP Node Advertisement Command is sent by a Z/IP Gateway in response to a unicast Zip Node Solicitation or a unicast Zip Inverse Node Solicitation. The Zip Node Advertisement SHOULD advertise valid information in both the IPv6 Address and NodeID fields if such information.

A Zip Node Advertisement MUST NOT be transmitted in unsolicited messages.

A Zip Node Advertisement MUST NOT be transmitted in multicast.

Table 5.202: Z/IP Node Advertisement Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_ND (0x58)							
Command = ZIP_NODE_ADVERTISEMENT (0x01)							
Reserved					Local	Validity	
NodeID							
IPv6 Address 1							
...							
IPv6 Address 16							
Home ID 1							
...							
Home ID 4							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Local (1 bit)

The flag indicates that the requester asked for the site-local address (a.k.a. ULA).

A ULA address is returned. A global address may exist.

Validity (2 bits)

A two-bit codeword that indicates the validity of the returned information.

Table 5.203: Zip Node Advertisement::Validity parameter encoding

Value	Validity identifier	Comment
0x00	INFORMATION_OK	The Node Advertisement contains valid information in both the IPv6 Address and NodeID fields.
0x01	INFORMATION_OBSOLETE	The information in the IPv6 Address and NodeID fields is obsolete. No node exists in the network with this address information. The information should only be used to inform a user that the actual node is no more present in the network.
0x02	INFORMATION_NOT_FOUND	The responding Z/IP Gateway could not locate valid information. IPv6 Address and NodeID fields MUST be ignored.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

NodeID (8 bits)

The NodeID MUST correspond to the IPv6 Address contained in this Zip Node Advertisement message.

IPv6 Address (16 bytes)

The IPv6 Address MUST correspond to the NodeID contained in this Zip Node Advertisement message.

An IPv6 host may have more than one IPv6 address.

If the Zip Node Advertisement is a response to a Zip Node Solicitation, the IPv6 Address MUST be the same as the one carried in the Zip Node Solicitation.

A Z/IP Gateway returning a Zip Node Advertisement in response to a Zip Inverse Node Solicitation may have several IPv6 addresses to choose from. The reported IPv6 Address MUST be selected according to the following priority list:

If ”local” flag is set:

1. Unique Local Address (ULA) prefix

If ”local” flag is not set:

1. Global routable address
2. Unique Local Address (ULA) prefix

In other words, if the Z/IP node has a globally routable address then that address MUST be reported.

Else the locally routable address constructed from a ULA prefix and the NodeID MUST be reported.

If a Z/IP Inverse Node Solicitation command is transmitted in an IPv6 packet the returned Z/IP Node Advertisement MUST carry the IPv6 address of the actual node.

If a Z/IP Inverse Node Solicitation command is transmitted in an IPv4 packet the returned Z/IP Node Advertisement MUST carry the IPv4 address of the actual node formatted as an IPv4-mapped IPv6 address RFC 4291.

The IP address carried in the Z/IP Node Advertisement MAY be all zeros. The reason may be that the Z/IP Gateway is still waiting for a DHCP response after including a new node. A Z/IP client MAY re-issue another a Z/IP Inverse Node Solicitation command after a delay of 2 seconds. The delay MUST be doubled before each new attempt. The delay SHOULD be capped at 32 seconds.

Home ID (4 bytes)

Unique network address of the link layer network. All nodes in a Z-Wave network share the same Home ID. The Home ID MAY be used for bookkeeping of complete node information in managed installations.

5.2.17 Z/IP ND Command Class, version 2

5.2.17.1 Compatibility Considerations

The Z/IP ND Command Class, version 2 introduces support for Extended NodeIDs. This version is backwards compatible with version 1.

All fields not described in this version MUST remain unchanged from version 1. The following commands are updated:

- Z/IP Inverse Node Solicitation Command
- Z/IP Node Advertisement Command

5.2.17.2 Interoperability considerations

Refer to Section 5.2.16.1 Interoperability considerations.

5.2.17.3 Security considerations

Refer to Section 5.2.16.2 Security considerations.

5.2.17.4 Z/IP Inverse Node Solicitation Command

The Z/IP Inverse Node Solicitation Command is used to resolve a NodeID (link-layer address) of a Z-Wave node to an IPv6 address of that node in its actual Z-Wave HAN / IP subnet.

The Zip Inverse Node Solicitation MUST be transmitted in unicast to the Z/IP Gateway of the actual Z/IP HAN. A Z/IP Gateway MUST NOT respond to Zip Inverse Node Solicitation commands received via multicast.

A Zip Node Advertisement MUST be returned in response to the Zip Inverse Node Solicitation.

Table 5.204: Z/IP Inverse Node Solicitation Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_ND (0x58)							
Command = ZIP_INV_NODE_SOLICITATION (0x04)							
Reserved				Local	Reserved		
NodeID							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

Fields not described below MUST remain unchanged from version 1.

NodeID (8 bits)

This field is used to indicate the NodeID (Link-Layer Address) that is to be resolved to an IPv6 address.

The value 0xFF MUST indicate that the NodeID to be resolved MUST be read from the Extended *NodeID* field.

Extended NodeID (2 bytes)

This field is used to specify the NodeID (Link-Layer Address) that is to be resolved to an IPv6 address.

If the *NodeID* field is in the range 0x00..0xFE, this field MUST be set to the same value as the NodeID field by a sending node.

If the *NodeID* field is set to 0xFF, this field MUST indicate the NodeID that is to be resolved to an IPv6 address.

5.2.17.5 Z/IP Node Advertisement Command

The Z/IP Node Advertisement Command is sent by a Z/IP Gateway in response to a unicast Zip Node Solicitation or a unicast Zip Inverse Node Solicitation. The Zip Node Advertisement SHOULD advertise valid information in both the IPv6 Address and NodeID fields if such information.

A Zip Node Advertisement MUST NOT be transmitted in unsolicited messages.

A Zip Node Advertisement MUST NOT be transmitted in multicast.

Table 5.205: Z/IP Node Advertisement Command, version 2

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_ND (0x58)							
Command = ZIP_NODE_ADVERTISEMENT (0x01)							
Reserved					Local	Validity	
NodeID							
IPv6 Address 1							
...							
IPv6 Address 16							
Home ID 1							
...							
Home ID 4							
Extended NodeID (MSB)							
Extended NodeID (LSB)							

All fields not described below MUST remain unchanged from version 1.

NodeID (8 bits)

The NodeID MUST correspond to the IPv6 Address contained in this Zip Node Advertisement message.

The value 0xFF MUST indicate that the NodeID contained in this ZIP Node Advertisement message MUST be read from the *Extended NodeID* field.

Extended NodeID (2 bytes)

This field is used to specify the NodeID that is resolved to the specified IPv6 address.

If the *NodeID* field is in the range 0x00..0xFE, this field MUST be set to the same value as the *NodeID* field by a sending node.

If the *NodeID* field is set to 0xFF, this field MUST indicate the NodeID that is to resolved to the specified IPv6 address.

5.2.18 Z/IP Portal Command Class, version 1

The Z/IP Portal Command Class is used for configuration and management communication between a Z/IP portal server and a Z/IP gateway through a secure connection.

The Z/IP Portal command class is intended for use together with the Z/IP Gateway command class to provide a streamlined workflow for preparing and performing installation of Z/IP Gateways in consumer premises.

5.2.18.1 Interoperability considerations

This Command Class MUST NOT be used outside trusted environments, unless via a secure connection. This Command Class SHOULD be further limited for use only via a secure connection to an authenticated portal server.

Commands defined in this Command Class MUST be encapsulated in Z/IP Packets.

5.2.18.1.1 On the use of Z/IP Gateway and Z/IP Portal command classes

This section presents the concepts of tunnel creation, maintenance and bootstrapping of a Z/IP Gateway.

A secure connection is established by the Z/IP gateway connecting to a peer. The Z/IP Gateway::Gateway Peer Set command is used to define a peer.

A secure connection to a portal is a special case of the general secure connection. When connecting to a portal, the Z/IP Gateway is operated in portal mode; having most network configuration parameters pushed from the portal. In Portal mode, the Z/IP Gateway only accepts the creation of one peer.

The gateway Mode Set command controls whether the Z/IP gateway operates as a normal IP router; learning IP network information from the network or if the configuration is pushed from a portal.

A Z/IP Gateway has two modes of operation, each mode determines how the Z/IP Gateway can be configured and how it should react to a number of command classes. The mode of operation is determined by the customer depending on the type of product they wish to develop.

1. Service Provider (SP) (Only Portal Mode available)
    a. Through Secure Tunnel connection (Locked & Unlocked): MUST accept Portal & Gateway Command Classes, Firmware Command Class
    b. Factory default: Device remains locked, and attempts communication to portal, reverts to default firmware configuration.
    c. Any other attempt to use above command classes MUST be ignored
2. Consumer Electronics (CE) (Portal and Stand-Alone Mode available)
    a. Portal Mode:
        i. Through Secure Tunnel connection (Locked & Unlocked): MUST accept Portal & Gateway Command Classes, Firmware Command Class
        ii. Local Access (Unlocked only): MUST accept Portal & Gateway Command Classes, Firmware Command Class
        iii. Any other attempt to use above command classes MUST be ignored
        iv. Factory default: Device is unlocked, and may connect to portal if there is a default configuration containing portal configuration
    b. Stand-Alone Mode
        i. Local access (Unlocked only): MUST accept Portal & Gateway Command Classes, Firmware Command Class
        ii. Any other attempt to use above command classes MUST be ignored

iii. Factory default: Device is unlocked, and may connect to portal if there is a default configuration containing portal configuration

3. Gateway Lock MUST prevent any configuration parameter in Portal and Gateway from being modified locally. Configuration through portal is always allowed.
4. Only the secure tunnel is considered a trusted environment when locked. When unlocked the LAN is also considered "trusted".
5. In all cases, a Factory Default does not perform Z-Wave Default set, meaning the Z-Wave network is left intact. If required, Network Management Default Set MAY be called manually following a Factory Default.

5.2.18.2 Gateway Configuration Set Command

The command is used by a portal server to push settings to a Z/IP Gateway via a secure connection.

The Z/IP gateway MUST return a Gateway Configuration Status message in response to a Gateway Configuration Set message.

Table 5.206: Gateway Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_PORTAL (0x61)							
Command = GATEWAY_CONFIGURATION_SET (0x01)							
LAN IPv6 Address 1							
...							
LAN IPv6 Address 16							
LAN IPv6 Prefix Length							
Portal IPv6 Prefix 1							
...							
Portal IPv6 Prefix 16							
Portal IPv6 Prefix Length							
Default Gateway IPv6 Address 1							
...							
Default Gateway IPv6 Address 16							
PAN IPv6 Prefix 1							
...							
PAN IPv6 Prefix 16							

LAN IPv6 Address (16 bytes)

The LAN IPv6 address MUST be assigned to the LAN interface of the Z/IP Gateway in the consumer premises network. The LAN IPv6 address MUST be used in combination with the LAN IPv6 prefix length.

If the LAN IPv6 address is all zeros, the gateway MUST auto-configure a /64 IPv6 ULA prefix for use by IPv6 enabled hosts in the consumer premises network.

The LAN IPv6 prefix MUST be advertised in IPv6 RAs on the LAN.

LAN IPv6 Prefix Length (1 byte)

The LAN IPv6 prefix length MUST be used by the LAN interface of the Z/IP Gateway in the consumer premises network.

Portal IPv6 Prefix (16 bytes)

The Z/IP Gateway MUST route all IP traffic for the Portal IPv6 Prefix into the secure connection connecting the Z/IP Gateway to the Portal network.

The Portal IPv6 Prefix MUST be used in combination with the Portal IPv6 prefix length.

Portal IPv6 Prefix Length (1 byte)

The Portal IPv6 prefix length MUST be used to scope the routing entry created for the Portal IPv6 Prefix by the Z/IP Gateway.

Default Gateway IPv6 Address (16 bytes)

The Z/IP Gateway MUST send IP packets to the default gateway if the Z/IP Gateway has no routing information for the actual prefix; i.e the prefix is neither the LAN nor the PAN.

The Z/IP Gateway MAY be an address in the Portal IPv6 Prefix.

PAN IPv6 Prefix (16 bytes)

The PAN IPv6 address MUST be assigned to the PAN interface of the Z/IP Gateway. The PAN IPv6 address MUST be scoped by a /64 IPv6 prefix.

If the PAN IPv6 address is all zeros, the gateway MUST auto-configure a /64 IPv6 ULA prefix for use by Z-Wave nodes.

5.2.18.3 Gateway Configuration Status

The message is submitted by a Z/IP Gateway to confirm the reception and processing of a Gateway Configuration Get to a portal.

The Z/IP gateway MUST return a Gateway Configuration Status message in response to a Gateway Configuration Set message.

Table 5.207: Gateway Configuration Status Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_PORTAL (0x61)							
Command = GATEWAY_CONFIGURATION_SET (0x01)							
Status							

Status (1 byte)

Table 5.208: Gateway Configuration Status::Status Encoding

Value	Status indication
0x01	Invalid Configuration Block
0xFF	OK

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

5.2.18.4 Gateway Configuration Get Command

The message is used by a portal to read back configuration settings from a Z/IP Gateway via a secure connection.

Table 5.209: Gateway Configuration Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_PORTAL (0x61)							
Command = GATEWAY_CONFIGURATION_GET (0x03)							

5.2.18.5 Gateway Configuration Report Command:

The message is used by a Z/IP Gateway to return actual settings to a portal via a secure connection.

Table 5.210: Gateway Configuration Report Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_PORTAL (0x61)							
Command = GATEWAY_CONFIGURATION_REPORT (0x04)							
LAN IPv6 Address 1							
...							
LAN IPv6 Address 16							
LAN IPv6 Prefix Length							
Portal IPv6 Prefix 1							
...							
Portal IPv6 Prefix 16							
Portal IPv6 Prefix Length							
Default Gateway IPv6 Address 1							
...							
Default Gateway IPv6 Address 16							
PAN IPv6 Prefix 1							
...							
PAN IPv6 Prefix 16							

LAN IPv6 Address (16 bytes)

Actual IPv6 address assigned to the LAN interface of the Z/IP Gateway in consumer premises.

An all zeros address may have been configured by the portal using a Gateway Configuration Set command. The portal MUST accept receiving an auto-configured /64 IPv6 ULA address even if an all-zeros address was specified previously.

LAN IPv6 Prefix Length (1 byte)

Actual LAN IPv6 prefix length used by the LAN interface of the Z/IP Gateway in consumer premises.

Portal IPv6 Prefix (16 bytes)

Actual IPv6 Prefix used by the Z/IP Gateway to reach the portal end of the secure tunnel.

Portal IPv6 Prefix Length (1 byte)

Actual IPv6 Prefix Length used by the Z/IP Gateway to reach the portal end of the secure tunnel.

Default Gateway IPv6 Address (16 bytes)

Actual IPv6 default gateway address used by the Z/IP Gateway to reach off-link subnet prefixes.

PAN IPv6 Prefix (16 bytes)

Actual IPv6 Prefix used by the Z/IP Gateway to construct IPv6 addresses for Z-Wave nodes.

It may be the ULA prefix if ::/128 was specified in the set.

5.2.18.6 Gateway Unregister Command

The message is used by a portal to force the client to close the existing tunnel.

Table 5.211: Gateway Unregister Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ZIP_PORTAL (0x61)							
Command = GATEWAY_UNREGISTER (0x05)							

6 Command Class Control

6.1 Command Class Control Overview

6.1.1 Controlled and Supported Command Classes

A node can *support* and/or *control* a given Command Class.

If a Command Class is *supported*:

CL:0000.00.11.0D.1 The node implements all the Command Class functionalities and can be set and read back by other nodes. When a Command Class is supported, the node is REQUIRED to implement the whole Command Class and observe the requirements specified in Section 2, Section 3, Section 4, and Section 5.

If a Command Class is *controlled*:

CL:0000.00.13.01.1 The node implements the ability to interview, read and/or set other nodes supporting the Command Class. Nodes controlling Command Classes MAY use a subset of the Commands within a Command Class (for example only Set commands).

6.1.1.1 Control via association groups

A Basic Set Command sent to Association Group destinations is a form of (static or partial) Command Class control via association groups.

CL:0000.00.11.02.1 Even if using a Command Class for controlling other nodes via association groups, the usage MUST comply with the Command Class description (Section 2, Section 3, Section 4, and Section 5) and the controlling node use properly formed commands.

6.1.1.2 "Full" control

CL:0000.00.11.03.1 When a Command Class is marked as *fully* controlled during certification, the controlling node is said to implement "full" control of the actual Command Class and it MUST observe the requirements for the Command Class defined in this document.

CL:0000.00.11.04.1 *Device Type v2 Specification* mandates some nodes to control a given set of Command Classes. It means that the controlling node MUST respect the requirements described in this document for the actual Command Classes.

6.1.1.3 Partial Control

CL:0000.00.13.03.1 When a Command Class is marked as partially controlled during certification, the controlling node is said to implement *partial* control of the actual Command Class and it MAY observe only a subset of the requirements for the Command Class defined in this document. The specific features which are CL:0000.00.11.10.1 partially controlled MUST be noted during certification.

CL:0000.00.13.04.1 If a Command Class is partially controlled, the controlling node MAY skip steps or perform them in a different order for the mandatory node interview detailed in this document. The controlling node MAY retrieve capabilities from a database based on a node's manufacturer specific information.

CL:0000.00.11.11.1 Even if the mandatory node interview is not followed, and not all capabilities are queried, the controlling node MUST NOT send commands or values that the supporting node does not support.

CL:0000.00.13.05.1 If a Command Class is partially controlled, the controlling node MAY implement a subset of the min-
CL:0000.00.12.01.1 imum end user functionalities and user interface. A controlling node SHOULD provide functionalities for the lowest non-deprecated, non-obsoleted Command Class version.

CL:0000.00.11.12.1 All Z-Wave commands sent for a partially controlled Command Class MUST be valid Z-Wave commands

CL:0000.00.11.13.1

All Z-Wave commands sent for a partially controlled Command Class MUST be sent at the highest Security Class granted during security bootstrapping.

CL:0000.00.11.14.1 Partial Control MAY be used for Application Command Classes only (Refer to Section 6.2). Management and Transport-Encapsulation Command Class MUST be fully controlled. (Refer to Section 6.3 and Section 6.4).

6.1.2 Command Classes Support Discovery Requirements

CL:0000.00.11.05.1 A controlling node MUST read the capabilities and secure capabilities of a node/End Point once before trying to control a given Command Class.

CL:0000.00.11.06.2 A controlling node MUST read the Command Class version number of a supporting node using the Version Command Class prior to controlling the Command Class. There can be 2 exceptions to this rule:

- If the controlling node controls only version 1 of a given Command Class, it may skip requesting the supporting node version for the actual Command Class.
- A controlling node may issue version 1 commands in the supporting node interview before knowing the supporting node Command Class' version. (e.g., when interviewing the Version Command Class itself)

CL:0000.00.11.07.1 A node controlling a given command class MUST allow an end user to control the command class in all listening supporting nodes present in the network operating with a security class that is granted to the controlling node.

CL:0000.00.11.08.1 A node controlling a Command Class MUST allow an end user to control a sleeping supporting node if it is the Wake Up destination.

6.1.3 Command Classes Control Requirements

This specification defines how to provide full control of Command Classes by specifying:

- A required Command Class interview or capability discovery
- A minimum required set of actions that an end user can perform or trigger and their associated Z-Wave commands
- A minimum required set of information relating to the supporting node that the end user can see or access.
- An optional set of additional requirements to controlling nodes

6.1.3.1 Mandatory supporting node interview

CL:0000.00.21.01.3 The mandatory node interview specifies the frame flow that MUST be observed on the Z-Wave radio for full control. Set commands MAY be skipped they would have no effect on the supporting node.

CL:0000.00.21.02.1 This interview MUST be performed prior to issuing any control command to the supporting node.

Preferably, it should be done during the commissioning phase of the supporting node or shortly after the inclusion of the controlling node.

CL:0000.00.23.01.1 A fully controlling node MAY issue additional commands that are not specified in the frame flows and MAY issue some commands in a different order. The mandatory frame flow indicates the minimum

CL:0000.00.21.03.1 set of commands that MUST be transmitted to a destination.

CL:0000.00.21.04.1 A fully controlling node MUST NOT skip or abort the interview of a Command Class if another independent Command Class could not be correctly interviewed.

CL:0000.00.23.02.1

A "partially" controlling node MAY skip steps in the specified frame flow if the queried data would not be used by the controller, or the data is retrieved from a separate source (i.e. database of Manufacturer Specific information and capabilities)

CL:0000.00.22.01.1 The following order for reading Command Class capabilities is RECOMMENDED after discovering the Security Class and encapsulation capabilities of a node:

- Version Command Class
- Z-Wave Plus Info Command Class
- Wake Up Command Class
- Association OR Multi Channel Association Command Class
- Association Group Information (AGI) Command Class
- Actuator Command Classes
- Data reporting Command Classes
- Multi Channel Command Class (and repeat the above order for each End Point, if any)

CL:0000.00.22.02.1 Some interviews or controlling scenarios may trigger no response in return to get type commands. In this case, it is RECOMMENDED to use timeouts according to *Role Type Specification*.

6.1.3.2 Minimum end user functionalities

The minimum end user functionalities section specifies what minimum set of actions an end user can perform on a supporting node when using the controlling node.

CL:0000.00.31.01.1 CL:0000.00.33.01.1 Each section describing a required functionality indicates the command(s) that MUST be transmitted on the Z-Wave radio for full control. Fields not present, not described or marked as "free" in the mandatory command(s) MAY be used freely by the controlling node either automatically or based on user input.

CL:0000.00.31.02.1 Fields indicated as "user defined" MUST be filled based on end user input. In this case, the end user MUST be able to select any value. (e.g. if the field is 2 bytes long, values from 0 to 65535 MUST be available)

CL:0000.00.31.03.1 Fields indicated as "User defined among supported" MUST be filled based on user input and the supported values by the supporting node. The controlling node MUST allow the end user to set values supported by a supporting node even if it does not know what a given value represents.

CL:0000.00.31.04.1 Fields indicated as "User defined among x..y and z" MUST be filled based on user input and MUST be able to select between the indicated values.

CL:0000.00.31.06.1 Fields indicated as "User defined or x" MUST be filled either based on user input or automatically by the controller using the indicated value.

CL:0000.00.31.05.1 The product documentation MUST point out how to perform each action associated to the minimum end user functionalities required by a Command Class.

CL:0000.00.31.07.1 If a Command Class is partially controlled and some minimum end user functionality are not met, the product documentation MUST describe how the partially controlled command class differs from the behavior of a fully controlled command class.

6.1.3.3 Command Class control version

CL:0000.00.11.09.1 Like a supporting node implements a version of its supported Command Classes, a controlling node MUST also implement a given version of a Command Class it controls. A node controlling version 3 of a Command Class will be able to control the functionalities of version 1, 2 and 3 supporting nodes.

CL:0000.00.11.0A.2 When issuing commands, a controlling node MUST NOT use the fields described in the mandatory commands sections if they belong to a newer version than the version it claims to control.

CL:0000.00.12.02.1 If a controlling node A controls a node B supporting a lower version, node A SHOULD still use the format (command payload) corresponding to the version node A controls, regardless of the version supported by node B.

CL:0000.00.11.0F.1 A controlling node A sending commands to a node B supporting a lower version MUST NOT use commands introduced in versions that are newer than Node B's version.

A node controlling a Command Class is not mandated to interview, provide minimum end user functionality or show node properties belonging to a newer version of the actual Command Class.

CL:0000.00.11.0B.1 When a version is indicated in the mandatory interview or mandatory commands for end user functionalities, it MUST represent the minimum common version number between the supporting and controlling node. For example, with a version 4 controlling node and version 2 supporting node, the interview, minimum end user functionalities and properties MUST follow version 1 and 2 indications.

6.1.4 Command Classes not Present in this Document

CL:0000.00.13.02.1 CL:0000.00.11.0C.1 A node MAY advertise that it controls a Command Class not (yet) defined in this document during certification. In this case, the controlling node MUST comply with the following guidelines:

- It MUST use all Get type commands to read the nodes capabilities during the node interview
- It SHOULD provide the end user actions using all Set type Commands available in the Command Class
- It SHOULD show or let the end user access all relevant state/property/configuration relating to the Command Class using the Report type Commands available in the Command Class.

A controller following this guideline is likely to be compliant when a new control specification is released for an actual Command Class.

6.2 Application Command Class Control Definitions

6.2.1 Anti-theft Unlock Command Class, version 1

6.2.1.1 Mandatory node interview

CL:007E.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.1.

Figure 6.1: Anti-Theft Unlock Command Class interview

6.2.1.2 Minimum end user functionalities

6.2.1.2.1 Unlock the node

CL:007E.01.31.01.1 CL:007E.01.31.01.1 If the supporting node is in the locked state and runs in restricted mode, the end user MUST be able send an unlock command. When the end user performs this action, the issued command MUST comply with Table 6.1.

Table 6.1: Anti-Theft Unlock::Unlock the node

Field	Value
Command	COMMAND_ANTITHEFT_UNLOCK_SET
Magic Code length	Controlling node defined in 0x01..0xA, based on user input.
Magic Code	User defined CL:007E.01.32.01.1 The Magic Code SHOULD be shown as a hexadecimal value to the end users.

6.2.1.3 Node properties

CL:007E.01.41.01.1 If the supporting node is in the locked state and runs in restricted mode, the controlling node MUST have a UI allowing the end user to see that the supporting node is restricted and that the supporting node requires unlocking to use the full functionality.

CL:007E.01.43.01.1 A controlling node MAY display the information provided in [25] for the reported Z-Wave Alliance locking entity ID.

6.2.2 Barrier Operator Command Class, version 1

6.2.2.1 Mandatory node interview

CL:0066.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.2.

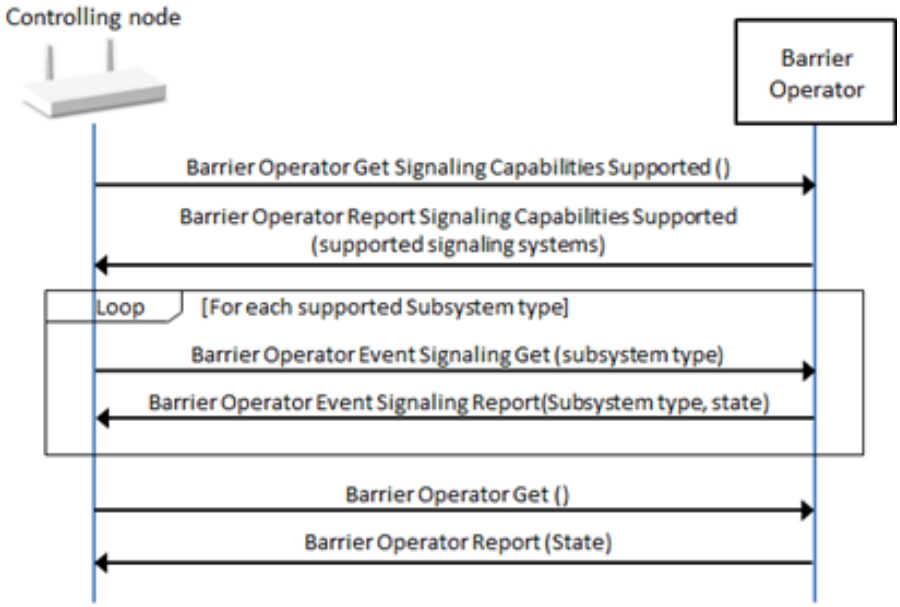


Figure 6.2: Barrier Operator Command Class interview

6.2.2.2 Minimum end user functionalities

CL:0066.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.2.2.1 Initiate opening (or stop closing)

CL:0066.01.31.03.1 When the end user performs this action, the issued command MUST comply with Table 6.2.

Table 6.2: Barrier Operator::Initiate opening (or stop closing)

Field	Value
Command	BARRIER_OPERATOR_SET
Target Value	0xFF

6.2.2.2.2 Initiate closing

CL:0066.01.31.04.1 When the end user performs this action, the issued command MUST comply with Table 6.3.

Table 6.3: Barrier Operator::Initiate closing

Field	Value
Command	BARRIER_OPERATOR_SET
Target Value	0x00

6.2.2.3 Node properties

CL:0066.01.42.01.1 The controlling node SHOULD have a UI allowing the end user to see/access the following properties:

- Last known Barrier State (Open, Closed, stopped at a % position or unknown)
- Last known Subsystems' state (ON/OFF), if any

6.2.2.4 Additional control requirements

CL:0066.01.52.01.2 A node controlling this command class SHOULD also control the Notification Command Class. A node controlling this command class SHOULD activate all supported subsystems by default.

6.2.3 Basic Command Class, version 1-2

6.2.3.1 Mandatory node interview

CL:0020.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.3.

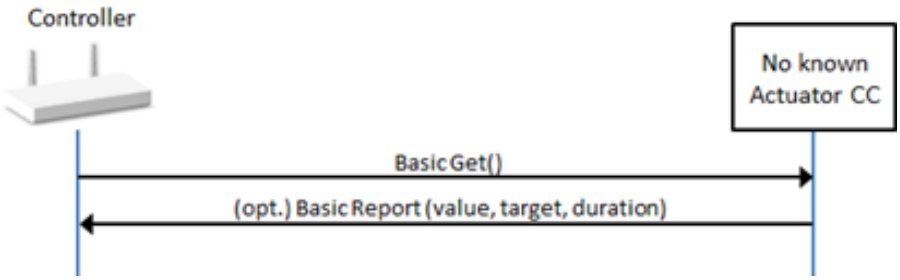


Figure 6.3: Basic Command Class interview

CL:0020.01.21.02.2 A controlling node MUST conclude that the Basic Command Class is not supported by a node (or endpoint) if no Basic Report is returned.

6.2.3.2 Minimum end user functionalities

CL:0020.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.3.2.1 Set the node On/Off state

CL:0020.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.4

Table 6.4: Basic::Set the node state

Field	Value
Command	BASIC_SET
Value	User defined among 0x00 and 0xFF. More values MAY be available to the end user.

6.2.3.3 Node properties

CL:0020.01.43.01.1 A controlling node MAY have a UI allowing the end user to see the following properties:

- Last known state (On or Off)

CL:0020.01.42.01.1 A controlling node SHOULD NOT assume that the last state is as defined in the last Set Command and SHOULD issue a subsequent Basic Get Command even if receiving a Supervision SUCCESS status after issuing a Basic Set Command.

6.2.3.4 Additional control requirements

CL:0020.01.51.01.3 A controlling node MUST NOT use the Basic Command Class for controlling nor showing status (receiving report) of a node (or endpoint) if the controlling node controls at least one actuator command class supported by a node (or endpoint). The actuator command classes are defined in *Application Command Classes*.

6.2.4 Binary Switch Command Class, version 1-2

6.2.4.1 Mandatory node interview

CL:0025.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.4.

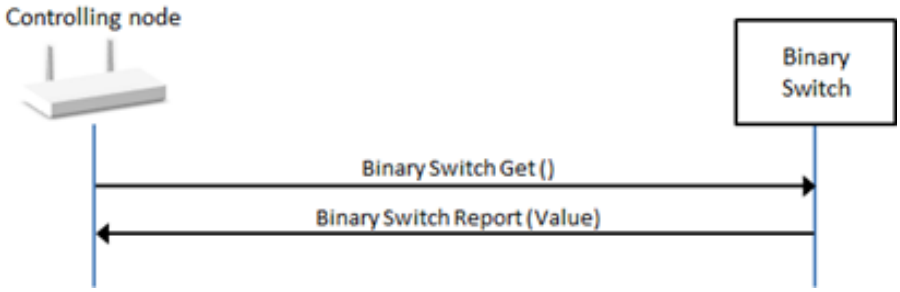


Figure 6.4: Binary Switch Command Class Interview

6.2.4.2 Minimum end user functionalities

CL:0025.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.4.2.1 Set the node On/Off state

CL:0025.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.5.

Table 6.5: Binary Switch::Set the node state

Field	Value
Command	SWITCH_BINARY_SET
Value	User defined among 0x00 and 0xFF.
Duration (v2)	User defined or 0xFF

6.2.4.3 Node properties

CL:0025.01.51.01.1 A node controlling this Command Class SHOULD have a UI allowing the end user to see the following properties:

- Last known state (On or Off)

6.2.5 Central Scene Command Class, version 1-3

6.2.5.1 Mandatory node interview

CL:005B.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.5.

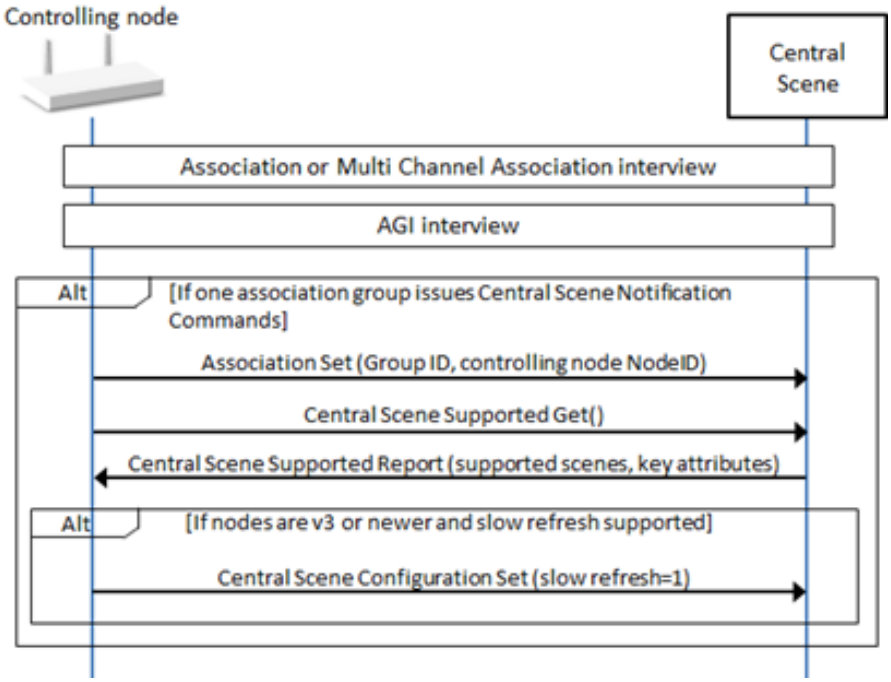


Figure 6.5: Central Scene Command Class interview

CL:005B.01.23.01.1 A node controlling this command class MAY skip the AGI Interview (refer to Section 6.3.2) if the supporting node is a Z-Wave Plus node and the controlling node has estalished a Lifeline Association.

6.2.5.2 Minimum end user functionalities

There is no minimum end user functionalities associated with the control of this Command Class.

CL:005B.01.32.01.1 A controlling node SHOULD allow the user to define which other nodes to actuate when receiving a Central Scene Notification from a given node with a given key attribute and SceneID.

6.2.5.3 Node properties

CL:005B.01.41.01.1 The controlling node MUST have a UI allowing the end user to see how many Scenes ID (or buttons) and key attributes are supported by a node.

CL:005B.01.41.02.1 The controlling node MUST make received Central Scene Notifications available to the end user.

6.2.5.4 Additional control requirements

CL:005B.01.51.01.1 A node controlling this command class MUST also control:

- Association Command Class, version 2
- Association Group Information, version 3

CL:005B.01.51.02.1 A controlling node MUST associate itself to a group issuing Central Scene Notification Commands in order to provide end user functionalities.

CL:005B.01.51.03.1 A controlling node MUST NOT remove associations in order to associate itself to an association group issuing Central Scene Notification Commands.

CL:005B.01.53.01.1 It is OPTIONAL for a controlling node to provide end user functionalities and node properties if it cannot associate itself to an association group sending Central Scene Notification Commands. (e.g. all Association Groups sending the relevant command are full)

6.2.6 Color Switch Command Class, version 1-3

6.2.6.1 Mandatory node interview

CL:0033.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.6.

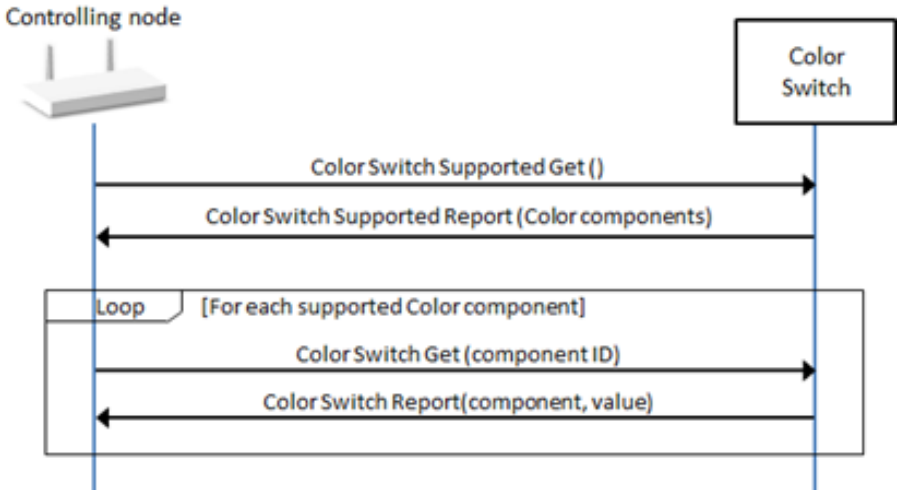


Figure 6.6: Color Switch Command Class interview

6.2.6.2 Minimum end user functionalities

CL:0033.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.6.2.1 Set the color

CL:0033.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.6.

Table 6.6: Color Switch::Set the color

Field	Value
Command	SWITCH_COLOR_SET
Color component count	Determined by the controlling node based on user input
Color Component ID x	User defined among supported
Value x	User defined
Duration (v2)	User defined or 0xFF

CL:0033.01.31.03.1 For this functionality, the color selected by the end user MUST be set using a single command.

6.2.6.2.2 Fade/enhance a color component

CL:0033.01.31.04.1 When the end user performs this action, the issued command MUST comply with Table 6.7.

Table 6.7: Color Switch::Fade/enhance a color component

Field	Value
Command	SWITCH_COLOR_START_LEVEL_CHANGE
Up/down	User defined
Color component ID	User defined among supported
Duration (v3)	User defined or 0xFF

6.2.6.2.3 Stop fading/enhancing a color component

CL:0033.01.31.05.1 When the end user performs this action, the issued command MUST comply with Table 6.8.

Table 6.8: Color Switch::Stop fading/enhancing a color component

Field	Value
Command	SWITCH_COLOR_STOP_LEVEL_CHANGE
Color component ID	User defined among supported

6.2.6.3 Node properties

CL:0033.01.42.01.1 A controlling node SHOULD have a UI allowing the end user to see the following properties:

- Last known configured color or color components values

6.2.6.4 Additional control requirements

CL:0033.01.51.01.1 A node controlling this Command Class MUST also control:

- Binary Switch Command Class, version 2
- Multilevel Switch, version 4

6.2.7 Configuration Command Class, version 1-4

6.2.7.1 Mandatory node interview

CL:0070.03.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.7.

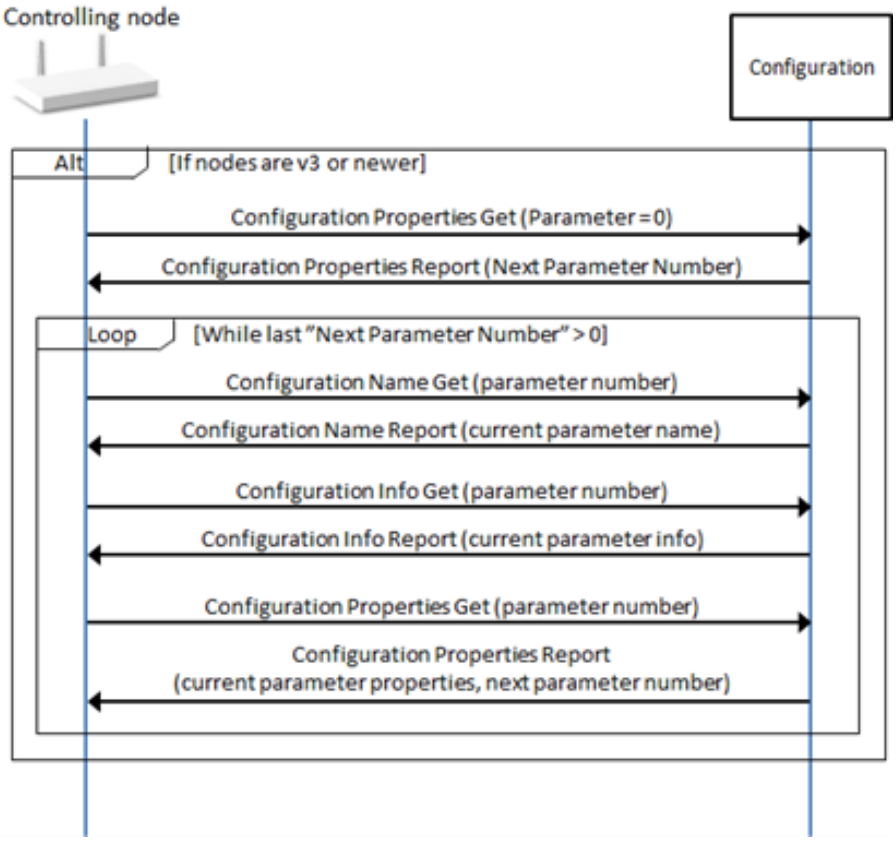


Figure 6.7: Configuration Command Class interview

6.2.7.2 Minimum end user functionalities

CL:0070.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.7.2.1 Set a configuration parameter Value

CL:0070.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.9 for parameters numbers smaller than 256 and Table 6.10 for parameter numbers greater or equal to 256.

Table 6.9: Configuration::Set a (normal) configuration parameter value

Field	Value
Command	CONFIGURATION_SET
Parameter number	For version 2 or older: User defined For version 3 or newer: User defined among supported parameter numbers.
Size	For version 2 or older: User defined among 1, 2 and 4. For version 3 or newer: Automatically determined from configuration parameter number properties
Default	User defined or 0x00 For version 3 or older: this field SHOULD NOT be set to 1.
Command	For version 2 or older: User defined. For version 3 or newer: User defined among supported values

Table 6.10: Configuration::Set an extended range configuration parameter value

Field	Value
Command (v2)	CONFIGURATION_BULK_SET
Parameter offset (v2)	For version 2: User defined among any value (256..65535). For version 3 or newer: User defined among supported parameter numbers.
Number of Parameters (v2)	User defined or 0x01.
Size (v2)	For version 2: User defined among 1, 2 and 4. For version 3 or newer: Automatically determined from configuration parameter number properties
Default (v2)	User defined or 0x00 For version 3 or older: this field SHOULD NOT be set to 1.
Handshake (v2)	0x00
Configuration Value (v2)	For version 2: User defined. For version 3 or newer: User defined among supported values

6.2.7.2.2 Reset all configuration parameter values to default

CL:0070.04.31.01.1 When the end user performs this action, the issued command MUST comply with Table 6.11.

Table 6.11: Configuration::Reset all configuration parameter values to default

Field	Value
Command (v4)	CONFIGURATION_DEFAULT_RESET

6.2.7.3 Node properties

CL:0070.03.41.01.1 If nodes are version 3, the controlling node MUST have a UI allowing the end user to see supported parameter numbers, their current value, their allowed value range and their default value.

Values MUST be presented according to the Format advertised in the Configuration Properties Report Command.

CL:0070.01.41.01.1 If nodes are version 1 or 2, the controlling node MUST have a UI showing the known parameters that have been set by the end user and their current value.

6.2.7.4 Additional control requirements

CL:0070.04.51.01.1 If nodes are version 4, a node controlling this command class MUST NOT issue a Bulk Set Command supporting nodes advertising “No Bulk support” in the Configuration Properties Report Commands.

CL:0070.04.51.02.1 If nodes are version 4, a node controlling this command class MUST NOT allow an end user to issue a Configuration Set or a Bulk Set Command for parameters advertised as “read-only” in the Configuration Properties Report Command.

6.2.8 Door Lock Command Class, version 1-4

6.2.8.1 Mandatory node interview

CL:0062.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.8.

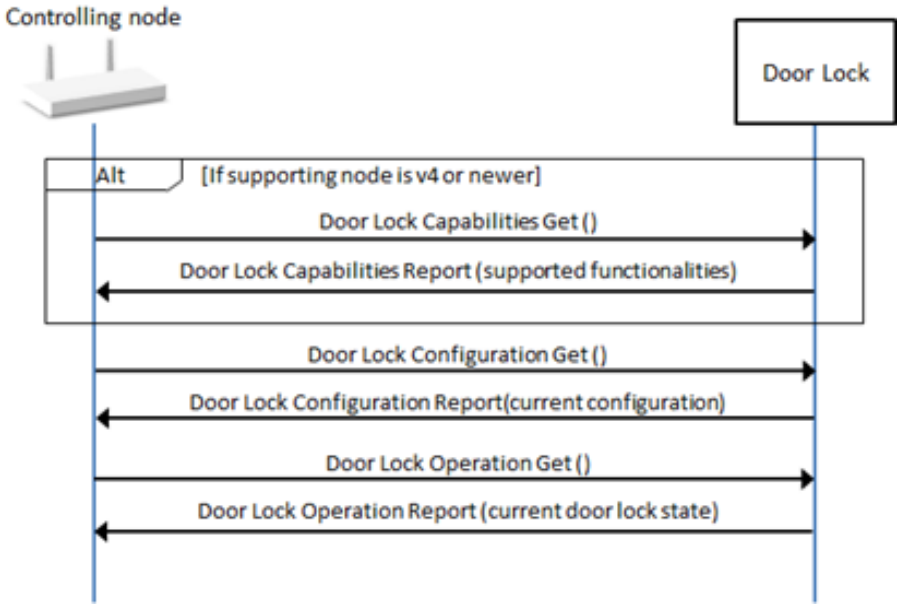


Figure 6.8: Door Lock Command Class interview

6.2.8.2 Minimum end user functionalities

CL:0062.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.8.2.1 Configure the door lock

CL:0062.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.12.

Table 6.12: Door Lock::Configure the door lock

Field	Value
Command	DOOR_LOCK_CONFIGURATION_SET
Operation type	For version 4 or newer: User defined among supported operation types. For version 3 or older: User defined among 0x01..0x02
Outside Door Handles Mode	Free (0xF recommended)
Inside Door Handles Mode	Free (0xF recommended)
Lock Timeout Minutes	User defined (0x00..0xFD) if Operation Type is set to 0x02, else 0xFE
Lock Timeout seconds	User defined (0x00..0x3B) if Operation Type is set to 0x02, else 0xFE
Auto-relock time (v4)	User defined if supported, else 0
Hold and release time (v4)	User defined if supported, else 0
BTB (v4)	User defined if supported, else 0
TA (v4)	User defined if supported, else 0

6.2.8.2.2 Set the door mode

CL:0062.01.31.03.1 When the end user performs this action, the issued command MUST comply with Table 6.13.

Table 6.13: Door Lock::Set the door mode

Field	Value
Command	DOOR_LOCK_CONFIGURATION_SET
Door Lock Mode	For version 4 or newer: User defined among supported modes. For version 3 or older: User defined among 0x00 and 0xFF. Timed Operation modes MUST NOT be selectable by the end user if the door lock is not configured in Timed Operation

6.2.8.3 Node properties

CL:0062.01.41.01.1 Controller SHOULD have a UI allowing the end user to see the following properties:

- Last known Door Lock mode (Secure, Unsecured, etc.)
- Current door lock configuration

6.2.9 Entry Control Command Class, version 1

6.2.9.1 Mandatory node interview

CL:006F.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.9.

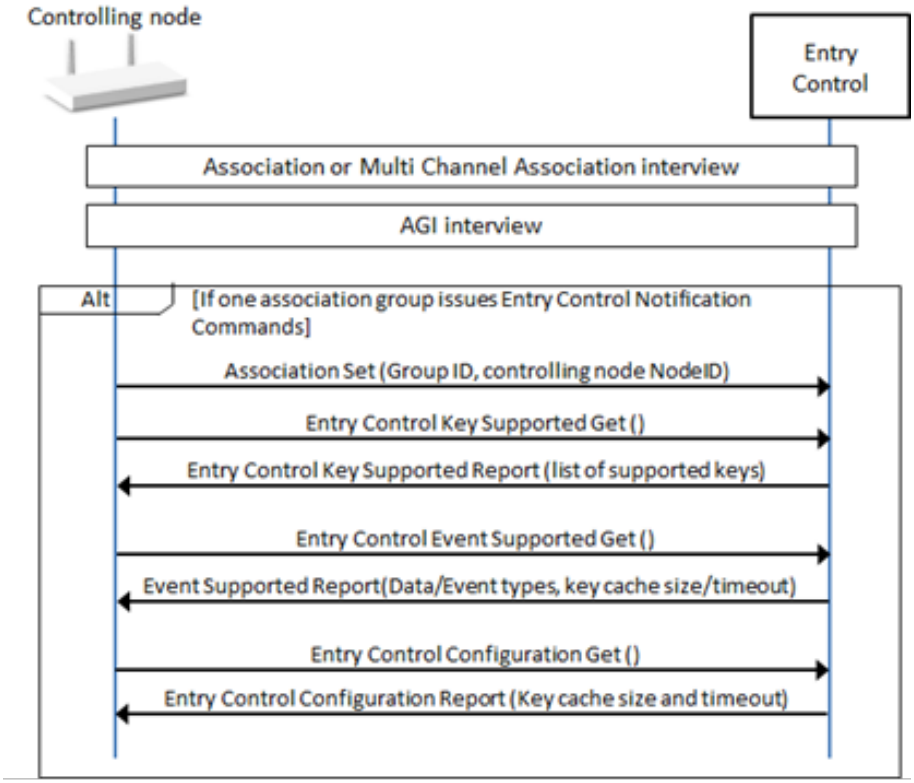


Figure 6.9: Entry Control Command Class interview

CL:006F.01.23.01.1 A node controlling this command class MAY skip the AGI Interview (refer to Section 6.3.2) if the supporting node is a Z-Wave Plus node and the controlling node has estalished a Lifeline Association.

6.2.9.2 Minimum end user functionalities

CL:006F.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.9.2.1 Configure the keypad

CL:006F.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.14.

Table 6.14: Entry Control Keypad::Configure the keypad

Field	Value
Command	ENTRY_CONTROL_CONFIGURATION_SET
Key Cache Size	User defined among supported values
Key Cache Timeout	User defined among supported values

6.2.9.3 Node properties

CL:006F.01.41.01.1 A controlling node MUST have a UI allowing the end user to see the received Entry Control Notifications (Event type and data)

6.2.9.4 Additional control requirements

CL:006F.01.51.01.1 A node controlling this command class MUST also control:

- Association Command Class, version 2
- Association Group Information, version 3

CL:006F.01.51.02.1 A controlling node MUST associate itself to a group issuing Entry Control Notification Commands before performing a supporting node interview and providing end user functionalities.

CL:006F.01.51.03.1 A controlling node MUST NOT remove associations in order to associate itself to an association group issuing Entry Control Notification Commands.

CL:006F.01.52.01.1 A controlling node SHOULD NOT provide end user functionalities if it cannot associate itself to an association group sending Entry Control Notification Commands. (e.g. all Association Groups sending the relevant command are full)

CL:006F.01.52.02.1 A controlling node SHOULD have a UI allowing the end user to define what actions to take based on received Entry Control Notifications.

CL:006F.01.52.03.1 A controlling node SHOULD also control Door Lock Command Class and Barrier Operator Command Class. It SHOULD also allow the user to set the door mode or initiate opening/closing of a given node based on received Entry Control Notifications.

6.2.10 IR Repeater Command Class, version 1

6.2.10.1 Mandatory node interview

CL:00A0.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.10.

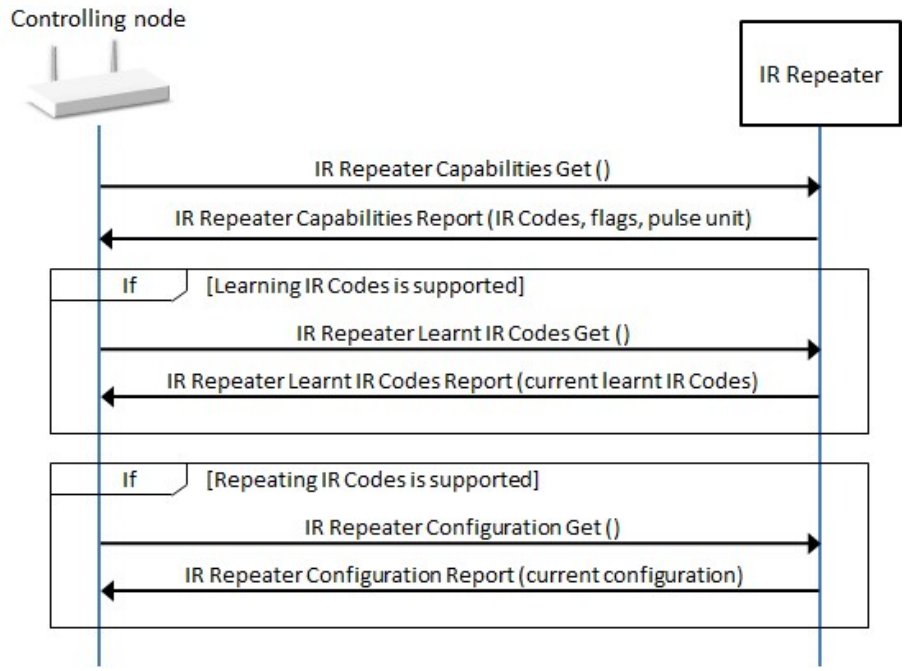


Figure 6.10: IR Repeater Command Class interview

6.2.10.2 Minimum end user functionalities

CL:00A0.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.10.2.1 Repeat an IR code

CL:00A0.01.31.02.1 This action MUST be available to the end user if the supporting node supports repeating IR Codes.

CL:00A0.01.31.03.1 When the end user performs this action, the issued commands MUST comply with Table 6.15.

Table 6.15: IR Repeater::Repeat an IR Code

Field	Value
Command	IR_REPEATER_REPEAT
Sequence number	Controlling node defined. It MUST be incremented at each new repeat action
Sub carrier	Controlling node defined within the capabilities of the supporting node
Duty Cycle	Controlling node defined within the capabilities of the supporting node
Pulse time unit	Controlling node defined within the range advertised by the supporting node in the IR Repeater Capabilities Report Command
Report Number	Controlling node defined.
Last	1 for the last command, 0 else.
Data	Controlling node defined. The end user MUST be able to select an IR Code that will be transferred.

6.2.10.2.2 Learn an IR code

CL:00A0.01.31.04.1 This action MUST be available to the end user if the supporting node supports Learning IR Codes.

CL:00A0.01.31.04.1 When the end user performs this action, the issued command MUST comply with Table 6.16.

Table 6.16: IR Repeater::Learn an IR Code

Field	Value
Command	IR_REPEATER_IR_CODE_LEARNING_START
IR Code identifier	User defined among available IR Codes slots
Timeout	Controlling node or User defined.
Pulse time unit	Controlling node defined within the range advertised by the supporting node in the IR Repeater Capabilities Report Command

The Controlling node MUST inform the user about the status of the IR Code learning when the IR Code Learning Status Command is received.

The controlling node MAY allow the end user to interrupt the learning process, if it does, the issued command MUST comply with Table 6.17.

Table 6.17: IR Repeater::Stop learning an IR code

Field	Value
Command	IR_REPEATER_IR_CODE_LEARNING_STOP

6.2.10.2.3 Erase a learnt IR code

CL:00A0.01.31.05.1 This action MUST be available to the end user if the supporting node supports Learning IR Codes.

CL:00A0.01.31.06.1 When the end user performs this action, the issued command MUST comply with Table 6.18.

Table 6.18: IR Repeater::Erase an IR Code from memory

Field	Value
Command	IR_REPEATER_LEARNT_IR_CODE_REMOVE
IR Code identifier	User defined among available IR Codes slots that have a defined IR Code

6.2.10.2.4 Repeat a learnt IR code

CL:00A0.01.31.07.1 This action MUST be available to the end user if the supporting node supports Repeating IR Codes.

CL:00A0.01.31.08.1 When the end user performs this action, the issued command MUST comply with Table 6.19.

Table 6.19: IR Repeater::Repeat a learnt IR code

Field	Value
Command	IR_REPEATER_REPEAT_LEARNT_CODE
IR Code identifier	User defined among available Learnt IR Codes slots

6.2.10.3 Node properties

If the supporting node supports IR Code learning:

CL:00A0.01.41.01.1 • A controlling node MUST have a UI allowing the end user to see the which IR Code Identifiers contain a valid learnt code.

6.2.11 Meter Command Class, version 1-5

6.2.11.1 Mandatory node interview

CL:0032.01.21.01.2 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.11.

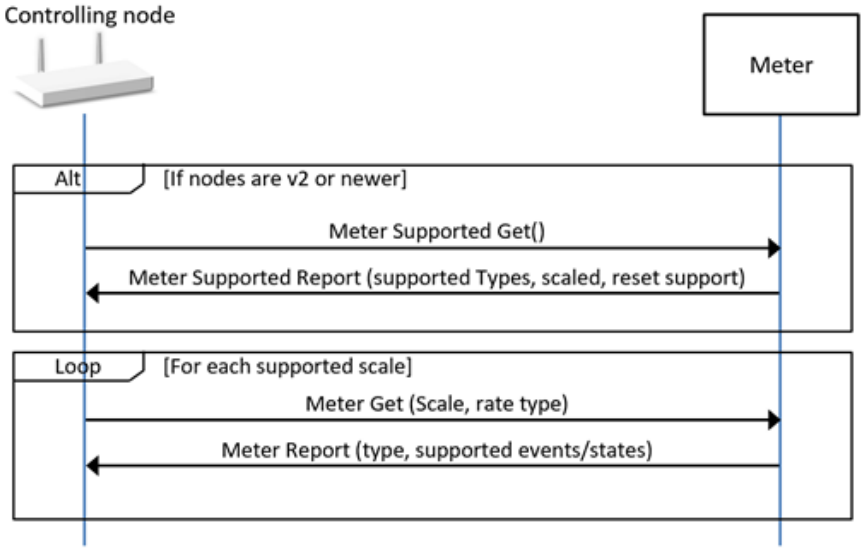


Figure 6.11: Meter Command Class interview

6.2.11.2 Minimum end user functionalities

CL:0032.02.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.11.2.1 Reset cumulated data

CL:0032.02.31.02.1 This action MUST be available to the end user if both nodes implement version 2 or newer and the supporting node indicates that meter reset is supported.

CL:0032.02.31.03.1 When the end user performs this action, the issued command MUST comply with Table 6.20.

Table 6.20: Meter::Reset cumulated data

Field	Value
Command (v2)	METER_RESET

6.2.11.3 Node properties

CL:0032.01.41.01.2 A controlling node MUST have a UI allowing the end user to see the last readings for each supported scale and all cumulated values. For version 1 supporting nodes, the scale reported in the Meter Report MUST be accessible to the end user.

CL:0032.01.41.02.1 A controlling node MUST always show the value even if the Type and/or Scale are unknown.

CL:0032.01.42.01.1 A controlling node SHOULD implement the capability to update its list of Meter Type and Scales, so that new Meter Types and scales added in more recent versions are not presented as unknown.

CL:0032.01.41.03.1 If a controlling node receives an unknown Meter Type or Scale, it MUST allow the end user to identify the Meter reading and MAY allow the user to assign a free-text description to that Meter reading

6.2.11.4 Additional control requirements

CL:0032.01.51.01.1 Unless unsolicited Meter Report Commands are received, a controlling node MUST:

- Probe the current meter readings for each supported scale (and rate type if v4 nodes) at least every 6 hours for listening nodes.
- Probe the current meter readings for each supported scale (and rate type if v4 nodes) when the supporting node issues a Wake Up Notification Command for sleeping nodes.

6.2.12 Multilevel Sensor Command Class, version 1-11

6.2.12.1 Mandatory node interview

CL:0031.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.12.

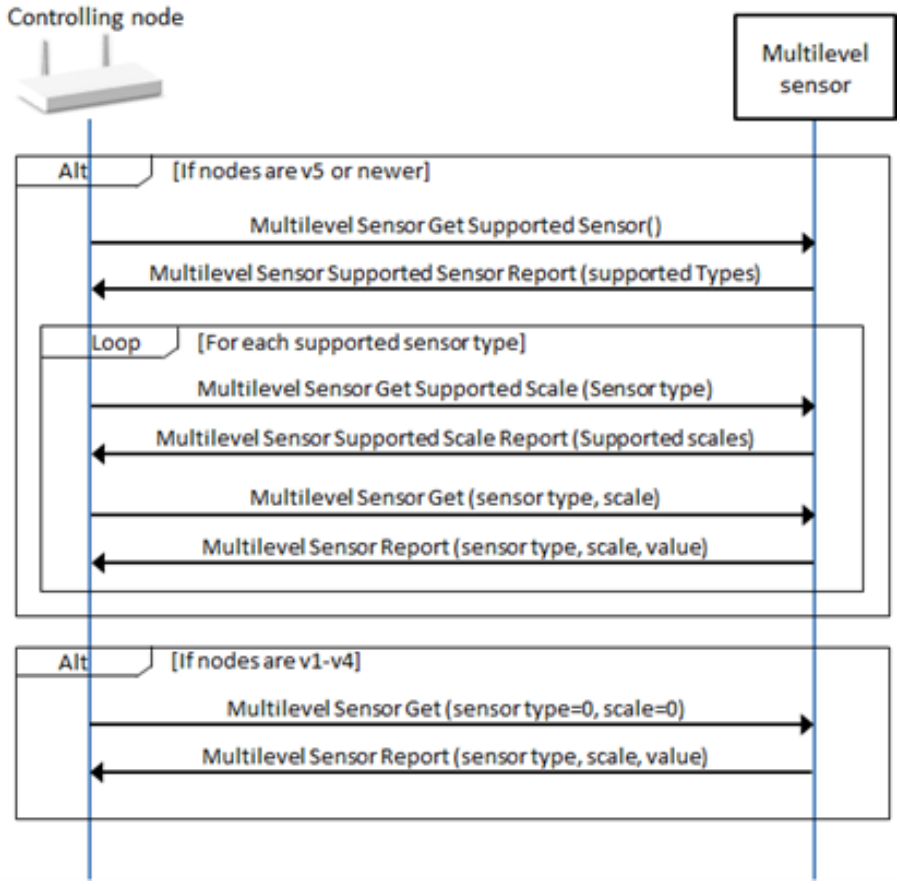


Figure 6.12: Multilevel Sensor Command Class interview

6.2.12.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.2.12.3 Node properties

CL:0031.01.41.01.1 A controlling node MUST allow the end user to see the received readings for every Sensor types

CL:0031.01.41.02.1 A controlling node MUST always show the sensor reading values even if the Sensor Type and/or Scale are unknown.

CL:0031.01.42.01.1 A controlling node SHOULD implement the capability to update its list of Sensor Type and Scales, so that new Sensor Types and Scales added in [27] are not presented as unknown.

CL:0031.01.41.03.2 If a controlling node receives an unknown Sensor Type or Scale, it MUST allow the end user to identify the Sensor reading and MAY allow the end user to assign a free-text description to the sensor reading.

6.2.12.4 Additional control requirements

CL:0031.01.51.01.1 Unless unsolicited Multilevel Sensor Report Commands are received, a controlling node MUST:

- Probe the current reading for each supported sensor type at least every 6 hours for listening nodes.
- Probe the current reading for each supported sensor type when the supporting node issues a Wake Up Notification Command for sleeping nodes.

CL:0031.01.52.01.1 A node controlling this command class SHOULD also control:

- Association Command Class, version 2
- Association Group Information, version 3

CL:0031.01.52.02.1 A controlling node SHOULD associate itself to an association group issuing Multilevel Report Commands after performing a supporting node interview and providing end user functionalities.

CL:0031.01.51.02.1 A controlling node MUST NOT remove associations in order to associate itself to an association group issuing Multilevel Report Commands.

CL:0031.01.52.03.1 Controller SHOULD have a UI allowing the end user to define rules or commands based on received readings.

6.2.13 Multilevel Switch Command Class, version 1-4

6.2.13.1 Mandatory node interview

CL:0026.01.21.01.2 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.13. It is OPTIONAL to issue the *Multilevel Switch Supported Get Command* if a controlling node does not use this information for its UI.

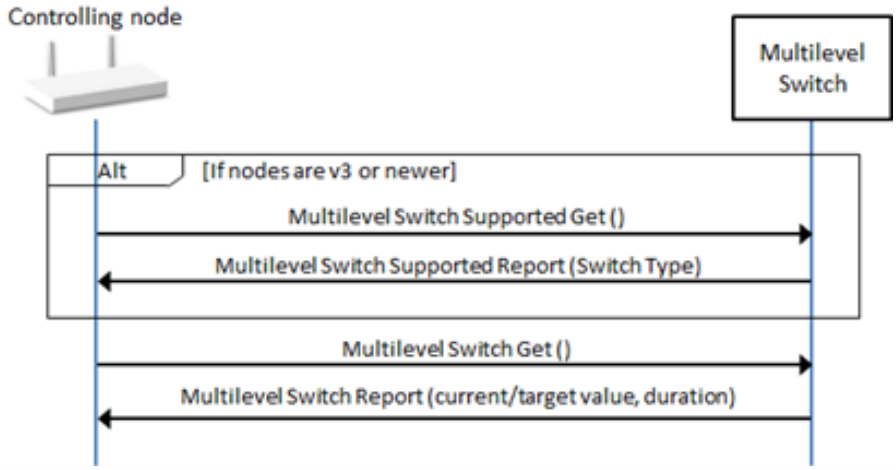


Figure 6.13: Multilevel Switch Command Class interview

6.2.13.2 Mimimum End User Functionalities

CL:0026.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.13.2.1 Set the Level

CL:0026.01.31.02.2 When the end user performs this action, the issued command MUST comply with Table 6.21.

Table 6.21: Multilevel Switch::Set the level

Field	Value
Command	SWITCH_MULTILEVEL_SET
Value	User defined among 0x00..0x63. Using 0xFF is optional.
Duration (v2)	User defined or 0xFF

6.2.13.3 Node Properties

CL:0026.01.42.01.1 Controller SHOULD have a UI allowing the end user to see the following properties:

- Last known state (xx%)

6.2.14 Notification Command Class, version 1-8

6.2.14.1 Mandatory Node Interview

CL:0071.02.21.01.2 A node controlling this command class MUST perform a supporting node interview according to Figure 6.14.

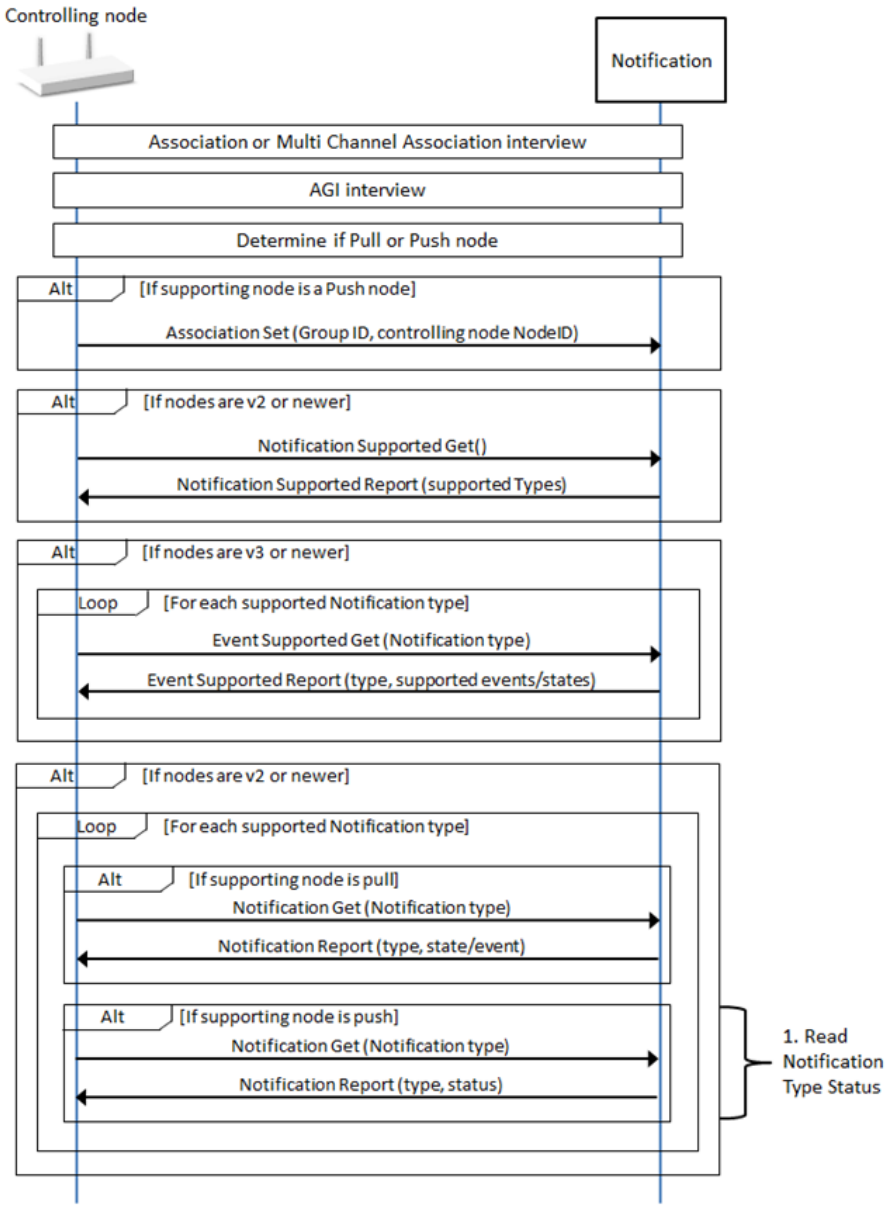


Figure 6.14: Notification Command Class interview

CL:0071.02.23.01.1 The step denoted as "1. Read Notification Type Status" in Figure 6.14 MAY be replaced by Notification Set commands for each Notification Type, with the Status field set to 0x00 or 0xFF.

CL:0071.02.23.02.1 A node controlling this command class MAY skip the AGI Interview (refer to Section 6.3.2) if the supporting node is a Z-Wave Plus node and the controlling node has estalished a Lifeline Association.

6.2.14.2 Minimum End User Functionalities

No minimum end user functionality is required for this Command Class.

6.2.14.3 Node Properties

CL:0071.01.41.01.1 A controlling node MUST have a UI allowing the end user to see the received Notification Reports (Notification type and event/state).

6.2.14.4 Additional Control Requirements

CL:0071.01.51.01.1 A controlling node MUST probe state/event notifications from pull nodes

- at least every 6 hours for listenting nodes.
- when the supporting node issues a Wake up Notification Command for sleeping nodes.

CL:0071.01.51.02.1 A node controlling this command class MUST also control:

- Association Command Class, version 2
- Association Group Information, version 3

CL:0071.01.51.03.1 For Push supporting nodes, a controlling node MUST associate itself to a group issuing Notification Report Commands before performing a supporting node interview and providing end user functionalities.

CL:0071.01.53.01.1 For Push supporting nodes, it is OPTIONAL for a controlling node to provide end user functionalities and node properties if it cannot associate itself to an association group sending Notification Report Commands. (e.g. all Association Groups sending the relevant command are full)

CL:0071.01.51.04.1 A controlling node MUST NOT remove associations in order to associate itself to an association group issuing Notification Report Commands.

CL:0071.01.52.01.1 Controller SHOULD have a UI allowing the end user to define rules or commands based on received notifications events/states.

CL:0071.01.52.02.1 A controlling node SHOULD implement the capability to update its Notifications list, so that new Notifications added in [36] are not presented as unknown.

CL:0071.01.51.05.2 If a controlling node receives an unknown Notification, it MUST allow the end user to identify this notification and MAY allow the end user to assign a free-text description to that Notification.

6.2.14.4.1 Detailed Controller Guidelines

This section presents guidelines for controlling legacy sensors nodes supporting the Notification Command Class, version 3-8.

Alarm Command Class version 2 supporting nodes operate in Push mode only. The guidelines presented in Section 6.2.14.4.7 Controlling Pull Mode Sensors can also be used for controlling Alarm CC version 2 supporting nodes.

6.2.14.4.2 Sensor database

The following sections present discovery methods for sensor properties, such as Push/Pull mode or the use of state idle notifications. Some sensor properties (e.g. configuration parameters, Association group properties, state variables dependencies) cannot always be discovered.

CL:0071.01.52.03.1 If a controlling node has a database of known certified Notification sensors, it SHOULD be consulted first before attempting to interview the Notification Command Class of a supporting node. If the supporting node is present in the database, a controlling node SHOULD read the sensor properties from the database and skip any discovery.

6.2.14.4.3 Push/Pull mode discovery

A supporting node does not advertise whether it operates in Push or Pull mode. Therefore a controlling
CL:0071.01.51.06.1 node MUST discover what mode a supporting node implements.

CL:0071.01.52.04.1 The RECOMMENDED discovery steps for a controlling node are the following:

If the supporting node does not support the Association Command Class, it may be concluded that the supporting node implements Pull Mode and discovery may be aborted.

If the supporting node supports the AGI Command Class, probe the AGI table in the following way or read the already probed AGI table:

1. Discover how many Association groups the supporting node (and its eventual End Points) implements by issuing an Association Supported Groupings Get Command.
2. For each association group, issue an Association Group Command List Get for the grouping identifier. Inspect the returned Association Group Command List Report and look for the following pair: {Command Class x = COMMAND_CLASS_NOTIFICATION, Command x = NOTIFICATION_REPORT}. If a match is found, conclude that the supporting node implements Push Mode and stop the discovery process.

If no match is found at the end of the AGI test, conclude that the supporting node implements Pull Mode. The AGI test is illustrated in Figure 6.15.

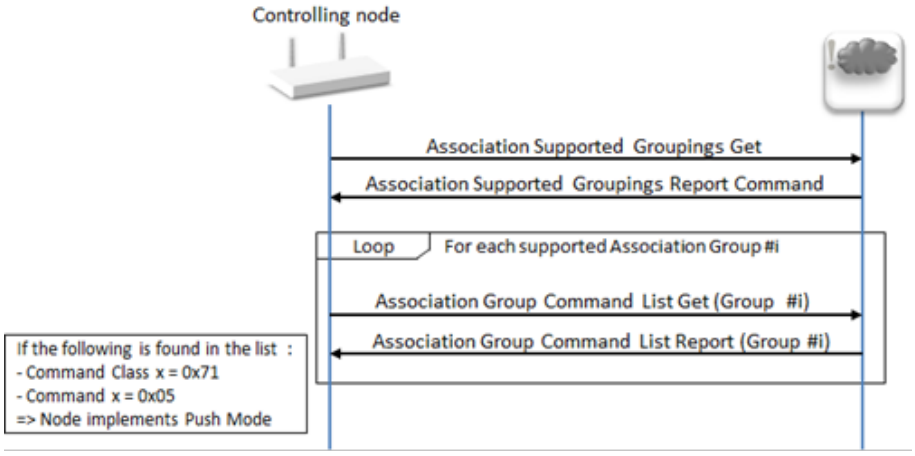


Figure 6.15: Push/Pull Node Discovery, AGI table probing

If the supporting node does not support the AGI Command Class, proceed with the following Notification Command Class test:

1. Discover the list of supported Notification Types by issuing a Notification Supported Get Command.
2. For each supported Notification Type, query the supported Events/States via the Event Supported Get Command.

3. Set the status of one of the Supported Notification Types at the target node using a Notification Set Command with the following values:
   a. Notification Type = (a supported type discovered in step 1)
   b. Notification Status = 0xFF
4. Issue a Notification Get Command with the following values:
   a. Notification Type = (Same as step 3.a).
   b. Notification Event = (a supported Event/State discovered in step 2).
5. The supporting node returns a Notification Report Command.
   a. If the Notification Status is 0xFF, the target node implements Push Mode
   b. If the Notification Status is 0xFE or 0x00, the target node implements Pull Mode
   c. If the target node did not return a report within 10 seconds, it implements Pull Mode

The Notification Command Class test is illustrated in Figure 6.16.

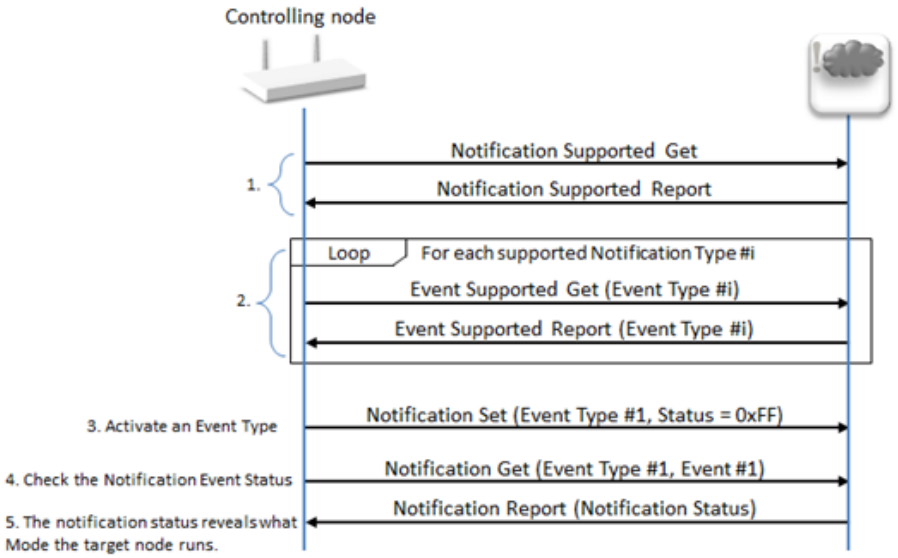


Figure 6.16: Push/Pull Node Discovery, Notification Command Class test

6.2.14.4.4 Unknown Notifications

CL:0071.01.52.05.1 This Command Class defines some Notifications as event or states. It is RECOMMENDED to treat an unknown Notification as its own state variable.

6.2.14.4.5 State idle

CL:0071.01.52.06.1 A controlling node receiving a "State idle" Notification for an event or state of which the "State idle" Notification does not apply (e.g. the "heartbeat" event) SHOULD ignore the Notification. Refer to [36] for state variables to which the "State idle" notification applies.

It is not mandatory for nodes supporting Notification Command Class, version 7 or older to send a "State idle" Notification for returning a state variable to idle. Thus, there is a risk that a "State idle" CL:0071.01.52.07.2 is never sent. A controlling node SHOULD NOT use timeouts to consider a state variable to be idle CL:0071.01.53.02.1 for example 5 minutes after the last received Notification for v8 or newer supporting nodes. It MAY allow the end user to mark states variables as idle.

6.2.14.4.6 Parameter encapsulation

CL:0071.01.53.03.1 A node implementing version 5 or older MAY use the following parameter encapsulation format: (User ID, User Code Report Command) instead of (User Code Report Command).

CL:0071.01.52.08.1 A controlling node SHOULD accept the end user Code Report Event Parameter with a User ID byte prefixed at the beginning of the Notification Event Parameter for backwards compatibility. A controlling node can detect the beginning of the end user Code Report by finding the following consecutive bytes: {byte x= COMMAND_CLASS_USER_CODE, byte x+1 = USER_CODE_REPORT}.

6.2.14.4.7 Controlling Pull Mode Sensors

The following sections present guidelines for controlling Pull sensors nodes.

6.2.14.4.8 Notification Get Command

This command is used to retrieve the next Notification from the receiving node's queue. Below are recommendations for what values controlling nodes should set in the fields of this command.

Notification Type (8 bits)

CL:0071.01.52.09.1 A controlling node SHOULD set the Notification Type field to 0xFF.

Earlier specification text suggested that Pull nodes must not ignore this command if the Notification Type is set to a supported type; rather than 0xFF. Therefore, Pull sensors may respond to a Notification Get for a supported Notification type by retrieving the next notification in the queue matching the specified Notification Type.

Notification Event /State (8 bits)

CL:0071.01.52.0A.1 A controlling node SHOULD set this field to 0x00.

A receiving node will have an unpredictable behavior if this field is set to a supported Event.

6.2.14.4.9 Detecting and clearing persistent notifications

It is optional for Pull nodes to specify a sequence number in notifications.

CL:0071.01.52.0B.1 A controlling node receiving twice the same Notification with identical sequence number SHOULD consider the Notification as persistent. An example is given in Figure 6.17.

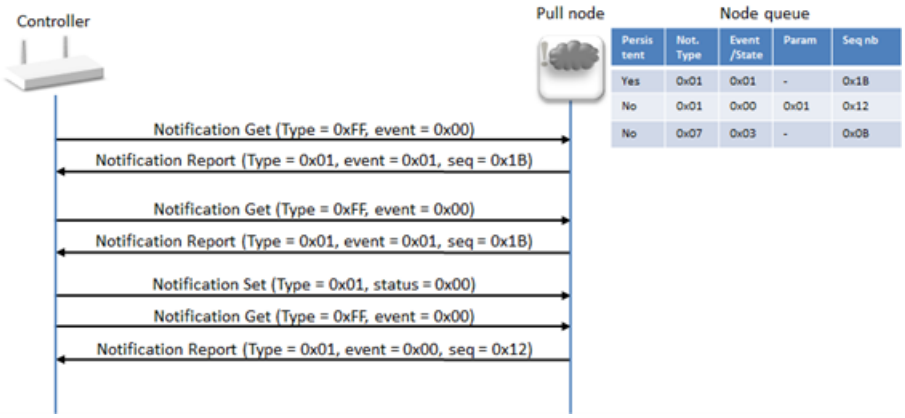


Figure 6.17: Detecting and Clearing Persistent Notifications, identical sequence numbers

CL:0071.01.52.0C.1 Due to unclear specification text, sensors may issue persistent Notifications without a sequence number field or with a new sequence number every time. Therefore, a controlling node SHOULD consider

a notification to be persistent if it receives the same Notification (type and event/state) 5 times or more. An example is given in Figure 6.18.

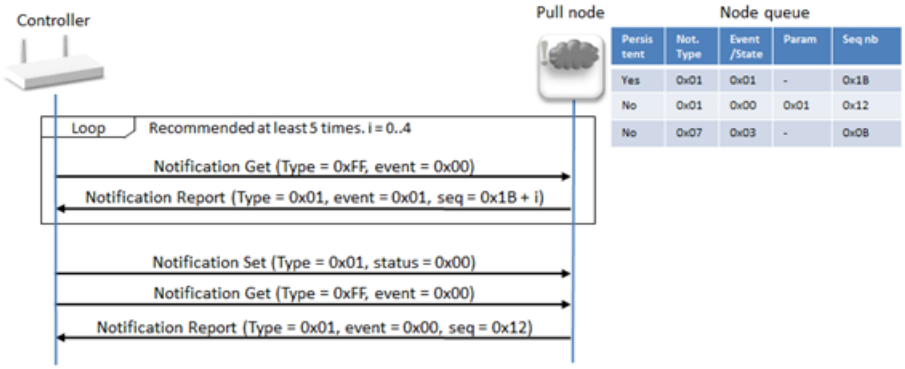


Figure 6.18: Detecting and clearing persistent notifications with incremental sequence numbers

6.2.14.4.10 Status and queue empty

CL:0071.01.52.0D.1 A controlling node SHOULD stop issuing Notification Get Commands when receiving a Queue empty notification (0xFE). Illustration is given in Figure 6.19.

CL:0071.01.53.04.1 All fields following the Status field MAY be omitted in the Notification Report if it advertises a Status of 0xFE (queue empty).

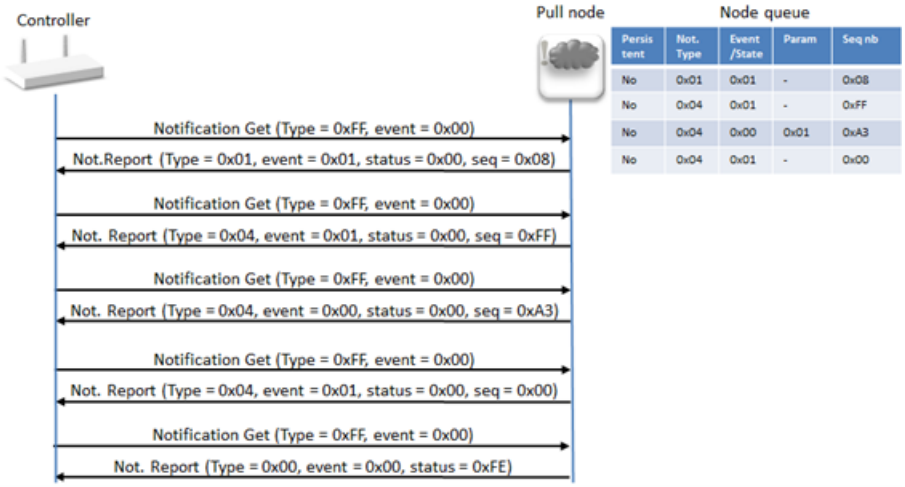


Figure 6.19: Pull node event retrieval until receiving queue empty

6.2.15 Simple AV Control Command Class, version 1-4

6.2.15.1 Mandatory Node interview

CL:0094.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.20.

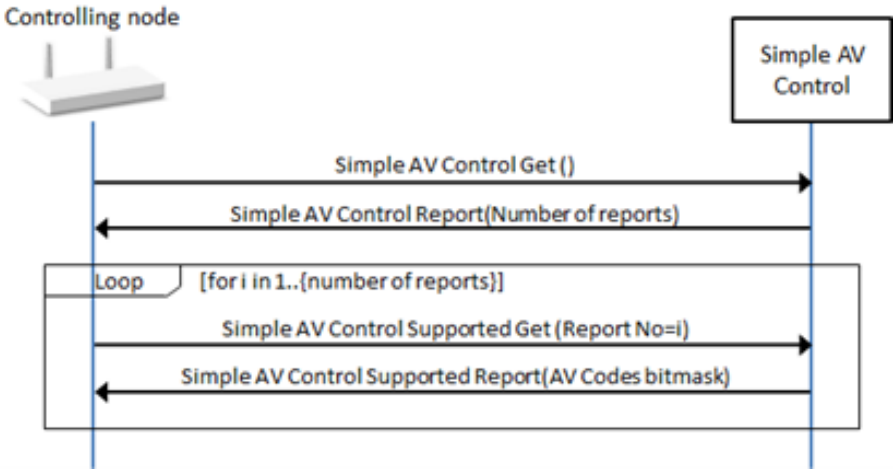


Figure 6.20: Simple AV Control Command Class interview

6.2.15.2 Minimum end user functionalities

CL:0094.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.15.2.1 Send a supported AV code

CL:0094.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.22.

Table 6.22: Simple AV Control::Send a supported AV code

Field	Value
Command	SIMPLE_AV_CONTROL_SET
Sequence number	Controlling node defined. This field MUST be incremented every time the end user perform this action
Key attributes	User defined
Command x	User defined among supported AV codes

CL:0094.01.31.03.1 The end user MUST be able to select among all supported AV codes even if the controlling node does not know what the code actually represents.

6.2.15.3 Node properties

CL:0094.01.41.01.1 Controller SHOULD have a UI allowing the end user to see the following properties:

- List of supported AV Codes

6.2.16 Sound Switch Command Class, version 1

6.2.16.1 Mandatory node interview

CL:0079.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.21.

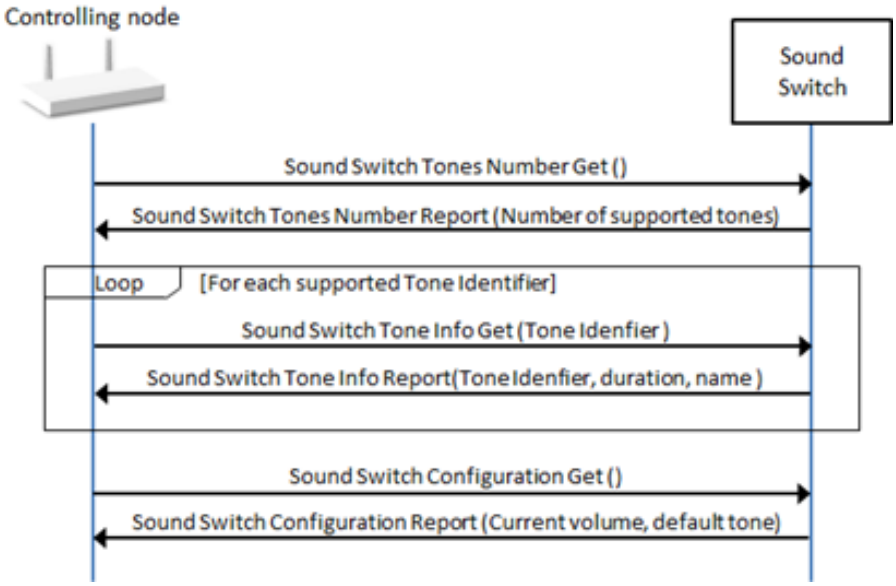


Figure 6.21: Sound Switch Command Class interview

6.2.16.2 Minimum end user functionalities

CL:0079.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.16.2.1 Configure the sound switch

CL:0079.01.31.02.1 The end user MUST be able to configure the volume and default tone at the supporting node. When the end user performs this action, the issued command MUST comply with Table 6.23.

Table 6.23: Sound Switch::Configure the sound switch

Field	Value
Command	SOUND_SWITCH_CONFIGURATION_SET
Volume	User defined among 0x00..0x64 or 0xFF
Default Tone identifier	User defined among supported or 0x00

6.2.16.2.2 Play/stop a selected or default tone

CL:0079.01.31.03.1 The end user MUST be able to play the default tone, play a selected tone or stop playing a tone. When the end user performs this action, the issued command MUST comply with Table 6.24.

Table 6.24: Sound Switch::Play/stop a selected or default tone

Field	Value
Command	SOUND_SWITCH_TONE_PLAY_SET
Tone identifier	User defined among supported tones, 0x00 and 0xFF.

6.2.16.3 Node properties

CL:0079.01.42.01.1 Controller SHOULD allow the end user to see/access the following properties:

- (Last known) configured volume
- (Last known) configured default tone
- List of supported tones and their duration

6.2.17 Thermostat Mode Command Class, version 1-3

6.2.17.1 Mandatory interview

CL:0040.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.22.

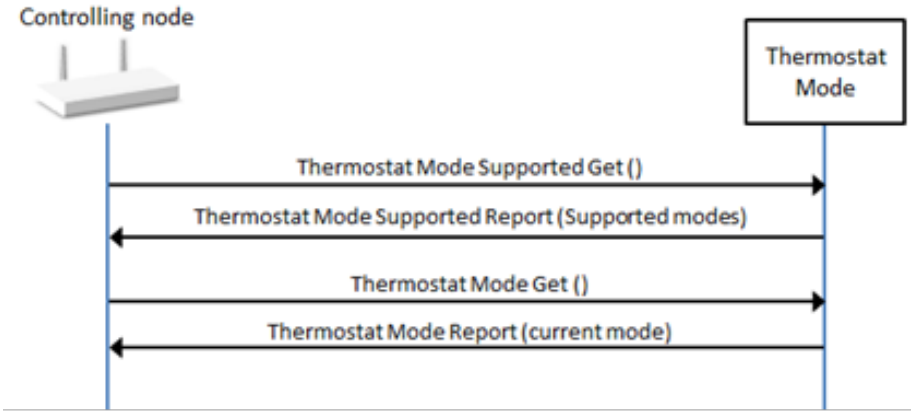


Figure 6.22: Thermostat Mode Command Class interview

6.2.17.2 Minimum end user functionalities

CL:0040.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.17.2.1 Change mode

CL:0040.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.25.

Table 6.25: Thermostat Mode::Change mode

Field	Value
Command	THERMOSTAT_MODE_SET
Mode	User defined among supported modes

CL:0040.01.31.03.1 The end user MUST be able to select between all supported modes by the supporting node, even if the controlling node does not know what a given mode represents.

6.2.17.3 Node properties

CL:0040.01.42.01.1 A controlling node SHOULD have a UI allowing the end user to see the following properties:

- Last known mode

6.2.18 Thermostat Setback Command Class, version 1

6.2.18.1 Mandatory interview

CL:0047.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.23.

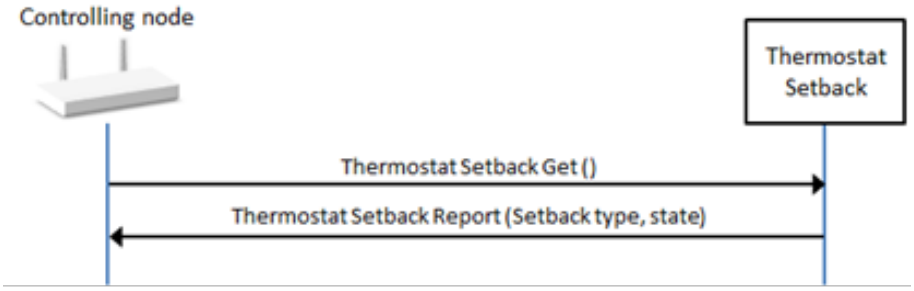


Figure 6.23: Thermostat Setback Command Class interview

6.2.18.2 Minimum end user functionalities

CL:0047.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.18.2.1 Configure setback

CL:0047.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.26.

Table 6.26: Thermostat Setback::Change setback for a supported type

Field	Value
Command	THERMOSTAT_SETBACK_SET
Setback Type	User defined among 0x00..0x02
Setpoint State	User defined among 0x00..0x7A and 0x80..0xFF

6.2.18.3 Node properties

CL:0047.01.42.01.1 Controller SHOULD have a UI allowing the end user to see the following properties:

- Last known setback type and state.

6.2.19 Thermostat Setpoint Command Class, version 1-3

6.2.19.1 Mandatory interview

CL:0043.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.24.

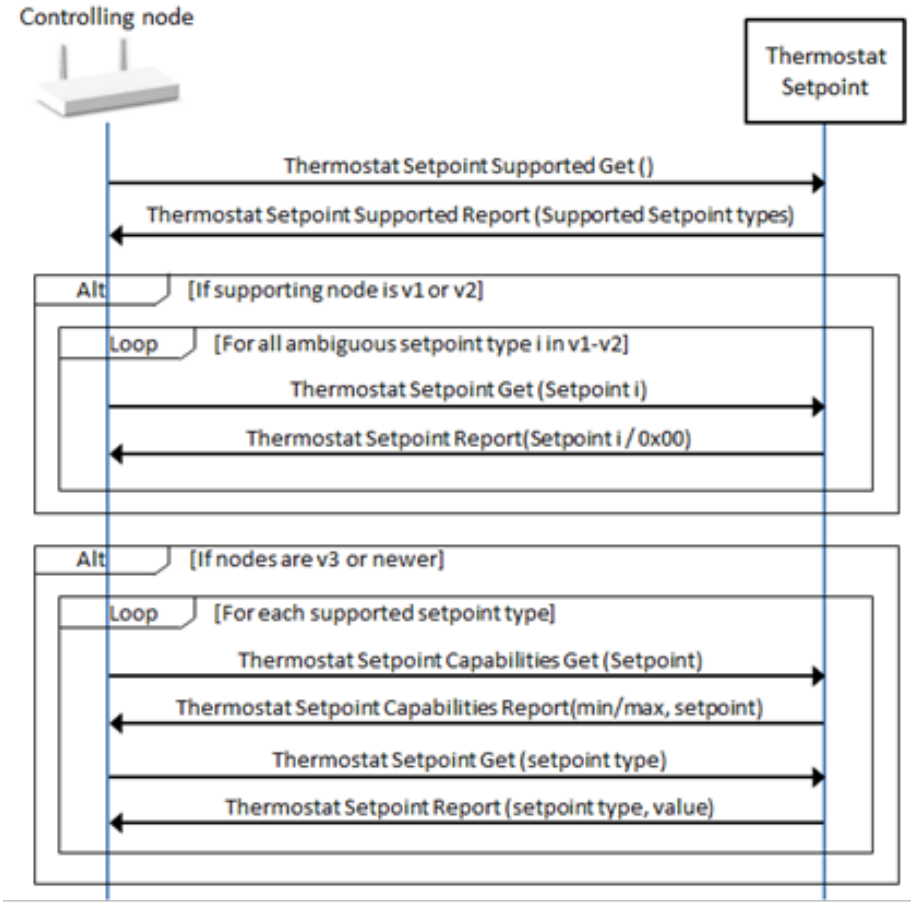


Figure 6.24: Thermostat Setpoint Command Class interview

It has been found that early implementations of this Command Class applied two non-interoperable interpretations of the bit mask advertising the support for specific Setpoint Types in the Thermostat Setpoint Supported Report Command.

Refer to the Thermostat Setpoint Command Class definition (Section 2.2.114, Section 2.2.115) for the possible bitmask interpretations.

CL:0043.01.22.01.1 A controlling node SHOULD determine the supported Setpoint Types of a version 1 and version 2 supporting node by sending one Thermostat Setpoint Get Command at a time while incrementing the requested Setpoint Type.

CL:0043.01.21.03.1 If the same Setpoint Type is advertised in the returned Thermostat Setpoint Report Command, the controlling node MUST conclude that the actual Setpoint Type is supported.

CL:0043.01.21.04.1 If the Setpoint Type 0x00 (type N/A) is advertised in the returned Thermostat Setpoint Report Command, the controlling node MUST conclude that the actual Setpoint Type is not supported.

6.2.19.2 Minimum end user functionalities

CL:0043.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.19.2.1 Change setpoint for a supported type

CL:0043.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.27.

Table 6.27: Thermostat Setpoint::Change setpoint for a supported type

Field	Value
Command	THERMOSTAT_SETPOINT_SET
Setpoint Type	User defined among supported
Setpoint Value	If nodes are version 3 or newer: User defined among supported values If nodes are version 1 or 2: User defined.
Precision	User defined
Scale	Controller defined
Size	User defined

CL:0043.01.31.03.1 If controlling a version 1 or 2 supporting node, the controlling node MUST allow the user to define the Setpoint Value freely. The controlling node MUST read back the value with a Thermostat Mode Get(Setpoint type) or use Supervision Get encapsulation and indicate to the end user if the operation was successful.

CL:0043.01.31.04.1 The end user MUST be able to set the setpoint for any setpoint type, even if the controlling node does not know what a given type represents.

CL:0043.01.31.05.1 The Scale field value MUST be identical to the value received in the Thermostat Setpoint Report for the actual Setpoint Type during the node interview.

CL:0043.01.42.01.1 The controlling node SHOULD let the user define the Setpoint Value in their preferred scale but MUST convert the value into the supporting node's scale for issuing the Z-Wave Command.

6.2.19.3 Node properties

CL:0043.01.42.01.1 A controlling node SHOULD have a UI allowing the end user to see the following properties:

- Last known setpoint value for each supported setpoint type.

6.2.20 User Code Command Class, version 1-2

6.2.20.1 Mandatory interview

CL:0063.01.21.01.2 A node controlling this command class MUST perform a supporting node interview according to Figure 6.25.

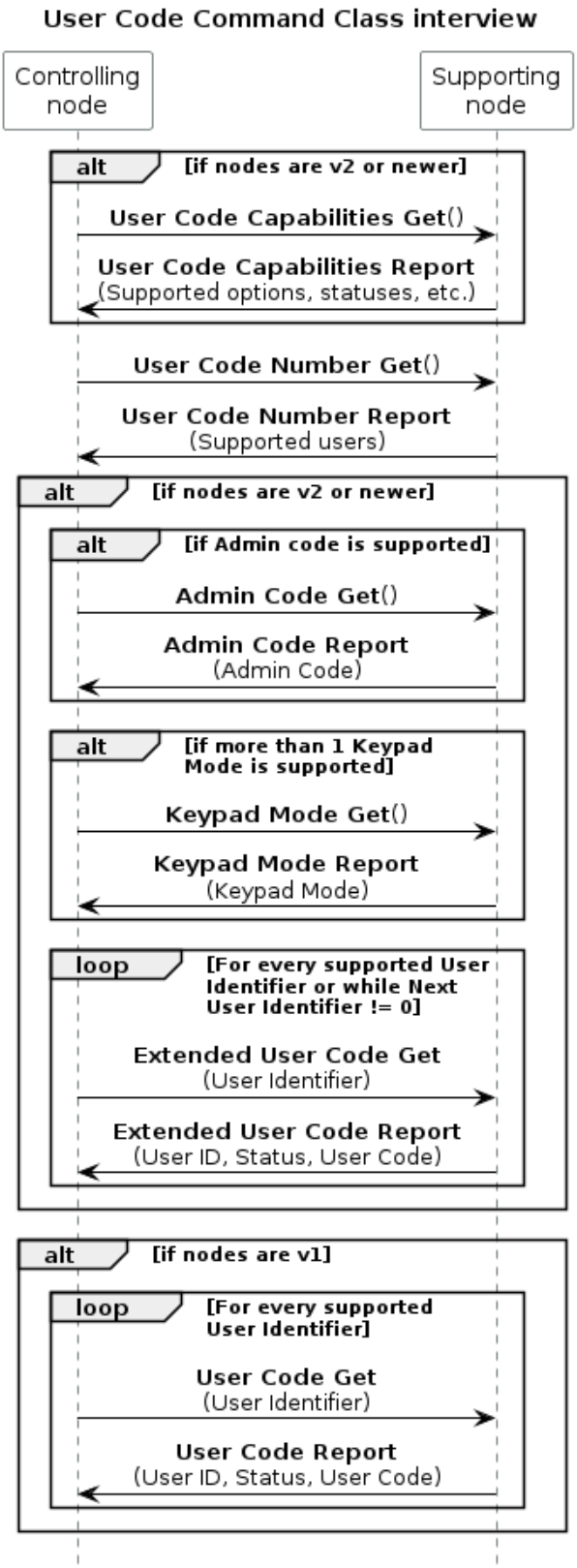


Figure 6.25: User Code Command Class interview

CL:0063.01.21.02.3 For a node controlling version 2 or newer:

- It is OPTIONAL to send an *Extended User Code Get Command* for every User Identifier to a node supporting version 2 or newer if:
  - The controlling node requested the checksum and it is set to 0, or
  - The controlling node issues an *Extended User Code Set Command* (User ID = 0, User ID Status = 0) to delete all user codes, or
  - The supporting node reports that no more User Identifiers are set in the *Extended User Code Report Command* with the *Next User Identifier* field.
- It is OPTIONAL to send a *User Code Get Command* for every User Identifier to a node supporting version 1 if the controlling node issues a *User Code Set Command* (User ID = 0, User ID Status = 0) to delete all user codes.

CL:0063.01.21.03.1 For a node controlling version 1:

- It is OPTIONAL to send a *User Code Get Command* for every User Identifier to any supporting node if the controlling node issues a *User Code Set Command* (User ID = 0, User ID Status = 0) to delete the user codes below User ID 256.

CL:0063.01.22.01.1 A controlling node SHOULD NOT automatically delete any user code unless it is the initial interview right after having included the supporting node in the network.

6.2.20.2 Minimum end user functionalities

CL:0063.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.20.2.1 Set/modify a User Code

CL:0063.01.31.02.1 When the end user performs this action, the issued command MUST comply with Table 6.28 if nodes are v1 or Table 6.29 if nodes are v2 or newer.

Table 6.28: User Code::Set a User Code

Field	Value
Command	USER_CODE_SET (0x01)
User Identifiers	User defined or controlling node defined among supported User Identifiers.
User ID Status	User defined among 0x01 and 0x02 0x02 must be used to set a reserved/forbidden user code.
User Code	User defined among 0x30..0x39 with a length in the range 4..10 bytes.

A forbidden or reserved User Code is a User Code that cannot be used at the supporting node and cannot be allocated to a new user, for example if User Code can also be updated locally via a user interface.

Table 6.29: User Code::Set a User Code (v2)

Field	Value
Command	EXTENDED_USER_CODE_SET (0x0B)
Number of User Codes (v2)	Controlling node defined
User Identifier 1..M (v2)	User defined or controlling node defined among supported User Identifiers.
User ID status (v2)	User defined among supported User ID statuses.
User Code Length (v2)	Controlling node defined based on the length of the User Code field.
User Code (v2)	User defined among supported ASCII characters with a length in the range 4..10 bytes.

6.2.20.2.2 Erase a User Code

CL:0063.01.31.03.1 When the end user performs this action, the issued command MUST comply with Table 6.30 if nodes are v1 and Table 6.31 if nodes are v2 or newer.

Table 6.30: User Code::Erase a User Code

Field	Value
Command	USER_CODE_SET
User Identifiers	User defined or controlling node defined among supported User Identifiers.
User ID Status	0x00
User Code	0x00000000

CL:0063.01.31.04.1 A controlling node MAY allow the end user to erase all user codes at once. In this case, the User Identifier field MUST be set to 0x00.

Table 6.31: User Code::Erase a User Code

Field	Value
Command	EXTENDED_USER_CODE_SET (0x0B)
Number of User Codes (v2)	Controlling node defined
User Identifier 1..M (v2)	User defined or controlling node defined among supported User Identifiers.
User ID status (v2)	0x00
User Code Length (v2)	0x00
User Code (v2)	Omitted

6.2.20.2.3 Set the keypad mode (v2)

CL:0063.01.31.05.1 This action MUST be available to the end user if nodes are v2 or newer and the supporting node supports more than one keypad mode. When the end user performs this action, the issued command MUST comply with Table 6.32.

Table 6.32: User Code::Set the Keypad Mode

Field	Value
Command (v2)	USER_CODE_KEYPAD_MODE_SET (0x08)
Keypad Mode (v2)	User defined among supported keypad modes.

6.2.20.2.4 Set the Admin Code (v2)

CL:0063.01.31.06.1 This action MUST be available to the end user if nodes are v2 or newer and the supporting node supports the Admin Code functionality. When the end user performs this action, the issued command MUST comply with Table 6.33.

Table 6.33: User Code::Set the Admin Code

Field	Value
Command (v2)	ADMIN_CODE_SET (0x0E)
Admin Code Length (v2)	Controlling node defined based on the length of the User Code field. It MUST be possible to de-activate the Admin Code if the supporting node supports Admin Code Deactivation.
Admin Code (v2)	User defined among supported ASCII characters with a length in the range 4..10 bytes.

6.2.20.3 Node properties

CL:0063.01.41.01.1 Controller MUST have a UI allowing the end user to see the following properties:

- Number of supported User Codes
- The list of last known set User Codes
- The current keypad mode, if the supporting node supports more than one (v2)
- The current set Admin Code, if the supporting node supports a Admin Code (v2)

6.2.20.4 Additional control requirements

It has been found that some version 1 nodes wrongfully report obfuscated User Codes in the User Code Report (e.g. '******').

CL:0063.01.52.01.1 A controlling node SHOULD understand that a code has been set correctly but cannot be read back with such nodes.

CL:0063.01.52.02.1 If nodes are version 2 or newer, a controlling node SHOULD verify the User Code checksum (if supported) periodically (e.g. once a day) to ensure that User Code databases are synchronized.

6.2.21 User Credential Command Class, version 1

6.2.21.1 Mandatory interview

CL:0083.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.26.

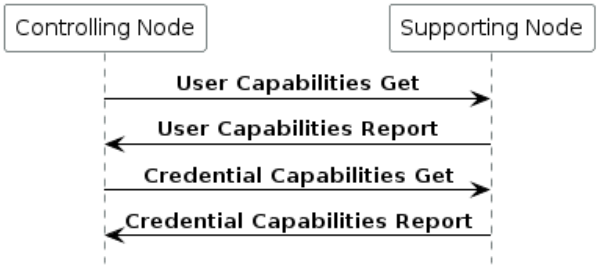


Figure 6.26: User Credential Command Class interview

CL:0083.01.22.02.1 If the All Users Checksum Support field is set in the *User Capabilities Report Command*, the controlling node SHOULD send an *All Users Checksum Get Command* to check if there are existing Users or Credentials present on the supporting node.

CL:0083.01.22.03.1 If the supporting node does not support the *All Users Checksum Get Command*, or if it does and the All Users Checksum field of the *All Users Checksum Report Command* is non-zero, the controlling node SHOULD perform an interview of the existing Users and Credentials according to Figure 6.27.

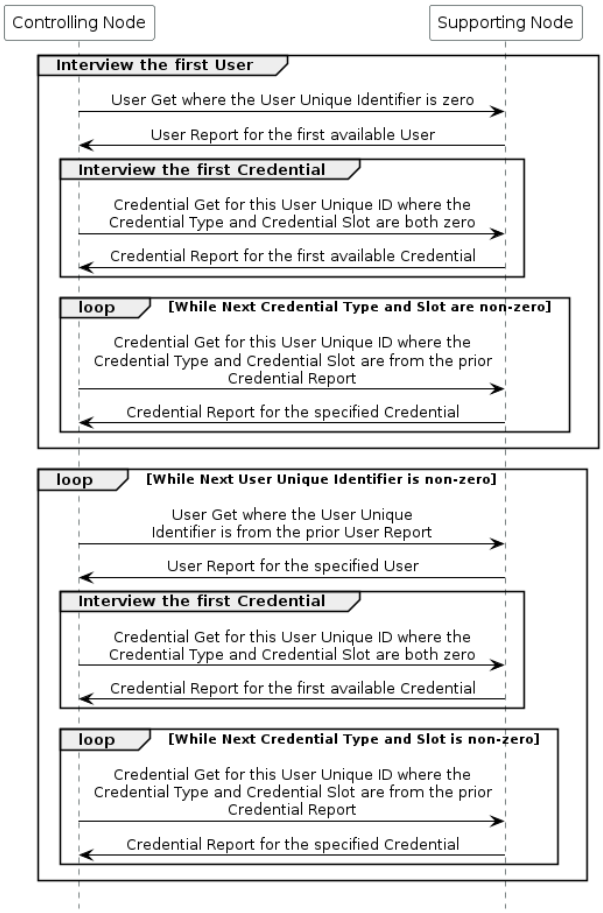


Figure 6.27: User Credential Command Class optional interview

6.2.21.2 Minimum end user functionalities

CL:0083.01.31.04.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.21.2.1 Add/Modify a User

CL:0083.01.31.05.1 When the end user performs this action, the issued command MUST comply with Table 6.34.

Table 6.34: User Credential::Add/Modify a User

Field	Value
Command	USER_SET (0x05)
Operation Type	Controlling node defined as Add (0x00) or Modify (0x01).
User Unique Identifier	User defined or controlling node defined among supported User Identifiers in the range of what was specified as supported in the *User Capabilities Report Command*.
User Type	User defined or controlling node defined among the Supported User Types as specified in the *User Capabilities Report Command*.
User Active State	User defined or controlling node defined.
Credential Rule	User defined or controlling node defined among the Supported Credential Rules as specified in the *User Capabilities Report Command*.
Expiring Timeout Minutes	User defined or controlling node defined. This MUST be non-zero if the User Type is Expiring User (0x07) and MUST be zero if the User Type is not Expiring User (0x07).
User Name Encoding	Controlling node defined.
User Name Length	Controlling node defined based on the length of the User Name.
User Name	User defined or controlling node defined among supported User Name Encoding characters and User Name Length.

CL:0083.01.33.06.1 A controlling node MAY offer the User Types under different names. If they are renamed they
CL:0083.01.32.07.1 SHOULD be accompanied by a description, and the name mapping MUST be noted in the product
CL:0083.01.31.08.1 documentation.

CL:0083.01.31.09.1 If a controlling node does not offer all User Types, it MUST make note of which User Types are not controlled in the product documentation.

CL:0083.01.31.10.1 If a controlling node does not offer all Credential Rules, it MUST make note of which Credential Rules are not controlled in the product documentation.

6.2.21.2.2 Delete a User

CL:0083.01.31.11.1 When the end user performs this action, the issued command MUST comply with Table 6.35.

Table 6.35: User Credential::Delete a User

Field	Value
Command	USER_SET (0x05)
Operation Type	Controlling node defined as Delete (0x02).
User Unique Identifier	User defined or controlling node defined among supported User Identifiers in the range of what was specified as supported in the *User Capabilities Report Command*.
User Type	Omitted or controlling node defined as the default value.
User Active State	Omitted or controlling node defined as the default value.
Credential Rule	Omitted or controlling node defined as the default value.
Expiring Timeout Minutes	Omitted or controlling node defined as the default value.
User Name Encoding	Omitted or controlling node defined as the default value.
User Name Length	Omitted or controlling node defined as the default value.
User Name	Omitted or controlling node defined as the default value.

CL:0083.01.33.12.1 A controlling node MAY allow the end user to erase all Users and their associated Credentials and
CL:0083.01.31.13.1 schedules at once. In this case, the User Unique Identifier field MUST be set to 0x0000.

6.2.21.2.3 Add/Modify a Credential

CL:0083.01.31.14.1 When the end user performs this action, the issued command MUST comply with Table 6.36.

Table 6.36: User Credential::Add/Modify a Credential

Field	Value
Command	CREDENTIAL_SET (0x0A)
User Unique Identifier	User defined or controlling node defined among supported User Identifiers in the range of what was specified as supported in the *User Capabilities Report Command*.
Credential Type	User defined or controlling node defined among supported Credential Types specified in the *Credential Capabilities Report Command*.
Credential Slot	User defined or controlling node defined in the range of one to the Number of Supported Credential Slots for the given Credential Type in the *Credential Capabilities Report Command*.
Operation Type	Controlling node defined as Add (0x00) or Modify (0x01).
Credential Length	Controlling node defined based on the length of the Credential Data. MUST not be less than the Min Length of Credential Data for the specific Credential Type specified in the *Credential Capabilities Report Command* and MUST not be more than the Max Length of Credential Data for the specific Credential Type specified in the *Credential Capabilities Report Command*.
Credential Data	User defined.

CL:0083.01.33.15.1 A controlling node MAY infer the Credential Type from the Credential Data provided by the end user.

CL:0083.01.31.16.1 If a controlling node does not offer all Credential Types, it MUST make note of which Credential Types are not controlled in the product documentation.

6.2.21.2.4 Delete a Credential

CL:0083.01.31.17.1 When the end user performs this action, the issued command MUST comply with Table 6.37.

Table 6.37: User Credential::Delete a Credential

Field	Value
Command	CREDENTIAL_SET (0x0A)
User Unique Identifier	User defined or controlling node defined among supported User Identifiers in the range of what was specified as supported in the *User Capabilities Report Command*.
Credential Type	User defined or controlling node defined among supported Credential Types specified in the *Credential Capabilities Report Command*.
Credential Slot	User defined or controlling node defined in the range of one to the Number of Supported Credential Slots for the given Credential Type in the *Credential Capabilities Report Command*.
Operation Type	Controlling node defined as Delete (0x02).
Credential Length	Zero or controlling node defined.
Credential Data	Omitted or User defined.

CL:0083.01.33.18.1 A controlling node MAY allow the end user to erase all Credentials of a Credential Type for a User
CL:0083.01.31.19.1 Unique Identifier at once. In this case, the User Unique Identifier and Credential Type MUST be non-zero and the Credential Slot MUST be 0x0000.

CL:0083.01.33.20.1 A controlling node MAY allow the end user to erase all Credentials of all Credential Types for a
CL:0083.01.31.21.1 User Unique Identifier at once. In this case, the User Unique Identifier MUST be non-zero and the Credential Type MUST be 0x00.

CL:0083.01.33.22.1 A controlling node MAY allow the end user to erase all Credentials for all Users at once. In this case,
CL:0083.01.31.23.1 the User Unique Identifier MUST be 0x0000.

CL:0083.01.33.24.1 A controlling node MAY allow the end user to erase all Credentials for a Credential Type at once. In
CL:0083.01.31.25.1 this case, the User Unique Identifier and Credential Slot MUST be 0x0000 and the Credential Type MUST be non-zero.

6.2.21.2.5 Set the Admin PIN Code

CL:0083.01.31.32.1 This action MUST be available to the end user if the supporting node supports the Admin PIN Code functionality. When the end user performs this action, the issued command MUST comply with Table 6.38.

Table 6.38: User Credential::Set the Admin PIN Code

Field	Value
Command	ADMIN_PIN_CODE_SET (0x1A)
Admin PIN Code Length	0 for Admin PIN Code deactivation, else in accordance with the Min..Max Credential Length of Credential Type PIN Code. It MUST be possible to de-activate the Admin PIN Code if the supporting node supports Admin Code Deactivation.
Admin PIN Code	User defined among supported ASCII characters (digits 0..9) with a length in the range 4..10 bytes. Zero length if deactivating the Admin PIN Code.

6.2.21.3 Node properties

CL:0083.01.41.26.1 A controlling node MUST have a UI allowing the end user to see the following properties:

- The list of Users and their basic properties:
  - The User Name
  - The User Active State
  - The Credentials assigned to a User and their basic properties:
    - The Credential Type

CL:0083.01.42.27.1 A controlling node SHOULD have a UI allowing the end user to see the following properties:

- The number of supported User Unique Identifiers
- The supported Credential Rules (Single, Dual, Triple)
- The supported Credential Types
- The list of Users and their additional properties:
  - The User Unique Identifier
  - The User Type
  - The Credential Rule
  - The Expiring Timeout Minutes, if the User Type is Expiring User (0x07)
  - The Credentials assigned to a User and their additional properties:
    - The Credential Slot
    - The Credential Data

CL:0083.01.41.28.1 A node controlling this command class MUST also control the Notification Command Class, version 8 or newer and have a UI allowing the end user to see Notification Reports with the following Notification Types.

- **Notification Type "Access Control" (0x06)**
  - "Credential lock/close operation" (0x23)
  - "Credential unlock/open operation" (0x24)
  - "Valid credential access denied due to User Active State being set to Occupied Disabled" (0x2F)
  - "Valid credential access denied due to the User's schedule being inactive" (0x30)
  - "User access denied due to not enough credentials entered for the User's Credential Rule" (0x31)
  - "Invalid credential used to access the node" (0x32)
  - "Non-Access credential entered via local interface" (0x33)
- **Notification Type "Emergency Alarm" (0x0A)**
  - "Panic Alert" (0x04)

6.2.21.4 Additional control requirements

CL:0083.01.51.29.1 If a node controlling this command class supports setting schedules for Unique User Identifiers, then they MUST also control the *Schedule Entry Lock Command Class, version 4 [NEVER CERTIFIED]*, or higher.

CL:0083.01.52.30.1 If the supporting node supports the *All Users Checksum Report Command*, the controlling node SHOULD verify the All Users Checksum periodically (e.g. once a day) to ensure that the Users and Credentials are synchronized between the supporting and controlling nodes.

CL:0083.01.52.31.1 A node controlling this command class SHOULD be resilient to receiving error messages in the User Report Type or Credential Report Type fields in *User Report Command* and *Credential Report Command* when sending a *User Set Command* or a *Credential Set Command*, respectively, and try to resend the command with corrected fields.

6.2.22 Window Covering Command Class, version 1

6.2.22.1 Mandatory interview

CL:006A.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.28.

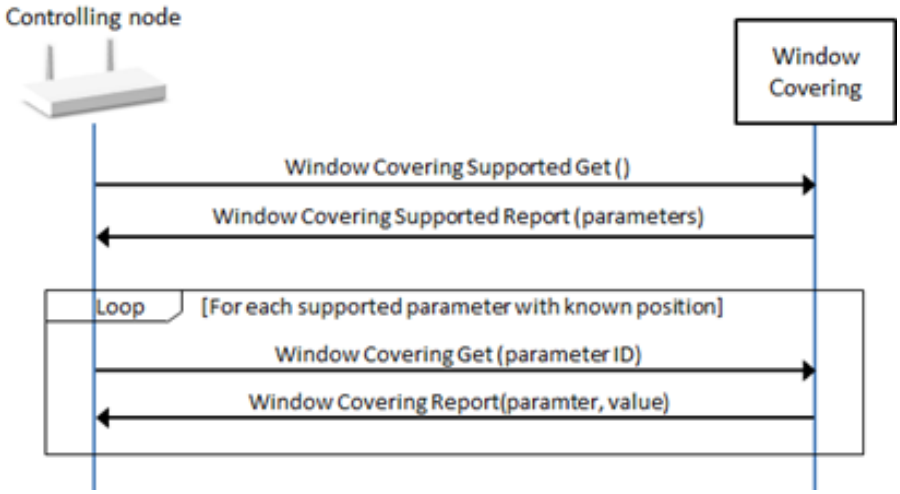


Figure 6.28: Window Covering Command Class interview

6.2.22.2 Minimum end user functionalities

CL:006A.01.31.02.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.2.22.2.1 Go to position

CL:006A.01.31.02.1 This action MUST be available to the end user for parameters ID with known positions (odd parameters IDs). When the end user performs this action, the issued command MUST comply with Table 6.39.

Table 6.39: Window Covering::Go to a Position

Field	Value
Command	WINDOW_COVERING_SET
Parameter count	User defined (0x01)
Parameter ID x	User defined among supported (odd values)
Value x	User defined among 0x00..0x63
Duration	User defined or 0xFF

6.2.22.2.2 Start level change up or down

CL:006A.01.31.03.1 This action MUST be available to the end user for all supported parameters ID. When the end user performs this action, the issued command MUST comply with Table 6.40.

Table 6.40: Window Covering::Start Level Change Up or Down

Field	Value
Command	WINDOW_COVERING_START_LEVEL_CHANGE
Up/Down	User defined among 0x00 and 0x01
Parameter ID	User defined among supported parameters ID
Duration	User defined or 0xFF

6.2.22.2.3 Stop level change

CL:006A.01.31.04.1 This action MUST be available to the end user for all supported parameters ID. When the end user performs this action, the issued command MUST comply with Table 6.41.

Table 6.41: Window Covering::Stop Level Change

Field	Value
Command	WINDOW_COVERING_STOP_LEVEL_CHANGE
Parameter ID	User defined among supported parameters ID

6.2.22.3 Node properties

CL:006A.01.42.01.1 Controller SHOULD have a UI allowing the end user to see the following properties:

- Last known position/value for all parameters IDs with known position

6.2.22.4 Additional control requirements

CL:006A.01.51.01.2 A controlling node MUST NOT interview and provide controlling functionalities for the Multilevel Switch Command Class for a node (or endpoint) supporting this Command Class, as it is a fully redundant and less precise application functionality.

6.3 Management Command Class Control Definitions

6.3.1 Association Command Class, version 1-4

6.3.1.1 Mandatory Node interview

CL:0085.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.29.

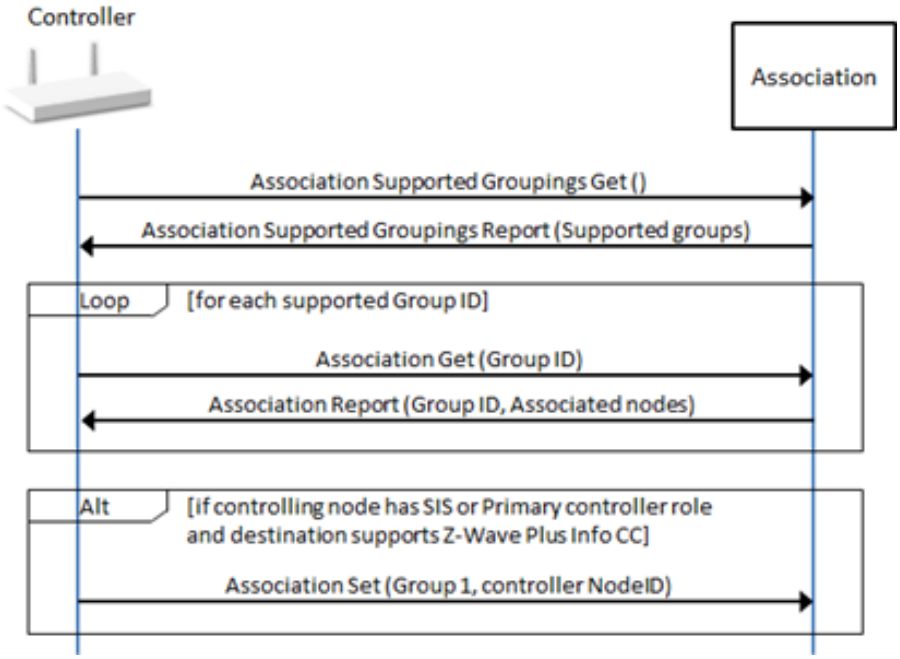


Figure 6.29: Association Command Class interview

6.3.1.2 Minimum End User functionalities

No minimum end user functionality is required for this Command Class.

CL:0085.01.31.01.1 A controlling node implementing a UI that allows an end user to establish association between nodes MUST NOT restrict the end user from establishing associations that are allowed. Refer to Section 6.3.1.4 for allowed associations.

6.3.1.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.3.1.4 Additional Control requirements

CL:0085.01.51.01.1 If the supporting node also supports Multi Channel Association and the controlling node controls Multi Channel Association, the controlling node MUST interview and control the Multi Channel Association Command Class instead of this Command Class.

CL:0085.01.51.02.1 A controlling node MUST use the Association Group Information (AGI) Command Class to probe the commands that a given association group will be sending before creating associations towards other nodes.

If an association group in a Node A sends one or more controlling commands:

CL:0085.01.51.03.3

- A controlling node MUST NOT associate Node A to a Node B destination that does not support the Command Class that the Node A will be controlling.

CL:0085.01.53.01.3 • A controlling node MAY create an association to a destination supporting an actuator Command Class if the actual association group sends Basic Control Command Class.

  For Association version 1 and version 2:

CL:0085.01.51.04.3 – A controlling node MUST NOT associate Node A to a Node B destination if Node A and Node B's highest Security Class are not identical.

  For Association version 3 or newer:

CL:0085.03.51.01.1 – A controlling node MUST NOT associate Node A to a Node B destination if Node A was not granted Node B's highest Security Class.

If an association group in Node A sends only supporting commands:

CL:0085.01.53.02.3 • A controlling node MAY create an association to any Node B destination if the actual association group sends commands reflecting the support of a Command Class by the sending Node/End Point. Refer to [25] for supporting/controlling commands.

CL:0085.01.51.06.1 • A controlling node MUST NOT associate Node A to a Node B destination if Node B was not granted Node A's highest Security Class.

6.3.1.4.1 Removing associations

CL:0085.01.51.05.1 A controlling node MUST NOT remove already associated nodes to a destination Group to associate themselves, unless:

- The destination NodeID/Endpoint has left the network (i.e. SIS has received a Device Reset Locally Notification)
- The destination of a group has changed capabilities and does not support the command received via the association group (also if a Multi Channel End Point is removed).
- An end user has actively confirmed to remove associations
- The lifeline group is full and the controlling node has the SIS Role.

6.3.2 Association Group Information (AGI) Command Class, version 1-3

6.3.2.1 Mandatory node interview

CL:0059.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.30.

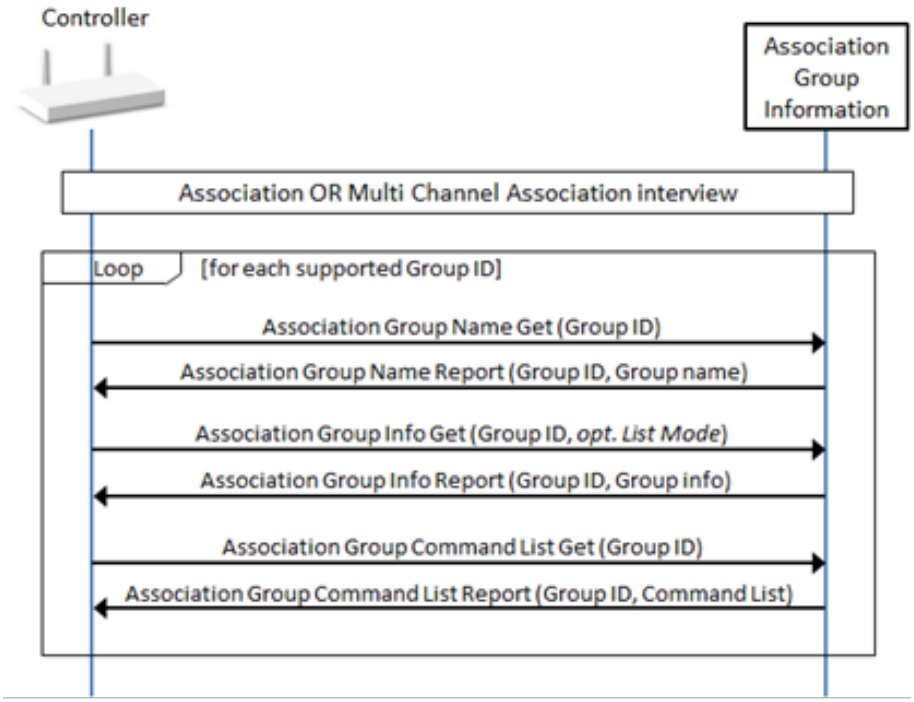


Figure 6.30: Association Group Information (AGI) Command Class interview

CL:0059.01.23.01.1 A controlling node MAY issue a single Association Group Info Get Command by using the List Mode flag.

6.3.2.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.3.2.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.3.3 Battery Command Class, version 1

6.3.3.1 Mandatory node interview

CL:0080.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.31

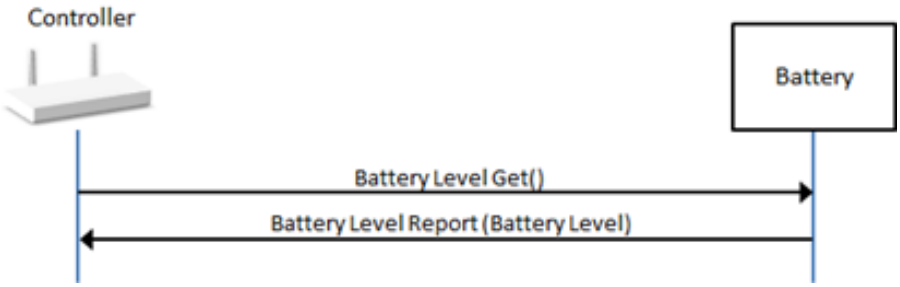


Figure 6.31: Battery Command Class interview

6.3.3.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.3.3.3 Node properties

CL:0080.01.41.01.1 A controlling node MUST allow the end user to see the last known battery level.

6.3.3.4 Additional control requirements

CL:0080.01.52.01.1 Unless unsolicited Battery Report Commands are received, a controlling node SHOULD Probe the current battery level at least every month

CL:0080.01.51.02.1 A controlling node MUST indicate to the end user that the battery needs to be replaced or reloaded when the supporting node issues a Battery Report with the *Battery Level* field set to 0xFF.

6.3.4 Device Reset Locally Command Class, version 1

6.3.4.1 Mandatory node interview

No mandatory node interview is required for this Command Class.

6.3.4.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.3.4.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.3.4.4 Additional control requirements

CL:005A.01.52.01.1 A controlling node receiving the Device Reset Locally Notification Command SHOULD consider the sending node to be a failing node and accordingly perform relevant maintenance operations like removing failing nodes, removing associations to failing nodes, etc.

CL:005A.01.51.01.1 A controlling node receiving the Device Reset Locally Notification Command MUST indicate to the end user that the node has been reset and left the Z-Wave network.

6.3.5 Firmware Update Meta Data Command Class, version 1-6

6.3.5.1 Mandatory node interview

CL:007A.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.32.

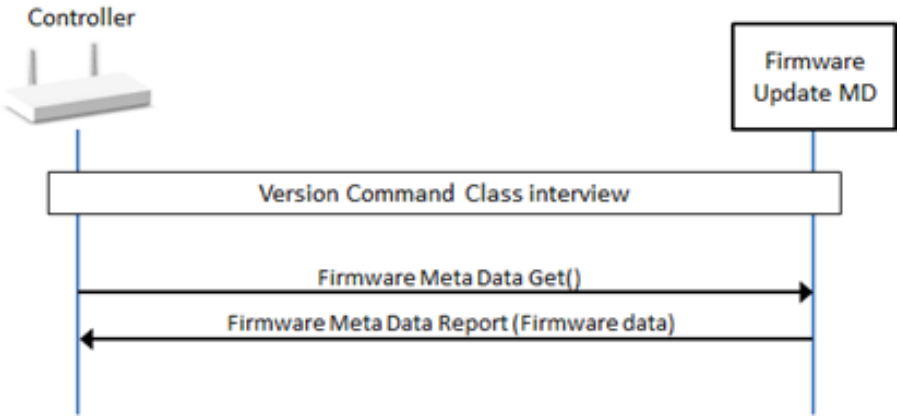


Figure 6.32: Firmware Update Meta Data Command Class interview

6.3.5.2 Minimum end user functionalities

6.3.5.2.1 Firmware update

CL:007A.01.32.01.1 A controlling node SHOULD provide a method for updating the firmware of a supporting node. The end user SHOULD be able to select a firmware file or ask the controller to look for updates automatically.

CL:007A.01.31.01.1 If this functionality is available, the issued commands MUST comply with Figure 6.33 when the end user performs this action.

CL:007A.01.33.01.1 A controlling node MAY use additional commands such as Firmware Update Activation Set Command for the firmware update.

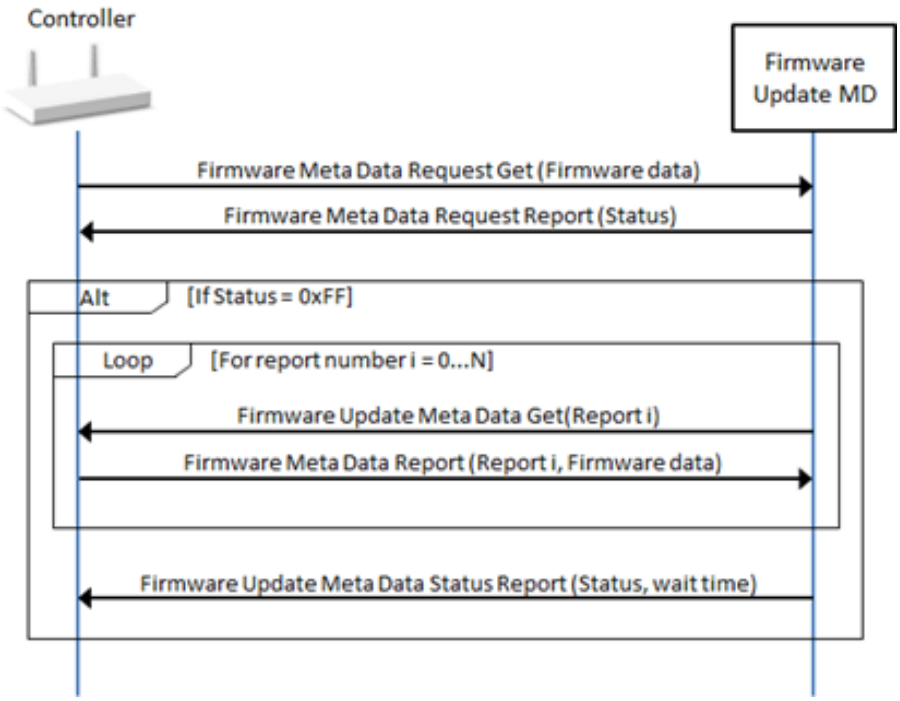


Figure 6.33: Firmware Update Meta Data::Firmware Update

CL:007A.01.32.02.1 If a controlling node allows the end user to interrupt an ongoing firmware update transfer, it SHOULD issue a FIRMWARE_UPDATE_MD_REPORT Command with the Last field set to 1 prematurely.

6.3.5.3 Node properties

No node property is required to be made available to the end user for this Command Class.

CL:007A.01.42.01.1 A controlling node SHOULD have a UI allowing the end user to see the Firmware version of a supporting node.

6.3.5.4 Additional control requirements

CL:007A.01.52.01.1 If the supporting node supports Battery Command Class and the controlling node controls Battery Command Class, the controlling node SHOULD issue a Battery Get and read the battery level before initiating a Firmware Update.

CL:007A.01.52.02.1 A controlling node SHOULD NOT initiate a Firmware Update if the supporting node Battery level is less than 50%.

CL:007A.01.51.01.1 A controlling node MUST perform a full interview of a node after performing a firmware update.

6.3.6 Indicator Command Class, version 1-3

6.3.6.1 Mandatory node interview

CL:0087.01.21.01.1 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.34.

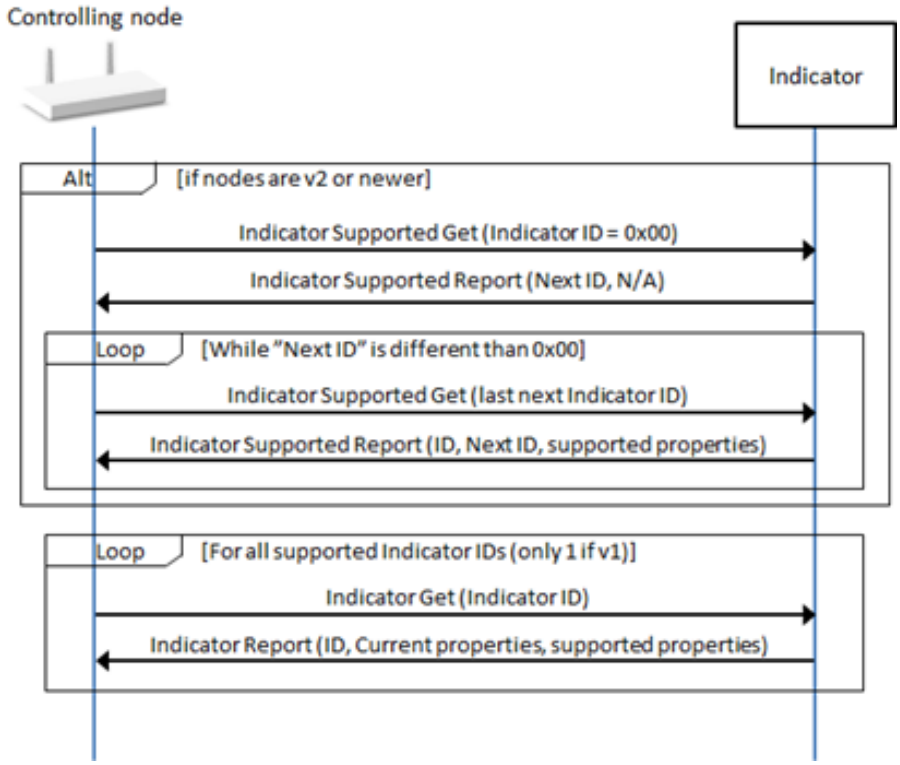


Figure 6.34: Indicator Command Class interview

6.3.6.2 Minimum end user functionalities

CL:0087.01.31.01.1 A node controlling this command class MUST allow the end user to perform the actions described below.

6.3.6.2.1 Identify

CL:0087.01.31.02.1 If the supporting and controlling nodes are version 3 or newer, the end user MUST be able to instruct the node to identify itself. When the end user performs this action, the issued command MUST comply with Table 6.42.

Table 6.42: Indicator::Identity

Field	Value
Command	INDICATOR_SET
Indicator 0 Value	0x00
Indicator object count	0x03
Indicator ID 1	0x50
Property ID 1	0x03
Value 1	0x08
Indicator ID 2	0x50
Property ID 2	0x04
Value 2	0x03
Indicator ID 3	0x50
Property ID 3	0x05
Value 3	0x06

The supporting node will be switched on 600ms and switched off 200ms three times.

CL:0087.01.33.01.1 A controlling node MAY hide this functionality away from the end user if the supporting node also supports the Wake Up Command Class.

6.3.6.3 Node properties

CL:0087.01.42.01.1 The controlling node SHOULD have a UI allowing the end user to see/access the following properties:

- Last known Indicators' state/value (ON/OFF or x%), if any

6.3.6.4 Additional control requirements

CL:0087.01.51.01.1 A node controlling this command class MUST NOT reuse the identify command for any other purpose than a node identification application.

CL:0087.01.52.01.3 A node controlling this Command Class SHOULD NOT make a supporting node blink 3 times for any indication (except when using the Identify Indicator for the Identify purpose).

CL:0087.01.52.02.1 A node controlling this Command Class SHOULD allow the end user to actuate additional indicating resources.

6.3.7 Multi Channel Association Command Class, version 2-5

6.3.7.1 Mandatory node interview

CL:008E.01.21.01.1 A node controlling this command class MUST perform a supporting node interview according to Figure 6.35.

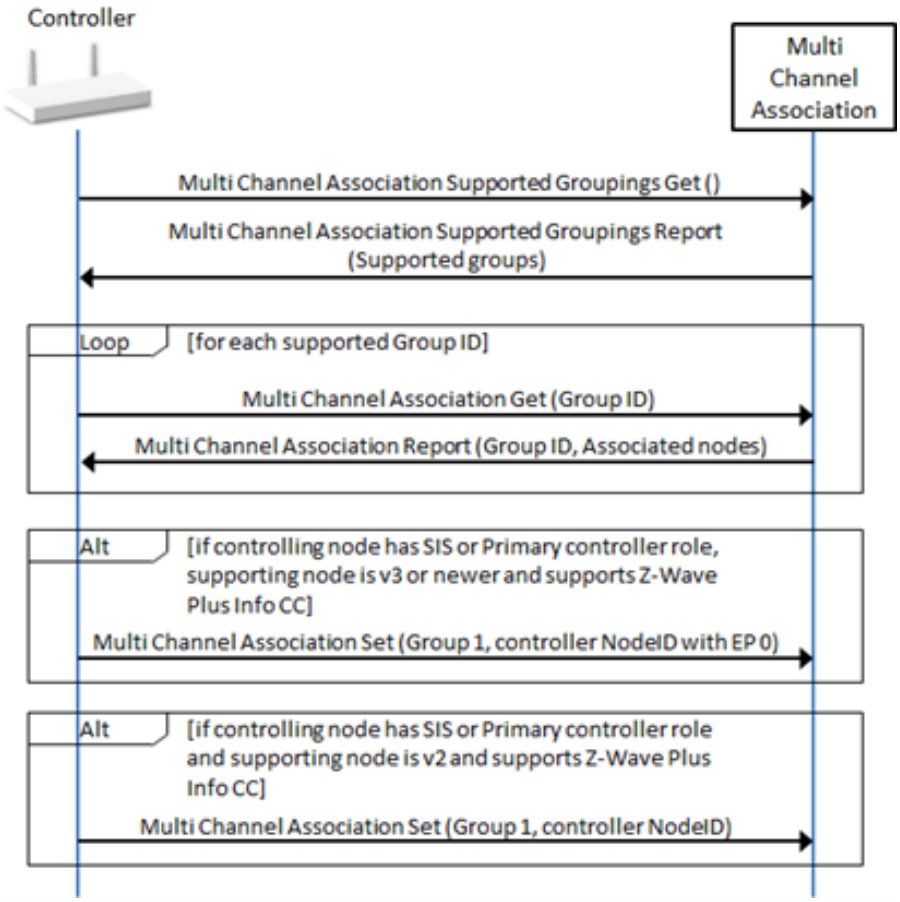


Figure 6.35: Multi Channel Association Command Class interview

CL:008E.01.21.02.2 The lifeline association MUST be an End Point Association (controller NodeID after the MULTI_CHANNEL_ASSOCIATION_SET_MARKER) if:

- Both nodes implement Multi Channel Association, version 3 or newer.
- The supporting node also supports the Multi Channel Command Class.

CL:008E.01.22.01.1 The lifeline association SHOULD be a NodeID association in any other case.

6.3.7.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

CL:008E.01.31.01.1 A controlling node implementing a UI which allows an end user to establish association between nodes MUST NOT restrict the end user from establishing associations that are allowed. Refer to Section 6.3.7.4 for allowed associations.

6.3.7.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.3.7.4 Additional control requirements

CL:008E.01.51.01.1 If the supporting node also supports Association and the controlling node controls Association, the controlling node MUST NOT interview and control the Association Command Class.

CL:008E.01.51.02.1 A controlling node MUST use the Association Group Information (AGI) Command Class to probe the commands that a given association group will be sending before creating associations towards other nodes.

If an association group in a Node A sends one or more controlling commands:

CL:008E.01.51.03.3 - A controlling node MUST NOT associate Node A to a Node B destination that does not support the Command Class that the Node A will be controlling.

CL:008E.01.53.01.3 - A controlling node MAY create an association to a destination supporting an actuator Command Class if the actual association group sends Basic Control Command Class.

  For Multi Channel Association version 2 and version 3:

CL:008E.02.51.01.1 – A controlling node MUST NOT associate Node A to a Node B destination if Node A and Node B's highest Security Class are not identical.

  For Multi Channel Association version 4 or newer:

CL:008E.04.51.01.1 – A controlling node MUST NOT associate Node A to a Node B destination if Node A was not granted Node B's highest Security Class.

If an association group sends only supporting commands:

CL:008E.01.53.02.2 - A controlling node MAY create an association to any destination if the actual association group sends commands reflecting the support of a Command Class by the sending Node/End Point. Refer to [25] for supporting/controlling commands.

CL:008E.01.51.06.1 - A controlling node MUST NOT associate Node A to a Node B destination if Node B was not granted Node A's highest Security Class.

6.3.7.4.1 Removing associations

CL:008E.01.51.05.1 A controlling node MUST NOT remove already associated nodes to a destination Group to associate themselves, unless:

- The destination NodeID/Endpoint has left the network (i.e. SIS has received a Device Reset Locally Notification)
- The destination has changed capabilities and does not support the command received via the association group (also if a Multi Channel End Point is removed).
- An end user has actively confirmed to remove associations
- The lifeline group is full and the controlling node has the SIS Role.

6.3.8 Version Command Class, version 1-3

6.3.8.1 Mandatory node interview

CL:0086.01.21.01.2 A node controlling this command class MUST perform a supporting node interview according to Figure 6.36.

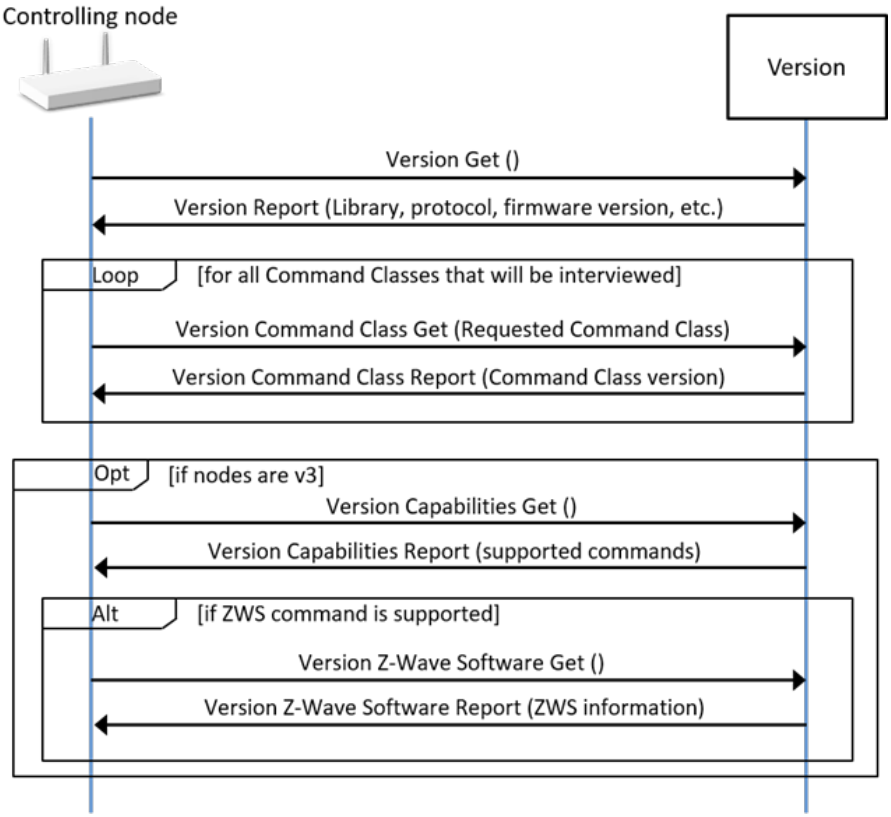


Figure 6.36: Version Command Class interview

CL:0086.01.21.02.1 The interview part starting from the Version Capabilities Get Command is optional. The Controlling node MUST interview the Version Capability Get Command before issuing the Version Z-Wave Software Get Command if the Controlling node has the intent of using the Z-Wave software version.

6.3.8.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.3.8.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.3.8.4 Additional control requirements

CL:0086.01.51.01.1 A controlling node interviewing a Multi Channel End Point MUST request the End Point's Command Class version from the Root Device if the End Point does not advertise support for the Version Command Class.

6.3.9 Wake-Up Command Class, version 1-2

6.3.9.1 Mandatory node interview

CL:0084.01.23.01.1 If the controlling node has the inclusion controller or secondary controller role in a network, it MAY
CL:0084.01.21.01.2 skip the node interview. In any case, it MUST NOT issue a Wake-Up Interval Set to a supporting node.

CL:0084.01.21.02.1 If the controlling node has the SIS or primary controller role in a network, it MUST perform a supporting node interview according to Figure 6.37.

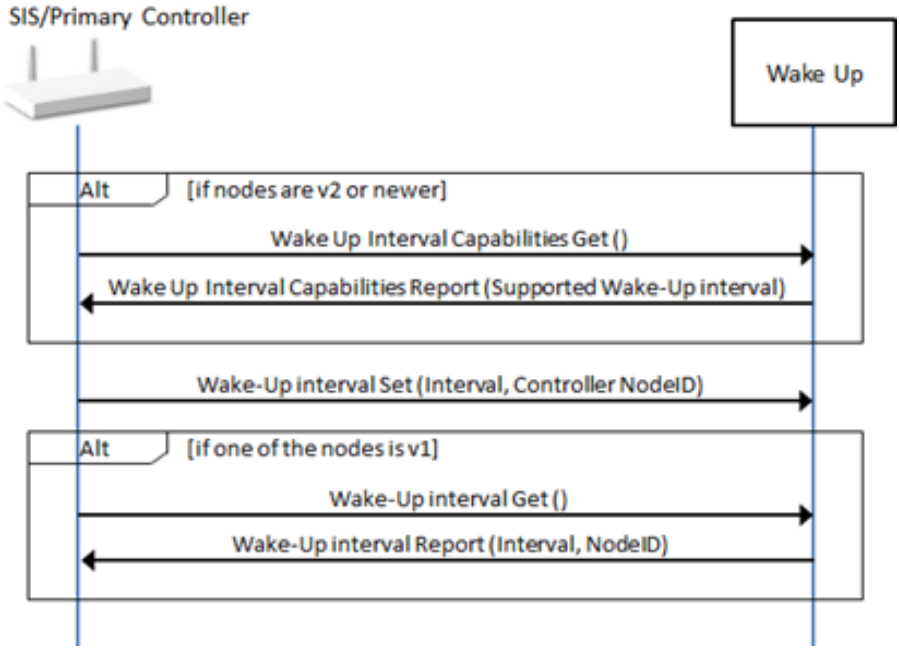


Figure 6.37: Wake Up Command Class interview

CL:0084.01.21.03.1 A controlling node MUST set a supported Wake Up Interval time value when commissioning a version 2 supporting node.

CL:0084.01.21.04.1 A controlling node MUST set its own NodeID as the Wake-Up destination.

6.3.9.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.3.9.3 Node properties

No node property is required to be made available to the end user for this Command Class.

CL:0084.01.42.01.1 A controlling node queuing commands for a Wake-Up node SHOULD indicate to the end user that the commands will be transmitted when the destination wakes up again.

CL:0084.01.43.01.1 A controlling node MAY show to the end user the expected time until the next Wake-Up.

6.3.9.4 Additional control requirements

CL:0084.01.52.01.1 A controlling node SHOULD verify that the Wake Up Interval Set Command executed successfully by either using Supervision encapsulation or reading back the Wake Up Interval settings.

CL:0084.01.52.02.1 If the Wake Up Interval Set Command was ignored by a version 1 supporting node, the controlling node SHOULD try again using the currently defined interval at the supporting node and its NodeID.

CL:0084.01.52.03.1 A controlling node SHOULD read the Wake Up Interval of a supporting node when the delays between Wake Up periods are larger than what was last set at the supporting node.

6.4 Transport Encapsulation Command Class Control Definitions

6.4.1 CRC-16 Encapsulation Command Class Control Definitions, version 1

6.4.1.1 Mandatory node interview

No node interview is required for this Command Class

6.4.1.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.4.1.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.4.1.4 Additional Control Requirements

CL:0056.01.52.01.1 A controlling node SHOULD use CRC-16 encapsulation to communicate with a supporting node when no security encapsulation is used and the communication speed is lower than 100 kbits/s.

CL:0056.01.51.01.1 A controlling node MUST support the CRC-16 Command Class and correctly handle received commands encapsulated with CRC-16.

CL:0056.01.51.02.1 A controlling node MUST use CRC-16 encapsulation to return a response to a command if the request was received using CRC-16 encapsulation (aka “answer-as-asked”).

6.4.2 Multi Channel Command Class, version 3-4

6.4.2.1 Mandatory node interview

CL:0060.01.21.01.3 A node controlling this Command Class MUST perform a supporting node interview according to Figure 6.38.

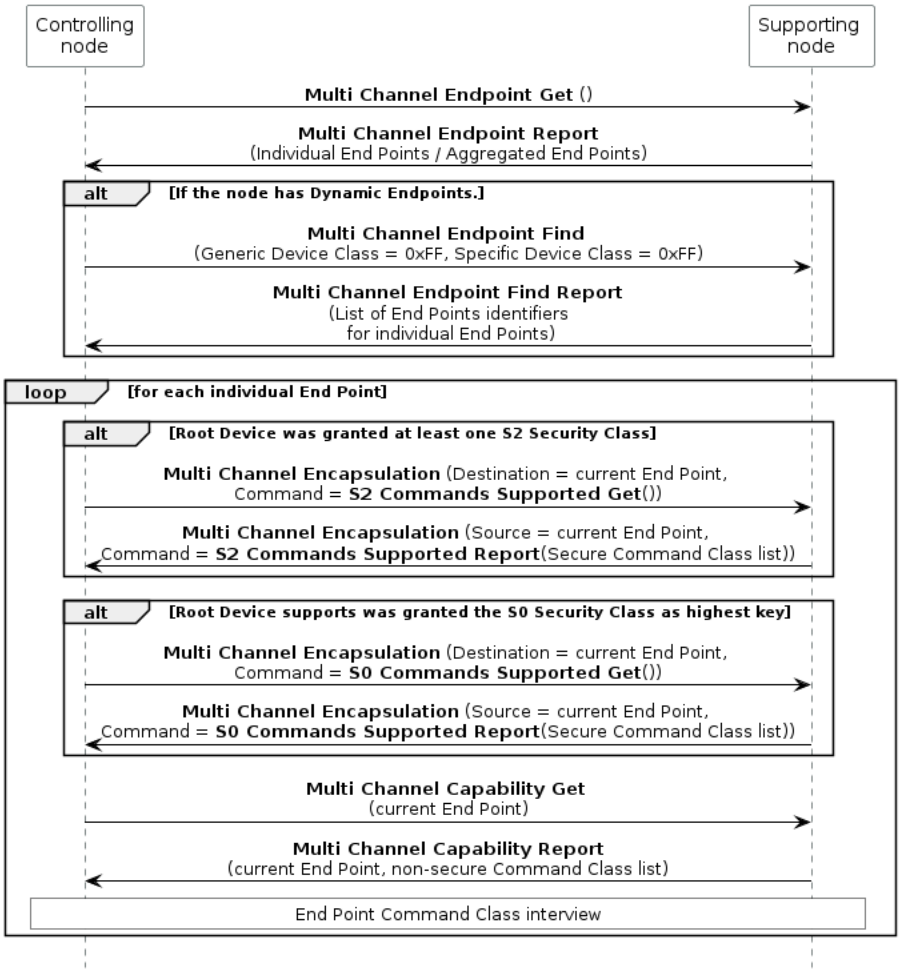


Figure 6.38: Multi Channel Command Class interview

CL:0060.01.23.01.1 A controlling node MAY skip the interview of aggregated End Points and MAY skip issuing a Multi Channel Capability Get for any of the aggregated endpoints.

CL:0060.01.23.02.1 A controlling node MAY skip sending a *Multi Channel End Point Find Report Command* if there are no dynamic endpoints. (*Dynamic* field set to 0 in the *Multi Channel End Point Report Command*).

6.4.2.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.4.2.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.4.2.4 Additional Control Requirements

CL:0060.01.51.01.2 A node controlling this Command Class MUST provide control of all its controlled command classes on every individual End Point.

6.4.2.4.1 Root Device and End Point Command Classes

When End Point functionality is advertised in the Root Device NIF, service discovery mechanisms like mDNS and installer-style GUIs risk presenting a Root Device functionality which is actually a mirror representation of an End Point functionality.

CL:0060.01.52.01.1 Therefore, application command classes of the Root Device capabilities that are also advertised by at least one End Point SHOULD be filtered out by controlling nodes before presenting the functionalities via service discovery mechanisms like mDNS or to users in a GUI.

6.4.2.4.2 S0 only Multi Channel nodes

The following considerations apply for nodes supporting S0 and Multi Channel Command Class but do not support S2 Command Class.

Legacy S0 only nodes may implement some secure and some non-secure End Points. Such a node supporting S0 must advertise S0 in the Multi Channel Capability Report Command for a given End point if the End Point can be addressed with S0 encapsulation.

CL:0060.01.52.02.1 A controlling node SHOULD use S0 encapsulation with all End Points if the Root Device was bootstrapped with the S0 Command Class.

CL:0060.01.53.01.1 A controlling node MAY interview and control Command Classes present in the Multi Channel Capability Report of an End Point non-securely if the End Point does not respond to S0 encapsulated traffic.

6.4.2.4.3 Association and Multi Channel Association mapping

The following considerations apply for nodes supporting Association and/or Multi Channel Association and Multi Channel Command Class.

The Association groups functionality may be fully or partially mirrored between the Root Device and End Points. For example, an Association Remove Command issued to the Root Device may clear the association destination at the End Point groups.

CL:0060.01.52.03.1 A controlling node SHOULD read back the destinations in every group, including End Points groups after configuring associations.

6.4.3 Security 0 Command Class

6.4.3.1 Mandatory node interview

CL:0098.01.21.01.1 A node controlling this Command Class MUST observe the *Role Type Specification* requirements when including an S0 node.

6.4.3.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.4.3.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.4.3.4 Additional Control Requirements

A node is controlling S0 when it can perform S0 bootstrapping of other nodes. The S0 bootstrapping
CL:0098.01.51.01.1 process is described in *Security 0 (S0) Command Class, version 1* and a controlling node MUST observe these requirements.

CL:0098.01.51.02.1 A node controlling this Command Class MUST provide control of all its controlled command classes using S0 encapsulation.

6.4.4 Security 2 Command Class

6.4.4.1 Mandatory node interview

CL:009F.01.21.01.1 A node controlling this command class MUST observe the *Role Type Specification* requirements when including an S2 node.

6.4.4.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.4.4.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.4.4.4 Additional Control Requirements

S2 Security is controlled when a node can perform S2 bootstrapping of other nodes. The Bootstrapping
CL:009F.01.51.01.1 process is described in *Security 2 (S2) Command Class, version 1* and a controlling node MUST observe these requirements.

CL:009F.01.51.02.1 A node controlling this Command Class MUST provide control of all its controlled command classes using S2 encapsulation and at all its granted S2 Security Classes.

6.4.5 Supervision Command Class, version 1

6.4.5.1 Mandatory node interview

There is no mandatory node interview for a node controlling this Command Class.

6.4.5.2 Minimum end user functionalities

No minimum end user functionality is required for this Command Class.

6.4.5.3 Node properties

No node property is required to be made available to the end user for this Command Class.

6.4.5.4 Additional Control Requirements

CL:006C.01.51.01.1 A node issuing Supervision Get Commands MUST declare the Supervision Command Class as controlled during certification.

CL:006C.01.51.02.1 Any node issuing Supervision Get Commands MUST comply with the following:

- The Session ID field MUST be incremented each time a new unique Supervision Get Command is issued.
- A sending node MAY use the same Session ID for a multicast and singlecast follow-up carrying the same encapsulated command. A sending node MAY also use the same Session ID for all destinations of singlecast follow-up commands.

CL:006C.01.52.01.1 It has been found that some nodes issue Wake Up Notifications using the same Supervision SessionID every time. A controlling node SHOULD accept Wake Up Notifications even if they use the same SessionID.

7 Device Type v2 Specification

7.1 Introduction

This document describes the requirement associated to the Z-Wave Plus v2 compliant Device Types. It contains a list of requirements applying for all Z-Wave Plus v2 compliant nodes as well as requirements specifics to each defined Device Type.

7.2 Common Z-Wave Plus v2 Device Type Requirements

7.2.1 How to detect Z-Wave Plus v2 compliant nodes

DT:00.11.0001.1 A Z-Wave Plus v2 node MUST advertise version 0x02 in the Z-Wave Plus Version field of the Z-Wave Plus Info Report. Multi Channel End Points MUST advertise the same version number in their Z-Wave Plus Info Report

DT:00.11.0002.1 A Z-Wave Plus v2 node MUST set the Optional Functionality bit to 1 in its NIF. For the NIF's description, refer to Section 2.

7.2.2 Command Classes Support Requirements

DT:00.11.0003.1 A Z-Wave Plus v2 node MUST support the command classes listed in the following sections. When a version number is indicated, the node MUST support or control the indicated version or a newer one.

7.2.2.1 Root Device level

DT:00.11.0004.1 All Root Devices or nodes MUST support:

- Association, version 2
- Association Group Information
- Device Reset Locally
- Firmware Update Meta Data, version 5
- Indicator, version 3
- Manufacturer Specific
- Multi Channel Association, version 3
- Powerlevel
- Security 2
- Supervision
- Transport Service, version 2
- Version, version 2
- Z-Wave Plus Info, version 2

7.2.2.2 End Point level

DT:00.11.0005.1 All Multi Channel End Points MUST support:

- Association, version 2
- Association Group Information
- Multi Channel Association, version 3
- Supervision
- Z-Wave Plus Info, version 2

7.2.3 Identify

DT:00.11.0006.2 A Z-Wave Plus v2 node MUST support an Identify indicator (Indicator ID 0x50) which can be used for an Identify function.

DT:00.12.0004.1 The node is RECOMMENDED to use a visible LED for an identify function if it has an LED. If the node is itself a light source, e.g. a light bulb, this MAY be used in place of a dedicated LED.

DT:00.11.0007.1 The Root Device of a node MUST support the Indicator Command Class, version 3 or newer and support the Indicator ID 0x50 (Identify) and Properties ID 0x03, 0x04 and 0x05.

DT:00.13.0001.1 Multi Channel devices MAY support an Identify indicator on End Points, if the corresponding End Point has its own LED or light source.

DT:00.11.0008.1 If a Multi Channel device only implements a single indicator for the entire device, the End Points MUST NOT support the Identify Indicator, while the Root Device MUST support the Identify Indicator.

7.2.4 Dynamic Capabilities and Node Discovery

DT:00.11.0009.1 A controller MUST have a menu or method for an (advanced) end user to request the controller to perform a capability discovery for a given node (i.e. to perform a complete commissioning interview)

DT:00.13.0002.1 Nodes (and their End Points) MAY change capabilities based on a user action, such as changing configuration parameters or the physical addition/removal of a module.

DT:00.11.000A.1 End Point changing capabilities based on a user action MUST NOT be advertised as Dynamic End Points.

DT:00.13.0003.1 If supporting Configuration Command Class, nodes MAY issue a Configuration Report advertising a dynamic capabilities parameter value change in order to let the lifeline destination(s) know that some capabilities have changed.

DT:00.11.000B.1 However, a controlling node MUST NOT perform the node interview unless instructed to do so by the end user.

DT:00.11.000C.1 The configuration of Command Classes that are available before and after a capability change MUST remain unchanged. For instance, the Lifeline Association Group destination and Wake Up destination MUST stay identical when a node changes capabilities.

DT:00.11.000D.1 A node MUST stay compliant and observe Z-Wave Plus v2 Device Type requirements when and after changing capabilities. A node MAY change its Device Type when altering its capabilities.

DT:00.11.000E.1 A node being able to change between a secure only Device Type (S2 Access Control) and a regular S2 Device Type MUST observe the S2 Access Control Device Type requirements (3.6.8.2.1) even if configured to be a regular Device Type.

7.2.5 Controller Functionalities

A controller plays an important role in a Z-Wave network because this device hosts important functionality to create, maintain and configure the network and the home automation application. The following sections describe important rules to ensure that a controller is capable of fulfilling this important role.

7.2.5.1 Interoperability

DT:00.11.0011.1 To ensure interoperability, a controller MUST comply with the following requirements:

1. It is not acceptable to block interoperability by any means.
2. It is not acceptable to prevent inclusion of certified devices into a system or force exclusion of non- preferred devices after inclusion.
3. Devices from non-preferred manufacturers MAY be placed in a special section of the user interface; this section should be referred to as "Additional Z-Wave Ecosystem Devices". Additionally, it is acceptable to inform the user, upon inclusion of non-preferred devices that the device being included is not part of the vendors preferred ecosystem, and that control and support of the device by the vendor may be limited.

   a. It is not permitted to display additional pop-ups, ask for pin codes or implement any other blocking or discouraging behavior for inclusion or control of non-preferred devices. b. The Z-Wave Alliance recommends wording as follows. "You are about to include a Z-Wave compatible device that is not promoted by 'service provider name' for use in this application. While the device will work as expected the device may or may not support all of the features of the 'service provider name' recommended device."

7.2.5.2 Minimal Control Functionality

DT:00.11.0012.1 If a controller product supports short range wireless non-Z-Wave technology smarthome products (e.g. light bulbs, thermostats, door locks and the like) and Z-Wave technology products, it MUST, at a minimum, control the following Command Classes:

- Door Lock Command Class
- Binary Switch Command Class
- Multilevel Switch Command Class
- Thermostat Mode Command Class
- Thermostat Setpoint Command Class

It is acceptable to provide additional controlling functionalities for nodes from a preferred manufacturer as long as the controller provides the minimal required control functionalities for all nodes. Refer to Section 6 for Command Class control requirements.

7.2.6 Command Class Support Specific Requirements

Certain rules must be fulfilled depending on which command classes are supported. The following subsections detail the requirements of special command classes. Details about individual Command Classes can be found in Section 2, Section 3, Section 4 and Section 5.

7.2.6.1 Anti-theft Command Class

DT:00.11.002A.1 If the Anti-Theft Command Class is supported, it MUST be version 3 or newer.

7.2.6.2 Application Status Command Class

DT:00.11.0015.1 If a node is temporarily not capable to service a Get or Set Command request, it MUST support the Application Status Command Class and return an Application Busy Report Command to the initiator of the Get or Set.

DT:00.13.0004.1 If a node is always capable of servicing the Get and Set requests, it is OPTIONAL to support the Application Status Command Class.

7.2.6.3 Association requirements

7.2.6.3.1 Mandatory groups

DT:00.11.0016.1 The Root Device and End Points of a Z-Wave Plus v2 node MUST advertise the Association Groups indicated in Table 7.1 as a minimum.

Table 7.1: Z-Wave Plus v2 minimum required AGI table

Group identifier	Profile (2 bytes)	Command Class & Command list (N bytes)	Group name (UTF-8) (M bytes)
1	General:Lifeline	Refer to Section 7.2.6.3.2	Lifeline

DT:00.11.0017.1 A Node or Root Device MUST advertise a “Max Nodes Supported” value of 1 or more for the Lifeline Association group in the Association Report and Multi Channel Association Report Commands.

DT:00.12.0006.1 It is RECOMMENDED to support 5 lifeline destinations.

DT:00.11.0018.1 End Points MUST advertise a “Max Nodes Supported” of 0 for the Lifeline Group and MUST report their Lifeline Commands via the Root Device’s Lifeline Group when an End Point Association is established for the Lifeline Group. (Refer to Multi Channel Association Command Class)

7.2.6.3.2 Lifeline reports

DT:00.11.0019.1 A Z-Wave Plus v2 node MUST issue all the commands defined in [23] via the Lifeline Association Group to reflect its state changes if the corresponding command is supported by the node.

DT:00.12.0001.1 Report or Notification Commands SHOULD NOT be issued while performing a transitions from a Command Class state to another, but only when the supporting node has reached a final state. Intermediate transition state values SHOULD be advertised only if a long transition takes place (e.g. transition longer than 1 minute)

DT:00.12.0002.1 Any other Command Class state or configuration relevant to the control of the node or relevant for GUI information SHOULD be reported via the Lifeline when changed.

DT:00.11.001A.1 If the state change was triggered by other means than a Z-Wave Command, a node MUST issue the corresponding Report/Notification Command immediately to the lifeline destination(s).

If the state change was triggered by a Z-Wave Command:

DT:00.12.0003.1 • A node SHOULD NOT issue any Report/Notification Commands via the Lifeline if the actual lifeline destination issued the Set Command.

DT:00.11.001B.1 • A node MUST NOT issue any Report/Notification Command after a Command received via Multicast/broadcast addressing.

DT:00.11.002D.1 • A node MUST issue a Report/Notification Command after a Command received via Singlecast using Multi Channel multi-endpoint bit addressing.

DT:00.11.001C.1 • Unless the lifeline destination issued the command, a node MUST issue a Report/Notification Command after a command was received using singlecast addressing (including Multi Channel multi-End Point destination) via the Lifeline.

DT:00.11.001D.1 • If a node has more than one lifeline destination, it MUST issue a Report/Notification Command after a command was received using singlecast addressing (including Multi Channel multi-End Point destination) via the Lifeline.

An example of the expected frame flow is shown in Figure 7.1.

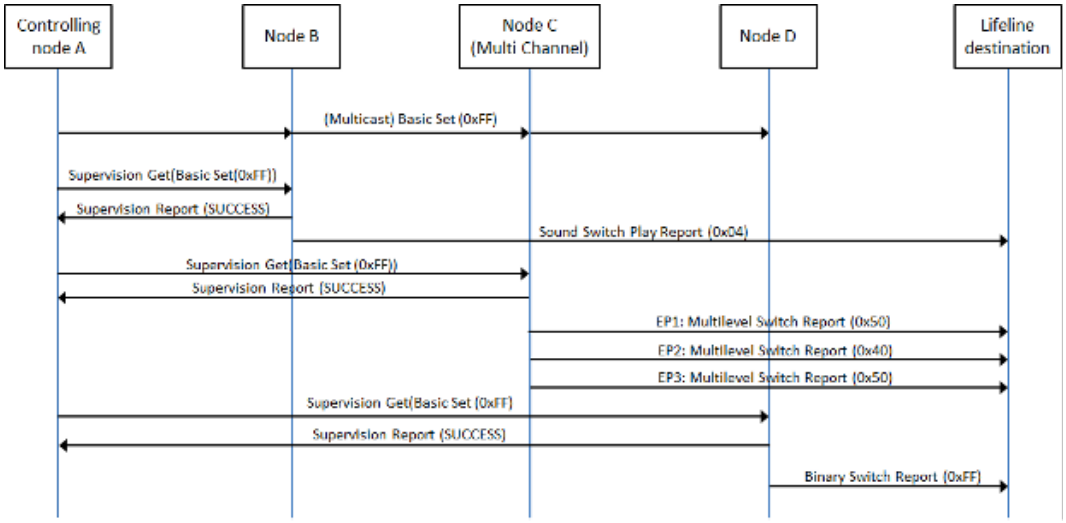


Figure 7.1: Lifeline status reports frame flow after multicast (example)

7.2.6.4 Configuration Command Class

DT:00.11.001E.1 If the Configuration Command Class is implemented, it MUST NOT replace any existing Command Class functionality.

DT:00.11.001F.1 If the Configuration Command Class is supported, it MUST be version 4 as a minimum.

7.2.6.5 Firmware Update Meta Data Command Class

DT:00.11.0020.1 The configuration of Command Classes that are available before and after a firmware update MUST remain unchanged. For instance, the Lifeline Association Group destination, Wake Up destination and any command class setting MUST stay identical before and after performing a firmware update.

7.2.6.5.1 SmartStart QR Code accuracy

DT:00.11.0032.1 A node undergoing a firmware update MAY have inaccurate data in its QR Code TLVs. An OTA Firmware Update MUST NOT lead to a loss of functionality that is indicated as supported in the QR Code.

DT:00.13.0005.1 The following TLVs MAY be inaccurate after a firmware update:

- The Product Type
- The ProductID

DT:00.11.0033.1 The following information MUST remain accurate after a firmware update:

- The DSK
- The requested keys (for non-SmartStart S2 inclusions)
- UUID16 TLV

DT:00.12.0005.1 The Supported Protocols TLV MAY become inaccurate. The Supported Protocol SHOULD only be inaccurate if more Protocols are supported than what the QR Code's Supported Protocols TLV indicates

7.2.6.6 Wake Up Command Class

DT:00.11.0021.1 If the node supports the Wake Up Command Class, the node MUST support manual Wake Up triggered by a user activation.

7.2.6.7 Multi Channel support

DT:00.11.0022.1 Actuator functionalities MUST reside in individual Multi Channel End Points according to the list of actuator Device Types.

DT:00.11.0023.1 Multi Channel devices MUST support the Multi Channel Command Class, version 4 or newer

DT:00.11.0024.1 A node supporting the Multi Channel Command Class MUST issue commands to the Lifeline destination from all of its End Points if an End Point Association has been established on the Root Device Lifeline association group.

DT:00.11.0025.1 A command issued to the Lifeline destination from a Multi Channel End Point MUST be Multi Channel encapsulated if an End Point Association has been established.

7.2.6.8 Security 2 Command Class

7.2.6.8.1 S2 bootstrapping and functionalities

DT:00.21.0001.1 After network inclusion, a node MUST consider S2 Bootstrapping as started after receiving the S2 KEX Get Command.

DT:00.21.0002.1 If a node times out waiting for security bootstrapping after network inclusion, it MUST NOT consider that bootstrapping failed and MUST consider that it was included non-securely.

DT:00.23.0001.1 If S2 bootstrapping started and did not complete successfully, a supporting node MAY remove support of its implemented command classes until re-included. Refer to Section 2 regarding NIF contents depending on network inclusion and security bootstrapping.

DT:00.21.0003.1 A node supporting S2 MUST consider any Security Class lower than its highest granted Security Class as unsecure communication. Certain command classes, such as Transport Service or Z-Wave Plus Info, must always be supported non-securely and present in the NIF if they are supported by a node. In

this case, non-secure support requirements are specified in each individual command class definition. A list of these special Command Classes is also available in [25] under "Additional Comments".

DT:00.21.0004.1 By default, a node supporting S2 MUST support its Command Classes only at the highest granted Security Class. If no Security Class was granted, the node MUST support all its Command Class using non-secure communication. This does not apply to S2 Access Control nodes, which MAY (or sometimes MUST) remove support for some Command Classes if a particular Security Class has not been granted.

7.2.6.8.2 S2 Security Classes requirements

Security 2 (S2) Command Class, version 1 defines several Security Classes.

DT:00.21.0005.1 A Z-Wave Plus v2 node MUST request either S2 Access Control or S2 Authenticated as its highest key.

DT:00.21.0006.1 An S2 supporting node MUST comply with the requirements indicated in the subsection below (Section 7.2.6.8.3 or Section 7.2.6.8.4) associated to its highest requested Security Class during S2 bootstrapping.

DT:00.21.0007.1 Nodes requesting the S0 Security Class MUST also comply with requirement indicated in 3.6.8.3.

DT:00.21.0008.1 A Multi Channel Root Device and all its End Points MUST share the same highest S2 Security Class.

7.2.6.8.3 S2 Access Control Security Class

The S2 Access Control Class is the most trusted class and is intended for home access control devices such as door locks, garage door openers or central controllers.

DT:00.21.0009.1 A node requesting the S2 Access Control Security Class MUST carry a representation of its DSK on itself and/or make it visible on its UI at any time when Learn Mode is enabled. Refer to Section 7.2.6.8.6 for DSK format and representation.

DT:00.21.000A.1 A node based on a controlling Device Type (4.3) requesting S2 Access Control Security Class MUST request S2 Authenticated and S2 Unauthenticated Security Classes when being S2 bootstrapped.

DT:00.23.0002.1 A node based on a controlling Device Type (4.3) requesting the S2 Access Control Security Class MAY decide to not support a set of its implemented Command Classes if it has not been granted a certain Security Class during S2 bootstrapping.

DT:00.23.0003.1 A node based on a supporting Device Type (4.2) requesting S2 Access Control Security Class MAY request any other Security Class for control purposes.

DT:00.21.000B.1 A node based on a supporting Device Type (4.2) MUST support its Command Classes depending on Security bootstrapping as follows:

DT:00.21.000C.1
- If security bootstrapped, it MUST support its Command Classes only if the highest granted key is S0 or S2 Access Control Security Class. It MUST NOT support its Command Classes at all if its highest granted Security Class is any other class than S0 or S2 Access Control.

DT:00.21.000D.1
- If it timed out waiting for security bootstrapping or S0/S2 bootstrapping failed, it MUST NOT support its Command Classes non-securely.

DT:00.21.000E.1
- The above two requirements MUST NOT apply for Command Classes that MUST always be in the NIF (refer to Section 7.2.6.8.1 and [25]).

7.2.6.8.4 S2 Authenticated Security Class

The S2 Authenticated Class is the 2nd most trusted class and is intended for secure applications in home control deployments.

DT:00.21.000F.1 A node requesting the S2 Authenticated Security Class MUST carry a representation of its DSK on itself and/or make it visible on its UI at any time when Learn Mode is enabled. Refer to Section 7.2.6.8.6 for DSK format and representation.

7.2.6.8.5 S0 Security Class requirements

The S0 Class is used for backwards compatibility with S0 supporting nodes.

DT:00.21.0011.1 An S2 node MUST NOT request the S0 Security Class if it does not support the Security 0 Command Class. An S2 node MUST NOT request the S0 Security Class without requesting an S2 Security Class.

DT:00.22.0001.1 Nodes with controlling capabilities and controllers SHOULD request the S0 Security Class for application control purposes.

7.2.6.8.6 DSK format and representations

The S2 Command Class defines a Device Specific Key (DSK) that enables authentication as part of the S2 Bootstrapping process.

The DSK can be represented with the following pre-defined formats: PIN code, DSK string and QR code.

DT:00.21.0012.1 The PIN code MUST be 5 decimal digits representing the value of the first 2 bytes of the node's DSK and MUST be constructed according to Figure 7.2.

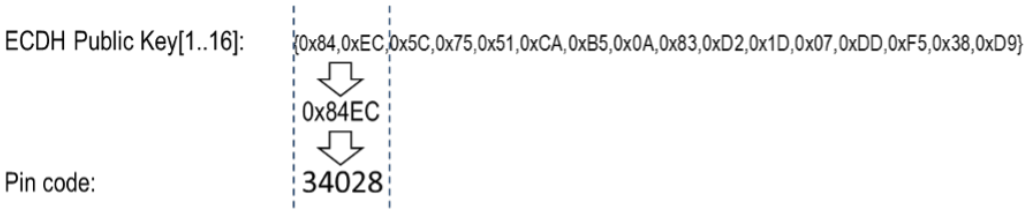


Figure 7.2: PIN code format

DT:00.21.0013.1 The DSK string MUST be 8 groups of 5 decimal digits, each representing 2 bytes of the DSK, separated with hyphens and MUST be constructed according to Figure 7.3.

Figure 7.3: DSK String format

DT:00.21.0014.1 The first five digits of the DSK string MUST be underlined to help the user identify the PIN code portion of the DSK string.

DT:00.21.0015.2 The QR code format MUST comply with the "Gen2" format defined in [29]. The QR code MUST include two TLV blocks: Type 0 (Product Type) and TLV Type 1 (Product ID) as defined by [28].

Additional formatting requirements for the market certification are given by the Z-Wave Alliance, refer to [30].

7.2.6.8.7 Mandatory DSK representations

DT:00.21.0016.1 If a node supports Learn Mode, the DSK representations present on a product MUST comply with Table 7.2 and the subsections below.

Table 7.2: Mandatory DSK representations for Z-Wave Plus v2 nodes

	On the Product	On the leaflet	On the box/package
PIN Code	At least one Required.	OPTIONAL	OPTIONAL
DSK String			
	Required in at least one place		
QR Code	Required if the node supports to be included using SmartStart inclusion	OPTIONAL	OPTIONAL

7.2.6.8.8 DSK on the product or UI

DT:00.21.0017.1 A product MUST have a QR code printed on the its outside or on its UI if it supports to be included using SmartStart inclusion.

DT:00.21.0018.1 A product MUST carry or display the DSK string or PIN code.

DT:00.21.0019.1 If the product carries the PIN code representation of the DSK, the product leaflet, documentation or packaging MUST contain the DSK string.

7.2.6.8.9 DSK on documentation or leaflet

DT:00.22.0002.1 It is RECOMMENDED that a leaflet inside the product's box, advertises the QR code and the full DSK string.

7.2.6.8.10 DSK on the product's box/package

DT:00.22.0003.1 The product packaging SHOULD carry the QR code and DSK string on its outside.

7.2.6.8.11 Filtering Security Class for controlling nodes

DT:00.23.0004.1 For Command Classes always supported non-securely (always in the NIF), a controlling node MAY accept a command at any security level shared with a sending S2 node.

DT:00.21.001A.1 For Command Classes supported securely, a controlling node MUST discard the command from a supporting node if not received at the highest common security level between the controlling node and the sending S2 node.

DT:00.21.001B.1 A controlling node MUST NOT discard a command if the sending node does not support the S2 Command Class.

A node is considered controlling or supporting based on which command it sends or receives. Each command node's role is marked in [25].

DT:00.22.0006.1 Further, a controller node that has performed S2 bootstrapping to a Node A SHOULD discard controlling commands from Node A if not received using the highest common security level for Command Classes that are always supported non-securely (always in the NIF).

7.2.6.8.12 Controlling nodes: Security Class learning

Any Z-Wave Plus v2 node controlling Command Classes (not using association groups) MUST discover the destination capabilities using every Security Class. If it intends to create associations between 2 other nodes, it MUST also discover which keys have been granted to both nodes.

If the controller is the SIS, it SHOULD skip the discovery as it knows which keys have been granted. If the controller is not the SIS or does not know which security levels to use with a destination, the following discovery algorithm is RECOMMENDED:

1. Request the NIF and read its contents, look for S2/Supervision and S0
2. If S2 is supported, for every S2 key starting from the highest:
    a. Issue the S2 Commands Supported Get
        i. If receiving no answer (or S2 Nonce Reports), conclude that the corresponding Security Class has not been granted to the node
        ii. If receiving an S2 Commands Supported Report with an empty list, conclude that the corresponding Security Class has been granted and is not the highest
        iii. If received an S2 Commands Supported Report with a non-empty list, conclude that the corresponding Security Class has been granted and is the highest.
3. If S0 is supported, discover if S0 key was granted:
    a. Issue a S0 encrypted S0 Security Command Supported Get command
    b. If S2 was in the NIF :
        i. If not receiving any response, conclude that the S0 Security Class has not been granted to the node
        ii. If receiving an S0 Commands Supported Report with an empty list, conclude that the S0 Security Class has been granted and is not the highest
        iii. If receiving an S0 Commands Supported Report with a non-empty list, conclude that the S0 Security Class has been granted and is the highest
    c. If S2 was not in the NIF:
        i. If receiving an S0 Commands Supported Report, conclude that S0 is the highest granted key
        ii. If not receiving any response, conclude that the S0 Security Class has not granted to the node

The *Role Type Specification* provides recommended timeouts when waiting for responses to get type commands.

7.2.7 Command Class Control Specific Requirements

Certain rules must be fulfilled depending on which command classes are controlled. The following subsections detail the requirements of special command classes. Details about individual Command Class control can be found in Section 6.

DT:00.11.0034.1 If a Device Type mandates that a Command Class MUST be controlled, that Command Class MUST be fully controlled. Partial control is not acceptable for Command Classes that have mandatory control requirements. Refer to *Command Class Control* for full and partial control definitions.

7.2.7.1 Anti-Theft Command Class

Control of this command class is limited to an entity that has been granted permission by the Z-Wave Alliance to control this Command Class and has been granted a locking entity ID. The list of granted locking entity IDs is defined in [22].

Any node controlling this Command Class without authorization will be failed in certification.

DT:00.11.002B.1 A node controlling this Command Class MUST NOT provide access to the locking functionality feature of the Command Class to a consumer/end-user of the node; i.e. end users MUST NOT be able to lock nodes themselves.

DT:00.11.002C.1 Control of this command class by a node which is also offered in a non-service market where an end consumer has access to network control features MUST use a different Product ID and Product Type ID between the service and consumer versions of the product.

7.2.8 SmartStart Requirements

DT:00.11.0026.1 A Z-Wave Plus v2 node MUST either support to be included in a network using SmartStart inclusion or provide SmartStart inclusion of other nodes. A controller providing SmartStart inclusion of other nodes MAY also support being included in a network using SmartStart inclusion.

DT:00.11.0027.1 A SmartStart product documentation MUST respect the requirements described in 3.10.4 Documentation related to SmartStart.

DT:00.11.0028.1 A node supporting to be included using SmartStart inclusion MUST have a QR code printed on its outside or its UI. (refer to Section 7.2.6.8.7). The QR code MUST indicate version 1 (SmartStart enabled nodes).

DT:00.11.0029.1 A SmartStart node MUST carry and keep the same Learn Mode DSK during its entire lifetime. Refer to Section 7.2.6.8.2 and Section 7.2.6.8.6.

7.2.9 Z-Wave Long Range Support

7.2.9.1 Nodes supporting to be included with Z-Wave Long Range

DT:00.11.002E.1 Nodes supporting Z-Wave Long Range MUST have a QR code printed on its outside or its UI (refer to 3.6.8.5 Mandatory DSK representations). The QR code MUST include a Type 4 (Supported Protocols) TLV block (refer to [28]) and indicate that Z-Wave Long Range is supported.

DT:00.11.002F.1 A node supporting Z-Wave Long Range MUST only request any Security Class that require Authentication for the S2 Bootstrapping during a Z-Wave Long Range SmartStart inclusion.

7.2.9.2 Nodes supporting to include using Z-Wave Long Range

DT:00.11.0030.1 Controller nodes that can include other nodes using Z-Wave Long Range MUST allow to configure the Bootstrapping Mode TLV of provisioning list entries to "Z-Wave Long Range SmartStart" (value 0x02) for entries that advertise Z-Wave Long Range as a supported protocol

DT:00.11.0031.1 A controller node MUST only grant Security Classes that require Authentication for the S2 Bootstrapping during a Z-Wave Long Range SmartStart inclusion.

7.2.10 Required Documentation

The following requirements for end user documentation apply to all Z-Wave Plus v2 compliant products. The documentation may be provided as product manuals, quick start guides, electronic help files, web pages, etc.

7.2.10.1 Terminology

DT:00.31.0001.1 The product documentation MUST use the terminology indicated in Table 7.3 for Z-Wave related functionality.

Table 7.3: Z-Wave Z-Wave Plus v2 documentation terminology

Z-Wave functionality	Documentation terminology	Example
Inclusion	Add	The process of adding a node to the Z-Wave network.
Exclusion	Remove	The process of removing a node from the Z-Wave network.
Replication	Copy	The process of copying network information from one controller to another.

7.2.10.2 Additional documentation required for Z-Wave Certification

DT:00.31.0002.1 In addition to the rules defined above the following technical documentation MUST be made available to the certification test lab upon submitting the product for certification:

- Documentation about how to activate any functionality available in the device related to Z-Wave behavior
- If any special procedures are REQUIRED to test any item in the certification form, such procedures MUST be clearly described
- If the product is a Z-Wave controller, documentation on how to send any controlled command from the controller MUST be included, refer to Section 6.

7.2.10.3 Documentation for Classic Inclusion and Exclusion

DT:00.31.0003.1 For Z-Wave end nodes Role Types, the documentation MUST describe:

- How to include and exclude the device in an existing network when using Classic inclusion.

DT:00.31.0004.2 For Z-Wave controller Role Types, the documentation MUST describe:

- How to include and exclude the device in an existing network using classic inclusion.
- How to include and exclude other devices.
- How to put the controller into learn mode to receive network information from another controller.

7.2.10.4 Documentation related to SmartStart

DT:00.31.0005.1 The documentation MUST describe:

- How to locate the DSK representation(s) on the product.
- How to access the DSK representation(s) via the UI, if available.

DT:00.31.0006.2 For nodes providing SmartStart functionalities, the documentation MUST include a short description of what is SmartStart. The following wording is RECOMMENDED:

> *SmartStart enabled products can be added into a Z-Wave network by scanning the Z-Wave QR Code present on the product with a controller providing SmartStart inclusion. No further action is required and the SmartStart product will be added automatically within 10 minutes of being switched on in the network vicinity.*

DT:00.31.0007.1 For controllers providing the SmartStart functionality, the documentation MUST describe:

- How to perform a secure inclusion of a SmartStart node (adding the node in the Node Provisioning List and powering up/installing the node)
- How to access and edit the Node Provisioning List.

DT:00.31.0019.1 For nodes supporting to be included using SmartStart, which do not follow the recommendation to enter SmartStart Learn Mode automatically after powering on, the documentation MUST describe:

- How to manually enter SmartStart Learn Mode if already powered up
- Whether or not the node keeps SmartStart Learn Mode enabled after a failed SmartStart inclusion.

DT:00.31.001A.1 For nodes supporting to be included using SmartStart, which do not follow the recommendation to stay in SmartStart Learn Mode forever until included, the documentation MUST describe:

- How long the node stays in SmartStart Learn Mode
- Whether or not the node keeps SmartStart Learn Mode enabled after a failed SmartStart inclusion.

7.2.10.5 Documentation related to devices from multiple manufacturers

DT:00.31.0008.1 The product documentation MUST include a section which describes how products from different manufacturers and product categories can be a part of the same Z-Wave network, and that the different mains powered nodes can act as repeaters regardless of manufacturers.

The following is the RECOMMENDED wording:

> *This product can be operated in any Z-Wave network with other Z-Wave certified devices from other manufacturers. All mains operated nodes within the network will act as repeaters regardless of vendor to increase reliability of the network.*

7.2.10.6 Documentation for Association Command Class

DT:00.31.0009.1 The documentation MUST include a description of the association groups available in the product.

DT:00.31.000A.1 Each group MUST include the following information:

- Grouping identifier
- Maximum number of devices that can be added to the group
- Description of how the association group is used and/or triggered by the product
- Description of any mapping between groups (e.g. Root Device mirrored End Point group)

7.2.10.7 Documentation for Configuration Command Class

DT:00.31.000B.1 If the product implements support of the Configuration Command Class, the documentation MUST include a description of each configuration parameter available in the product.

DT:00.31.000C.1 Each configuration parameter MUST be listed with the following information:

- Parameter number
- Description of parameter and its effect on the product
- Default value and allowed values
- Size (number of bytes)

DT:00.32.0001.1 The documentation SHOULD also list other configuration parameter properties such as read-only or advanced flag, etc.

7.2.10.8 Documentation for Wake Up Command Class

DT:00.31.000D.1 If the node supports the Wake Up Command Class, the product documentation MUST describe how to manually Wake Up the node.

7.2.10.9 Documentation for Security 2 Command Class

DT:00.31.000E.2 If a node based on an end node Role Type supports its Command Classes only when granted the Access Control Security key, the documentation MUST indicate that an S2 security enabled controller is required to operate the product.

DT:00.31.000F.1 The documentation MUST list the supported Command Classes, their version and their required Security class if any.

For example, a Lock Device Type list and a Binary Switch Device Type list are given in Table 7.4 and Table 7.5.

Table 7.4: Lock DT Supported Command Classes documentation example

Command Class	Version	Required Security Class
Association	2	S0 or Access Control
Association Group Information	3	S0 or Access Control
Basic	2	S0 or Access Control
Device Reset Locally	1	S0 or Access Control
Door Lock	4	S0 or Access Control
Firmware Update Meta Data	5	S0 or Access Control
Indicator	3	S0 or Access Control
Manufacturer Specific	1	S0 or Access Control
Multi Channel Association	3	S0 or Access Control
Powerlevel	1	S0 or Access Control
Security 0	1	None
Security 2	1	None
Supervision	1	None
Transport Service	2	None
Version	3	S0 or Access Control
Z-Wave Plus Info	2	None

Table 7.5: Binary Switch DT Supported Command Classes documentation example

Command Class	Version	Required Security Class
Association	2	Highest granted Security Class
Association Group Information	3	Highest granted Security Class
Basic	2	Highest granted Security Class
Binary Switch	2	Highest granted Security Class
Device Reset Locally	1	Highest granted Security Class
Firmware Update Meta Data	5	Highest granted Security Class
Indicator	3	Highest granted Security Class
Manufacturer Specific	1	Highest granted Security Class
Multi Channel Association	3	Highest granted Security Class
Powerlevel	1	Highest granted Security Class
Security 2	1	None
Supervision	1	None
Transport Service	2	None
Version	3	Highest granted Security Class
Z-Wave Plus Info	2	None

7.2.10.10 Documentation for Supervision Command Class

DT:00.31.0015.1 If a node supports the Supervision Command Class, the product documentation MUST describe the list of cases and/or conditions where it would issue a *Supervision Report Command* with a status indicating WORKING or FAIL, for a valid and supported set of parameters in the encapsulated command.

For example, it is not necessary to describe that a Supervision encapsulated *Association Set Command* would return FAIL if trying to establish a new association in a group that is already full.

7.2.10.11 Documentation for Basic Command Class

DT:00.31.0014.2 If the product supports Basic Command Class, the product documentation MUST include information on the usage of the Basic Command Class and the resulting product behavior

7.2.10.12 Documentation for Notification Command Class

DT:00.31.0015.1 If the product implements support of the Notification Command Class, the documentation MUST specify the implemented Notification Type(s) and Event(s).

7.2.10.13 Documentation for dynamic capabilities

DT:00.31.0010.1 If the product can alter its capabilities depending on a configuration parameter or based on a user interaction, the documentation MUST include a list of all events that can trigger capability change and describe:

- How to perform such actions
- What capabilities are being altered.

DT:00.31.0011.1 The product documentation MUST indicate to the end user that it is necessary to ask a controlling node to rediscover the product's capabilities after altering capabilities.

DT:00.31.0012.1 The product documentation MUST indicate that it is necessary to re-include the node in the network if the controller does not have any capability rediscovery option.

DT:00.31.0013.1 For nodes based on a controlling Device Type (4.3), the documentation MUST describe how an (advanced) end user can perform a capability rediscovery of a chosen node.

7.2.10.14 Documentation for Identity function

DT:00.31.0016.1 The product documentation MUST describe how product can be identified using the Indicator Command Class with the Indicator ID 0x50 (identify).

7.2.10.15 Documentation for Z-Wave Long Range

DT:00.31.0017.1 Nodes supporting Z-Wave Long Range MUST indicate in their documentation if they can be included and/or if they can include other nodes using Z-Wave Long Range.

7.2.10.15.1 Nodes supporting to include using Z-Wave Long Range

DT:00.31.0018.1 Controller nodes that can include other nodes using Z-Wave Long Range MUST indicate how to change the bootstrapping mode (Z-Wave SmartStart vs Z-Wave Long Range SmartStart) of provisioning list entries.

7.3 Z-Wave Plus v2 Device Type Definition

7.3.1 Optional Command Classes

DT:00.11.000F.1 Device Types MAY support optional Command Classes on top of the minimum mandatory set of supported Command Classes. However, the following Command Classes MUST NOT be supported optionally in a Device Type:

- Barrier Operator
- Color Switch
- Window Covering
- Multilevel Switch
- Thermostat Mode
- Thermostat Setpoint
- Thermostat Setback
- Sound Switch
- Simple AV Control
- Door Lock
- Binary Switch

DT:00.11.0010.1 It means that if supported, these command classes MUST fit the exact actuator Command Class list of a Device Type. If several actuator Command Classes not belonging to the Device Type need to be supported, they MUST be partitioned in End Points which match the actuator Command Class list of a Device Type.

Multi Channel Root Devices MAY still aggregate some of the above mentioned Command Classes from their end points for backwards compatibility. The optional Command Class rule applies:

- For the node (Root Device) if the node does not support Multi Channel Command Class
- For each and every end point if the node supports the Multi Channel Command Class

7.3.2 Supporting Device type overview

The Z-Wave Plus v2 certification program defines a new set of Device Type based on which Command Classes are supported. They are classified into 3 categories:

- Actuator supporting device types.
- Data reporting devices types
- Other devices types.

The list of Supporting Device Types is shown in Table 7.6, Table 7.7 and Table 7.8.

Table 7.6: Z-Wave Plus v2 Supporting Actuator Device Types overview

Device Type	Mandatory	Recommended options	Role Types
Lock	Door Lock, v4 Security 0 (S0)	User Code Entry Control	All
Motorized barrier	Barrier Operator Notification, v8 Security 0 (S0)		All
Color Switch	Color Switch, v3 Multilevel Switch v4 or Binary Switch v2	Multi Command	All
Window Covering	Window covering Multilevel Switch v4		All
Thermostat	Thermostat Mode, v3 Thermostat Setpoint	Clock (support) or Time (control) Multilevel Sensor Schedule, v4 Thermostat Setback	All
Sound Switch	Sound Switch		All
AV Control Point	Simple AV Control		All
Multilevel Switch	Multilevel Switch, v4		All
Binary Switch	Binary Switch, v2		All

Table 7.7: Z-Wave Plus v2 Supporting Data Reporting Device Types overview

Device Type	Mandatory	Recommended options	Role Types
Entry Control Keypad	Entry Control Security 0 (S0)		All
Multilevel Sensor	Multilevel Sensor, v11	Multi Command (control)	All
Notification Sensor	Notification, v8	Multi Command (control)	All
Meter Sensor	Meter, v5	Multi Command (control)	All
Central Scene	Central Scene, v3	Basic (control)	All

Table 7.8: Z-Wave Plus v2 Supporting Other Device Types overview

Device Type	Mandatory	Recommended options	Role Types
Repeater	No other Command Class than Section 7.2.2.1		AOEN
IR Repeater	IR Repeater Command Class		AOEN

7.3.3 Controlling Device type overview

The Z-Wave Plus v2 certification program defines a new set of Device Type for controllers based on which Command Classes are supported and controlled.

The controlling Device Type overview is shown in Table 7.9.

Table 7.9: Z-Wave Plus v2 Controlling Device Types overview

Device Type	Mandatory (support)	Mandatory (control)	Role Types
Gateway	CRC-16 Encapsulation Multi Command Node Provisioning Security 0 (S0) Time	Association, version 2 Association Group Information, version 3 Basic, version 2 Central Scene, version 3 CRC-16 Encapsulation Firmware Update Meta Data, version 5 Indicator, version 3 Meter, version 5 Multi Channel, version 4 Multi Channel Association, version 3 Multilevel Sensor, version 11 Notification, version 8 Security 0 (S0) Security 2 (S2) Version, version 2 Wake up, version 2	CSC
Generic Controller	Multi Command	Basic Indicator, version 3 The actuator Command Classes of at least 1 actuator Device Type Version, version 2 Wake up, version 2	CSC, SSC, RPC, PC, EN

7.3.4 From Z-Wave Plus to Z-Wave Plus v2 certification

Table 7.10 indicates the recommended transitions from a Z-Wave Plus Device Type to a Z-Wave Plus v2 Device Type.

Table 7.10: Equivalent Z-Wave Plus and Z-Wave Plus v2 Device Types

Z-Wave Plus DT(s)	Suggested Z-Wave Plus v2 DT(s)
On/Off Power Switch Power Strip Valve – Open/close Irrigation control	Binary Switch
Siren	Binary Switch Sound Switch
Door Lock – Keypad Lockbox	Lock
Light Dimmer Switch Fan Switch	Multilevel Switch
Set Top box TV Sub system Controller	Gateway Generic Controller
Remote Control – Multi purpose Remote control – Simple Wall controller	Generic Controller Central Scene
Sub Energy Meter Whole Home Meter	Meter Sensor
Gateway Central Controller	Gateway
Thermostat – HVAC Thermostat – Setback	Thermostat
Remote control -AV	Generic Controller
Window Covering	Window Covering

The following Device Types are unchanged and have the same equivalent Device Type in the Z-Wave Plus v2 certification program:

- Sensor - Notification
- Sensor - Multilevel
- AV Control Point
- Sound Switch
- Barrier Operator
- Entry Control keypad
- Repeater

The following Device Type is discontinued:

- Display - Simple

7.3.5 Actuator supporting types

7.3.5.1 Lock DT

The Lock Device Type is intended for nodes implementing a lock mechanism with optional handles. It can be a door lock, a lockbox as well as a safe.

7.3.5.1.1 Generic and Specific Device Class

DT:01.11.0001.1 The Lock Device Type MUST use the following Device Classes:

- GENERIC_TYPE_ENTRY_CONTROL (0x40)
- SPECIFIC_TYPE_DOOR_LOCK (0x01)

7.3.5.1.2 S2 Security Classes

DT:01.11.0002.1 The Root Device MUST request Access Control Security Class if it (or any End Point) uses this Device Type.

7.3.5.1.3 Mandatory Command Classes

DT:01.11.0003.1 The Lock MUST support the following Command Classes:

- Door Lock, version 4
- Basic, version 2
- Security 0 (S0)

Recommended optional command classes for advanced applications:

- User Code
- Entry Control
- Generic Schedule
- Authentication
- Authentication Media Write

7.3.5.1.4 Basic Command Class Requirements

DT:01.11.0004.1 The Basic Command Class MUST be mapped according to Table 7.11.

Table 7.11: Lock Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value)	Door Lock Operation Set (Door Lock Mode)
Basic Report (Current Value = 0x00)	Door Lock Operation Report (Door Lock Mode = 0x00)
Basic Report (Current Value = 0xFF)	Door Lock Operation Report (Door Lock Mode != 0x00)

7.3.5.2 Motorized Barrier DT

The Motorized Barrier Device Type is intended for barriers, gates or garage doors devices.

7.3.5.2.1 Generic and Specific Device Class

DT:02.11.0001.1 The Motorized Barrier Device Type MUST use the following Generic Device Class:

- GENERIC_TYPE_ENTRY_CONTROL (0x40)

DT:02.11.0002.1 The Motorized Barrier Device Type MUST use one of the following Specific Device Classes based on its capabilities within the Barrier Operator Command Class:

- SPECIFIC_TYPE_SECURE_GATE (0x06) if it can both open and close
- SPECIFIC_TYPE_SECURE_BARRIER_OPEN_ONLY (0x08) if it can open only
- SPECIFIC_TYPE_SECURE_BARRIER_CLOSE_ONLY (0x09) if it can close only

7.3.5.2.2 S2 Security Classes

DT:02.11.0003.1 The Root Device MUST request Access Control Security Class if it (or any End Point) uses this Device Type.

7.3.5.2.3 Mandatory Command Classes

DT:02.11.0004.1 The Motorized Barrier Devices MUST support the following Command Classes:

- Barrier Operator
- Notification, version 8
- Basic, version 2
- Security 0 (S0)

7.3.5.2.4 Basic Command Class Requirements

DT:02.11.0005.1 The Basic Command Class MUST be mapped according to Table 7.12.

Table 7.12: Motorized Barrier Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value)	Barrier Operator Set (Target Value)
Basic Report (Current Value = 0x00)	Barrier Operator Report (State = 0x00)
Basic Report (Current Value = 0xFF)	Barrier Operator Report (State > 0x00)

7.3.5.3 Color Switch DT

The Color Switch Device Type is intended for a lighting product having the ability to change its color.

7.3.5.3.1 Generic and Specific Device Class

DT:03.11.0001.1 The Color Switch Device Type MUST use the following Generic Device Classes:

- GENERIC_TYPE_SWITCH_BINARY (0x10) if supporting Binary Switch
- GENERIC_TYPE_SWITCH_MULTILEVEL (0x11) if supporting Multilevel Switch

DT:03.11.0002.1 The Color Switch Device Type MUST use the following Specific Device Classes:

- SPECIFIC_TYPE_COLOR_TUNABLE_BINARY (0x02) if supporting Binary Switch
- SPECIFIC_TYPE_COLOR_TUNABLE_MULTILEVEL (0x02) if supporting Multilevel Switch

7.3.5.3.2 Mandatory Command Classes

DT:02.11.0003.1 The Color Switch MUST support the following Command Classes:

- Color Switch, version 3
- Multilevel switch, version 4 or Binary Switch, version 2
- Basic, version 2

7.3.5.3.3 Basic Command Class Requirements

DT:02.11.0005.1 The Basic Command Class MUST be mapped according to Table 7.13 if supporting the Binary Switch Command Class or Table 7.14 if supporting the Multilevel Switch Command Class.

Table 7.13: Color Switch Device Type Basic mapping for Binary Switch

Basic Command	Mapped Command
Basic Set (Value)	Binary Switch Set (Target Value)
Basic Report (Current Value, Target Value, Duration)	Binary Switch Report (Current Value, Target Value, Duration)

Table 7.14: Color Switch Device Type Basic mapping for Multilevel Switch

Basic Command	Mapped Command
Basic Set (Value)	Multilevel Switch Set (Value)
Basic Report (Current Value, Target Value, Duration)	Multilevel Switch Report (Current Value, Target Value, Duration)

7.3.5.4 Window Covering DT

The Window Covering Device Type is intended for curtains or blinds allowing the end user to control the amount of light going through windows.

7.3.5.4.1 Generic and Specific Device Class

DT:04.11.0001.1 The Window Covering Device Type MUST use the following Generic Device Classes:

- GENERIC_TYPE_SWITCH_MULTILEVEL (0x11)

DT:04.11.0002.1 The Window Covering Device Type MUST use one of the following Specific Device Classes based on its capabilities within the Window Covering Command Class:

- SPECIFIC_TYPE_CLASS_A_MOTOR_CONTROL (0x05) if no position/endpoint awareness
- SPECIFIC_TYPE_CLASS_B_MOTOR_CONTROL (0x06) if endpoint aware
- SPECIFIC_TYPE_CLASS_C_MOTOR_CONTROL (0x07) if position and endpoint aware

7.3.5.4.2 Mandatory Command Classes

DT:04.11.0003.1 The Window Covering MUST support the following Command Classes:

- Window Covering, version 1
- Multilevel Switch, version 4 (MUST be redundant to Window Covering, i.e. actuating the same hardware)
- Basic, version 2

7.3.5.4.3 Basic Command Class Requirements

DT:04.11.0004.1 The Basic Command Class MUST be mapped according to Table 7.15.

Table 7.15: Window Covering Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value=0xFF)	Multilevel Switch Set (0xFF) if endpoint aware Start level change(Up/down) if not endpoint aware
Basic Set (Value=0x00)	Multilevel Switch Set (0x00) if endpoint aware Stop level change if not endpoint aware
Basic Set (Value=0x01..0x63)	Multilevel Switch Set (0x01..0x63) if position aware Start level change(Up/down) if not position aware
Basic Report (Current Value, Target Value, Duration)	Multilevel Switch Report (Current Value, Target Value, Duration) Current Value and Target Value MUST be set to 0xFE if not position aware

7.3.5.5 Thermostat DT

The Thermostat Device Type is intended by thermostats that support set points and modes. It is typically used for all mainstream thermostats that can support e.g. Heating, Cooling and Fans.

7.3.5.5.1 Generic and Specific Device Class

DT:05.11.0001.1 The Thermostat Device Type MUST use the following Device Classes:

- GENERIC_TYPE_THERMOSTAT (0x08)
- SPECIFIC_TYPE_THERMOSTAT_GENERAL_V2 (0x06)

7.3.5.5.2 Mandatory Command Classes

DT:05.11.0002.1 The Thermostat MUST support the following Command Classes:

- Thermostat Mode, version 3
- Thermostat Set Point
- Basic, version 2

Recommended optional command classes:

- Multilevel Sensor, supporting Sensor Type 0x01 (temperature)
- Clock
- Schedule, version 4
- Thermostat Setback

As an alternative to supporting Clock, the node can also control:

- Time

7.3.5.5.3 Basic Command Class Requirements

DT:05.11.0003.1 The Basic Command Class MUST be mapped according to Table 7.16.

Table 7.16: Thermostat Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value=0xFF)	Thermostat Mode Set (Mode = 0x01, 0x02 or 0x03)
Basic Set (Value=0x00)	Thermostat Mode Set (Mode = 0x00, 0x0B, 0x0C or 0x0D)
Basic Report (Current Value = 0xFF)	Thermostat Mode Report (Mode = 0x01, 0x02 or 0x03)
Basic Report (Current Value = 0x00)	Thermostat Mode Report (Mode = 0x00, 0x0B, 0x0C or 0x0D)

DT:05.11.0004.1 Other modes than 0x01, 0x02 or 0x03 MAY be mapped to Basic (Value=0xFF). In this case, it MUST be documented in the user manual.

DT:05.11.0005.1 Other modes than 0x00, 0x0B, 0x0C or 0x0D MAY be mapped to Basic (Value=0x00). In this case, it MUST be documented in the user manual.

7.3.5.6 Sound Switch DT

The Sound Switch Device Type is intended for products with the ability to issue sound notifications with a pre-programmed sound inventory. It can be used for a doorbell, chime, siren, alarm clock or any device issuing sounds.

7.3.5.6.1 Generic and Specific Device Class

DT:06.11.0001.1 The Sound Switch Device Type MUST use the following Device Classes:

- GENERIC_TYPE_AV_CONTROL_POINT (0x03)
- SPECIFIC_TYPE_SOUND_SWITCH (0x01)

7.3.5.6.2 Mandatory Command Classes

DT:06.11.0002.1 The Sound Switch MUST support the following Command Classes:

- Sound Switch
- Basic, version 2

7.3.5.6.3 Basic Command Class Requirements

DT:06.11.0003.1 The Basic Command Class MUST be mapped according to Table 7.17.

Table 7.17: Sound Switch Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value)	Sound Switch Tone Play Set (Tone Identifier)
Basic Report (Current Value = 0x00)	Sound Switch Tone Play Report (Tone Identifier = 0x00)
Basic Report (Current Value = 0xFF)	Sound Switch Tone Play Report (Tone Identifier > 0x00)

7.3.5.7 AV Control Point DT

The AV Control Point Device Type is intended for products with the ability to receive IR codes. It can be a TV, DVD player or any multimedia device that can now be also controller via Z-Wave.

7.3.5.7.1 Generic and Specific Device Class

DT:07.11.0001.1 The AV Control Point Device Type MUST use the following Device Classes:

- GENERIC_TYPE_AV_CONTROL_POINT (0x03)
- SPECIFIC_TYPE_NOT_USED (0x00)

7.3.5.7.2 Mandatory Command Classes

DT:07.11.0002.1 The AV Control Point Device Type MUST support the following Command Classes:

- Simple AV Control
- Basic, version 2

7.3.5.7.3 Basic Command Class Requirements

DT:07.11.0003.1 The Basic Command Class MUST be mapped according to Table 7.18.

Table 7.18: AV Control Point Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value)	Simple AV Set (Command). The associated AV Codes Commands to the values 0x00 and 0xFF chosen be the manufacturer MUST switch the node's main functionality On and Off. (such as play/pause or volume up/volume down)
Basic Report (Current Value)	None The reported value MAY indicate if the node's main functionality is On or Off. (such as play/paused) else current value SHOULD be set to 0x00.

7.3.5.8 Multilevel Switch DT

The Multilevel Switch Device Type is intended for products that can be switched between more than 2 discrete levels or states, such a light dimmer, water valve or a fan.

7.3.5.8.1 Generic and Specific Device Class

DT:08.11.0001.1 The Multilevel Switch Device Type MUST use the following Device Classes:

- GENERIC_TYPE_SWITCH_MULTILEVEL (0x11)
- SPECIFIC_TYPE_NOT_USED (0x00)

7.3.5.8.2 Mandatory Command Classes

DT:08.11.0002.1 The Multilevel Switch Device Type MUST support the following Command Classes:

- Multilevel Switch, version 4
- Basic, version 2

7.3.5.8.3 Basic Command Class Requirements

DT:08.11.0003.1 The Basic Command Class MUST be mapped according to Table 7.19.

Table 7.19: Multilevel Switch Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value)	Multilevel Switch Set (Value)
Basic Report (Current Value, Duration)	Multilevel Switch Report (Value, Duration)

7.3.5.9 Binary Switch DT

The Binary Switch Device Type is intended for any actuator functionality that can only be switched between 2 states (On and Off). It can be a valve, a light switch, a plug-in module.

7.3.5.9.1 Generic and Specific Device Class

DT:09.11.0001.1 The Binary Switch Device Type MUST use the following Device Classes:

- GENERIC_TYPE_SWITCH_BINARY (0x10)
- SPECIFIC_TYPE_NOT_USED (0x00)

7.3.5.9.2 Mandatory Command Classes

DT:09.11.0002.1 The Binary Switch MUST support the following Command Classes:

- Binary Switch, version 2
- Basic, version 2

DT:09.12.0001.1 If the node can measure energy, water or gas consumption, it is RECOMMENDED to support:

- Meter, version 5

7.3.5.9.3 Basic Command Class Requirements

DT:09.11.0003.1 The Basic Command Class MUST be mapped according to Table 7.20.

Table 7.20: Binary Switch Device Type Basic mapping

Basic Command	Mapped Command
Basic Set (Value)	Binary Switch Set (Value)
Basic Report (Current Value, Duration)	Binary Switch Report (Value, Duration)

7.3.6 Reporting supporting Device Types

7.3.6.1 Entry Control Keypad DT

The Keypad Device Type is intended for keypads or authentication devices reporting user input to a controlling application.

7.3.6.1.1 Generic and Specific Device Class

DT:11.11.0001.1 The Entry Control Keypad Device Type MUST use the following Device Classes:

- GENERIC_TYPE_ENTRY_CONTROL (0x40)
- SPECIFIC_TYPE_SECURE_KEYPAD (0x0B)

7.3.6.1.2 S2 Security Classes

DT:11.11.0002.1 The Root Device MUST request Access Control Security Class if it (or any End Point) uses this Device Type.

7.3.6.1.3 Mandatory Command Classes

DT:11.11.0003.1 The Entry Control Keypad MUST support the following Command Classes:

- Entry Control
- Security 0 (S0)

Recommended optional command classes:

- Indicator, version 3 (with other Indicator IDs than Identify)

7.3.6.1.4 Basic Command Class Requirements

DT:11.11.0004.1 Basic Command Class MUST NOT be supported

7.3.6.2 Multilevel Sensor DT

The Multilevel Sensor Device Type is intended for sensor reporting instantaneous numerical readings or measurements.

7.3.6.2.1 Generic and Specific Device Class

DT:12.11.0001.1 The Multilevel Sensor Device Type MUST use the following Device Classes:

- GENERIC_TYPE_SENSOR_MULTILEVEL (0x21)
- SPECIFIC_TYPE_ROUTING_MULTILEVEL_SENSOR (0x01)

7.3.6.2.2 Mandatory Command Classes

DT:12.11.0002.1 The Multilevel Sensor MUST support the following Command Classes:

- Sensor Multilevel, version 11

DT:12.12.0001.1 If the node issues more than one command regularly, it is RECOMMENDED to control:

- Multi Command

7.3.6.2.3 Basic Command Class Requirements

DT:12.11.0003.1 Basic Command Class MUST NOT be supported

7.3.6.3 Notification Sensor DT

The Notification Sensor Device Type is intended for sensors reporting events or local state changes.

7.3.6.3.1 Generic and Specific Device Class

DT:13.11.0001.1 The Notification Sensor Device Type MUST use the following Device Classes:

- GENERIC_TYPE_SENSOR_NOTIFICATION (0x07)
- SPECIFIC_TYPE_NOTIFICATION_SENSOR (0x01)

7.3.6.3.2 Mandatory Command Classes

DT:13.11.0002.1 The Notification Sensor MUST support the following Command Classes:

- Notification, version 8

DT:13.12.0001.1 If the node issues more than one command regularly, it is RECOMMENDED to control:

- Multi Command

7.3.6.3.3 Basic Command Class Requirements

DT:13.11.0003.1 Basic Command Class MUST NOT be supported

7.3.6.4 Meter Sensor DT

The Meter Sensor Device Type is intended for sensors measuring cumulated values. The most typical application is an electricity meter, but it can also be used for sensor measuring gas or water consumption.

7.3.6.4.1 Generic and Specific Device Class

DT:14.11.0001.1 The Meter Sensor Device Type MUST use the following Device Classes:

- GENERIC_TYPE_METER (0x31)
- SPECIFIC_TYPE_NOT_USED (0x00)

7.3.6.4.2 Mandatory Command Classes

DT:14.11.0002.1 The Meter Sensor MUST support the following Command Classes:

- Meter, version 5

DT:14.12.0001.1 If the node issues more than one command regularly, it is RECOMMENDED to control:

- Multi Command

DT:14.12.0002.1 For advanced metering applications, it is RECOMMENDED to support the following command classes:

- Meter Table Monitor, version 2
- Meter Table Push Configuration
- Rate Table Monitor

7.3.6.4.3 Basic Command Class Requirements

DT:14.11.0003.1 Basic Command Class MUST NOT be supported

7.3.6.5 Central Scene DT

The Central Scene Device Type is intended for nodes with buttons or GUI allowing to report user input/button press to a central application, which in turn will actuate or control other nodes.

This Device Type can be used for nodes such as wall switches or panels with a set of buttons. Such devices will send Scene Notifications to the Lifeline destination in order to trigger scenes.

7.3.6.5.1 Generic and Specific Device Class

DT:15.11.0001.1 The Central Scene Device Type MUST use the following Device Classes:

- GENERIC_TYPE_WALL_CONTROLLER (0x18)
- SPECIFIC_TYPE_NOT_USED (0x00)

7.3.6.5.2 Mandatory Command Classes

DT:15.11.0002.1 The Central Scene MUST support the following Command Classes:

- Central Scene, version 3

7.3.6.5.3 Basic Command Class Requirements

DT:15.11.0003.1 Basic Command Class MUST NOT be supported

7.3.6.5.4 Recommended options

DT:15.12.0001.1 The Central Scene Device Type SHOULD implement controlling capabilities using Association Groups. It is RECOMMENDED to have a group issuing Basic Set Commands.

DT:15.12.0002.1 Multi Channel End Point SHOULD NOT implement this Device Type.

7.3.7 Other Device Types

7.3.7.1 Repeater DT

The Repeater Device Type is intended for nodes being part of the network with no application functionalities. Such nodes help as Z-Wave repeaters and strengthen the network reliability.

7.3.7.1.1 Role Type

DT:21.11.0001.1 The Repeater Device Type MUST use the AOEN Role Type.

7.3.7.1.2 Generic and Specific Device Class

DT:21.11.0002.1 The Repeater Device Type MUST use the following Device Classes:

- GENERIC_TYPE_REPEATER_END_NODE (0x0F)
- SPECIFIC_TYPE_REPEATER_END_NODE (0x01)

7.3.7.1.3 Mandatory Command Classes

DT:21.11.0003.2 The Repeater Device Type MUST NOT support any other Application Command Class than the list defined in Section 7.2.2.1 *Root Device level*. Additional Management Command Classes MAY be supported.

DT:21.11.0004.1 The Repeater Device Type MUST NOT control any Command Class.

7.3.7.1.4 Basic Command Class Requirements

DT:21.11.0005.1 Basic Command Class MUST NOT be supported

7.3.7.2 IR Repeater DT

The IR Repeater Device Type is intended for nodes having the ability to read or repeat IR signals. They do not have any other application functionality. Such repeater nodes also help as Z-Wave repeaters and strengthen the network reliability.

7.3.7.2.1 Role Type

DT:22.11.0001.1 The IR Repeater Device Type MUST use the AOEN Role Type.

7.3.7.2.2 Generic and Specific Device Class

DT:22.11.0002.1 The Repeater Device Type MUST use the following Device Classes:

- GENERIC_TYPE_REPEATER_END_NODE (0x0F)
- SPECIFIC_TYPE_IR_REPEATER (0x03)

7.3.7.2.3 Mandatory Command Classes

DT:22.11.0003.1 The IR Repeater MUST support the following Command Classes:

- IR Repeater, version 1

DT:22.11.0004.1 The IR Repeater Device Type MUST NOT support any additional Command Class (other than the mandatory list above and the list defined in Section 7.2.2.1 *Root Device level*.

DT:22.11.0005.1 The IR Repeater Device Type MUST NOT control any Command Class.

7.3.7.2.4 Basic Command Class Requirements

DT:22.11.0006.1 Basic Command Class MUST NOT be supported

7.3.8 Controlling Device Types

7.3.8.1 Gateway DT

The Gateway Device Type is intended for all gateway controllers that provide access in and potentially
DT:31.13.0002.1 out of the Z-Wave network as well as extensive controlling capabilities over Z-Wave. This DT MAY provide transparent access for all types of IP Packets between several network technologies.

7.3.8.1.1 Role Type

DT:31.11.0001.1 The Gateway Device Type MUST use the CSC Role Type.

7.3.8.1.2 Generic and Specific Device Class

DT:31.11.0002.1 The Gateway Device Type MUST use the following Device Classes:

- GENERIC_TYPE_STATIC_CONTROLLER (0x02)
- SPECIFIC_TYPE_GATEWAY (0x07)

7.3.8.1.3 S2 Security Classes

DT:31.11.0003.1 If bootstrapped in another network, it MUST request all S2 Security Classes. (S2 Access Control, S2 Authenticated and S2 Unauthenticated)

7.3.8.1.4 Mandatory Command Classes

DT:31.11.0004.1 The Gateway MUST support the following Command Classes in a Z-Wave network:

- CRC-16 Encapsulation
- Inclusion Controller
- Multi Command
- Security 0 (S0)
- Time

DT:31.11.0005.2 The Gateway SHOULD support Node Provisioning Command Class in a Z-Wave network.

DT:31.11.0005.2 The Gateway MUST support the following Command Classes in an IP network:

- Z/IP, version 4

DT:31.12.0001.1 If the Z/IP Gateway relies on a Z/IP client to provide application functionalities, it SHOULD support the following Command Classes in an IP network:

- Z/IP Gateway
- Z/IP ND
- Z/IP Portal
- Mailbox
- Network Management Proxy, version 2
- Network Management Inclusion, version 3
- Network Management Basic, version 2

DT:31.11.0006.3 The Gateway MUST provide full control of the following Command Classes:

- Association, version 2

- Association Group Information, version 3
- Basic, version 2
- Central Scene, version 3
- CRC-16 Encapsulation
- Firmware Update Meta Data, version 5
- Indicator, version 3
- Meter, version 5
- Multi Channel, version 4
- Multi Channel Association, version 3
- Multilevel Sensor, version 11
- Notification, version 8
- Security 0 (S0)
- Security 2 (S2)
- Version, version 2
- Wake up, version 2

DT:31.13.0001.1 A Gateway Device Type MAY provide a subset of its capabilities (supported and controlled command classes) if it has the Secondary Controller Role in a network.

DT:31.11.0008.1 A Gateway Device Type MUST provide full control of any Command Class that it controls. It MUST NOT provide partial control for any Command Class.

7.3.8.1.5 Recommended options

DT:31.12.0002.1 A Gateway Device Type SHOULD support reading and interpreting data form legacy sensors supporting the following Command Classes:

- Alarm Sensor
- Binary Sensor
- Alarm/Notification, version 1.
- Multilevel Sensor
- Meter

DT:31.12.0003.1 In order to achieve this, it is RECOMMENDED to implement a database of known devices.

DT:31.12.0004.2 A Gateway controller SHOULD control the Command Classes from all actuator Device Types (Table 7.6).

7.3.8.1.6 Basic Command Considerations

DT:31.11.0007.1 Basic Command Class MUST NOT be supported.

7.3.8.2 Generic Controller DT

The Generic Controller Device Type is intended for all more constrained or simple controllers allowing users to make basic use of the Z-Wave network and controlling a pre-defined set of actuator nodes as well as unknown actuator nodes.

7.3.8.2.1 Generic and Specific Device Class

DT:32.11.0001.1 The Generic Controller Device Type MUST use the following Device Classes:

- GENERIC_TYPE_GENERIC_CONTROLLER (0x01)
- SPECIFIC_TYPE_NOT_USED (0x00)

7.3.8.2.2 Mandatory Command Classes

DT:32.11.0002.1 The Generic Controller MUST support the following Command Classes:

- Multi Command

DT:32.11.0003.3 The Generic Controller MUST provide full control of the following Command Classes:

- Basic
- Indicator, version 3
- The mandatory actuator Command Classes of at least one actuator Device Type
- Version, version 2
- Wake up, version 2

DT:32.13.0001.1 A Generic Controller Device Type MAY provide full control or partial control of any other Command Class not listed above.

7.3.8.2.3 Recommended options

DT:32.12.0001.2 A Generic Controller SHOULD control Command Classes from as many actuator Device Types as possible.

7.3.8.2.4 Basic Command Considerations

DT:32.11.0004.1 Basic Command Class MUST NOT be supported.

8 Role Type Specification

8.1 Introduction

8.1.1 Purpose

This document describes the Z-Wave Plus Role Types. The purpose of the Role Type is to provide a high level definition of how Z-Wave nodes must react from a Z-Wave networking perspective.

This document is not meant to be read in full. It is aimed at being a scalable documentation process for network specific functionality for various Z-Wave devices. It should be read together with the Device Type specification [34], which highlights what Role Types should be used for different Device Types. A device will typically have one Role Type associated with it, but in some cases there can be more than one. The developer now only needs to look at one Role Type to determine the implementation of the network specific functionality to pass certification.

It is however necessary to understand how the Central Static Controller (CSC) works as most devices will heavily depend on it for direct communication.

8.2 Z-Wave Compliance Overview

RT:00.11.0001.1 The following sections present Z-Wave properties applying to all Z-Wave Plus Role Types defined in this document. Requirements presented in this chapter MUST be respected by all Z-Wave Plus devices

8.2.1 SIS Assignment

8.2.1.1 Non-SIS capable Primary Controllers

A Z-Wave network may have no SIS capable controller. For instance this is the case if the network consists of a Portable Controller (PC) which is used to include a number of Always On End Nodes (AOEN). In this case, the PC acts as the Primary Controller.

RT:00.11.0006.2 If no SIS is present in the network, when including a controller supporting SIS functionality, a non-SIS capable Primary Controller MUST assign the SIS role to the newly included controller.

8.2.1.2 SIS capable controllers

RT:00.11.0007.2 All controllers that support the SIS functionality MUST accept to become SIS upon request from a Primary Controller.

RT:00.11.0008.2 A controller that supports SIS functionality MUST assume the SIS role when creating a new network.

8.2.1.3 SIS return route assignment

RT:00.11.0009.1 When the SIS is present, an including node MUST always assign SIS return route when including an end node.

8.2.2 Network Inclusion and Exclusion

Learn Mode

RT:00.11.000A.1 A Z-Wave Plus compliant node MUST support both direct-range and Network Wide Inclusion (NWI).

Inclusion Process outlines the inclusion process.

Add Mode

Add mode is used by a controller for including a new node to a network.

RT:00.11.003E.2 The SIS MUST show the new added nodes in the list of included nodes after an inclusion has been carried out by an Inclusion Controller.

8.2.3 Security bootstrapping

8.2.3.1 Security 0 Command Class

RT:00.21.0001.1 Controllers MUST be able to perform Security 0 bootstrapping if they support the Security 0 Command Class. Refer to [34].

RT:00.21.0002.1 If a controller has the Inclusion Controller role in a network and includes a node that supports Security 0 Command Class only (i.e. does not support Security 2 Command Class), it MUST perform Security 0 bootstrapping immediately after including the node.

RT:00.21.0003.1

If the SIS and the Inclusion Controller both support the Inclusion Controller Command Class, the inclusion controller MUST NOT perform S0 bootstrapping unless instructed by the SIS with an Inclusion Controller Initiate Command (S0_INCLUSION).

RT:00.21.0005.1 If the SIS and the Inclusion Controller both support the Inclusion Controller Command Class, the SIS MUST NOT perform S0 bootstrapping. The SIS should instruct the Inclusion Controller to perform S0 bootstrapping or interview the included node non-securely.

RT:00.21.0004.1 If an error happens during S0 bootstrapping of an S0 capable controller, the included controller MAY refuse to provide network functions (others than Learn Mode). In this case, the included controller MUST indicate to the user that it needs to be excluded and re-included in the Z-Wave network.

8.2.3.1.1 Upgrading non-secure networks

RT:00.23.0001.1 If a controller is included in a non-secure network as an inclusion controller, it MAY start using its own S0 network key and perform S0 bootstrapping with newly included nodes.

A controller MUST NOT start using its own S0 network key if S0/S2 bootstrapping failed.

8.2.3.2 Security 2 Command Class

The following sections describe requirements for controllers supporting the Security 2 Command Class

8.2.3.2.1 Bootstrapping capabilities

Security 2 mandates certain functionalities depending on the controller's role in the network.

If a controller has the SIS role:

RT:00.21.0006.1 - It MUST support the SIS side of the Inclusion Controller Command Class

RT:00.21.0007.1 - It MUST perform Security 2 bootstrapping.

RT:00.21.0008.1 - It MUST support inclusion of nodes that implement any combination of Security 2 Security Classes

RT:00.21.0009.1 - It MUST have input and display method for support of all Security Classes.

RT:00.21.0006.1 - It MAY support inclusion using CSA

If a controller has the Inclusion Controller role:

RT:00.21.000A.1 - It MUST support the Inclusion Controller side of the Inclusion Controller Command Class

RT:00.21.000B.1 - It MUST NOT perform Security 2 bootstrapping

If a controller has the Primary Controller role:

RT:00.21.0003.1 - It MAY perform Security 2 bootstrapping

8.2.3.2.2 Granting Security Classes

RT:00.21.000C.1 A controller with a user interface for PIN code input (and optionally a QR scanning capability) MUST comply with following when bootstrapping S2 nodes:

- It MUST grant membership of all requested Classes if the joining node requests membership of the S2 Access Control Class (unless specified otherwise by a user).
- It MAY ask the user for confirmation before granting S2 Authenticated Class key if the node does not request membership of the S2 Access Control Class.
- It SHOULD provide a way to inspect and adjust the list of the Security Class memberships that will be granted to the joining node

A constrained controller with no QR scanning capability and no user interface for PIN code input
RT:00.21.000D.1 MUST comply with following when bootstrapping S2 nodes:

- It MUST grant membership of the S2 Unauthenticated Class if the joining node requests membership of the S2 Unauthenticated Class.
- It MUST abort the S2 bootstrapping entirely (grant no key) if the joining node does not request membership of the S2 Unauthenticated Class.

8.2.3.2.3 Informing the user about security

If a node has been security bootstrapped with the S0 Command Class in a S2 capable network, the
RT:00.21.000E.1 SIS/Primary controller MUST issue a warning message to the user informing that the node has not
RT:00.22.0001.1 been included securely. The SIS/Primary controller SHOULD request a new NIF to the included node after security bootstrapping to verify if the included node supports S2 before issuing the message to the user.

This is made to ensure that the end user is aware of which security level a node has been bootstrapped and therefore identify if a S0 downgrade attack took place during bootstrapping or if a non-S2 inclusion controller bootstrapped the joining node.

If an S2 node has not been granted the highest requested S2 key during bootstrapping, the SIS/Primary
RT:00.21.000F.1 controller MUST issue a warning message to the user informing that the node has not been included with the highest security. This is OPTIONAL if the user has actively chosen which keys to grant and security bootstrapping completed successfully.

In an Inclusion Controller scenario, the SIS' UI may not be active during S2 bootstrapping. In this case, the following rules apply:

- If the SIS automatically grants unauthenticated key for a node that request the S2 Unauthen-
RT:00.23.0005.1 ticated Class, it MAY notify the end user the next time it uses the UI.
- RT:00.22.0002.1 If the SIS timed out during S2 bootstrapping, it SHOULD instruct the end user that the node needs to be excluded and re-included.

8.2.4 Device Reset Locally support

RT:00.11.000B.1 If a device can be reset to factory default locally on the device, the device MUST be able to issue a Device Reset Locally Command via its Lifeline to notify the Lifeline destination that the device has been reset to its factory default state. The product documentation MUST include instructions on how to perform a reset to factory default operation.

RT:00.11.000C.1 If a device cannot be locally (or manually) reset to factory default, the device MUST NOT implement the Device Reset Locally functionality and MUST NOT list the Device Reset Locally Command Class identifier in the NIF.

RT:00.11.000D.1 If a device is reset, it MUST perform the reset operation regardless of whether the delivery of the Device Reset Locally Notification is successful or not. It is RECOMMENDED that devices implement a mechanism that allows the user to determine when the reset operation is completed.

When a node is reset:

- RT:00.11.000E.1 it MUST forget its current HomeID and consider itself excluded from the network.
- RT:00.13.0001.1 The configuration of *Application Command Classes* MAY stay unchanged (e.g configuration parameters, Thermostat Setpoint, Clock, Door lock Timeout configuration, User codes, …)
- The configuration of other Command Classes (Section 3, Section 4 or Section 5) MUST be reset
RT:00.11.000F.1 to default (i.e. S2 keys are forgotten, Associations and Wake-Up configurations are cleared, etc.)

8.2.5 Node interview and response timeouts

RT:00.11.0010.1 During a node capability discovery or interview, as well as traffic generated due to user activation, a controlling nodes MUST timeout waiting for responses (reports) as part of the capability discovery or controlling scenarios.

Two timers named CommandTime and ReportTime are used for timing out during a node discovery interview. Illustrations are given for secure and non-secure cases in Figure 8.1 and Figure 8.2

- CommandTime is measured by the application

RT:00.12.0001.1 - ReportTime timeout SHOULD be set to CommandTime + 1 second.

The communication flow MUST be as shown in Figure 8.1 and Figure 8.2

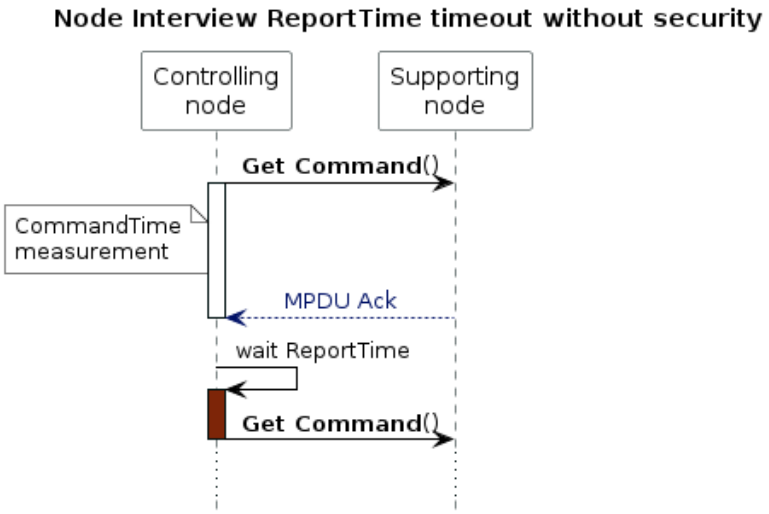


Figure 8.1: Node Interview ReportTime timeout without security

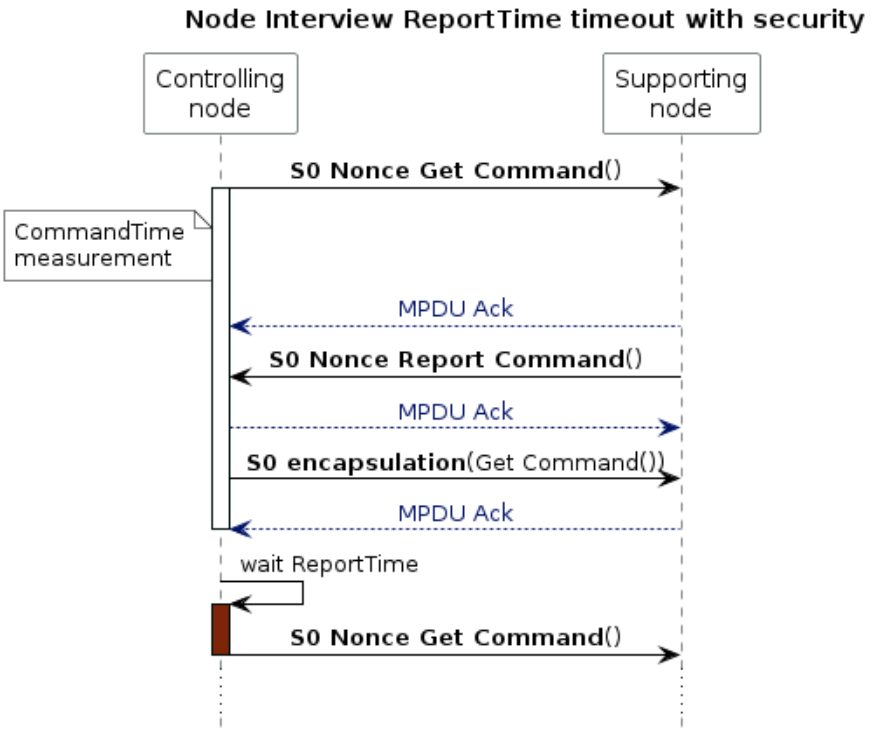


Figure 8.2: Node Interview ReportTime timeout with security

8.2.6 Polling Devices

A controlling device may monitor nodes or issue requests for status information.Communication patterns include, but are not limited to, the transmission of a:

- No Operation (NOP) Command to verify that a node is operational
- Get Command requesting status information in a Report Command
- Set Command followed by a Get Command requesting status information in a Report Command

RT:00.11.0011.1 Communication MUST be considered polling if a controlling device autonomously sends requests to one or more nodes in a repeating fashion to monitor nodes or to get information from nodes. This applies to any combination of commands.

RT:00.11.0012.1 Z-Wave is a radio technology with limited bandwidth. Therefore, it is NOT RECOMMENDED to use polling. If used, polling communication MUST comply with the requirements stated in the following subsections *Polling with no errors*, *Polling with transmit error* and *Polling with missing Report Frame*

RT:00.11.0013.1 Communication MUST NOT be considered as polling if:

- A node issues one or more commands in a burst initiated by a user action. This applies to any combination of commands; also requests.
- A node issues one or more commands initiated by the inclusion of another node. This applies to any combination of commands; also requests.

8.2.6.1 Polling with no errors

Two timers named CommandTime and PollTime are used for polling requirements with no error. Illustrations are given for secure and non-secure cases in Figure 8.3 and Figure 8.4

The following requirements apply to the normal case where a polling request is successful:

RT:00.11.0014.1 • CommandTime MUST be measured by the application

RT:00.11.0015.1 • The application MUST wait PollTime before polling any other node

RT:00.12.0002.1 • PollTime SHOULD be 10 seconds + CommandTime or more

RT:00.11.0016.1 • PollTime MUST NOT be less than 1 second + CommandTime

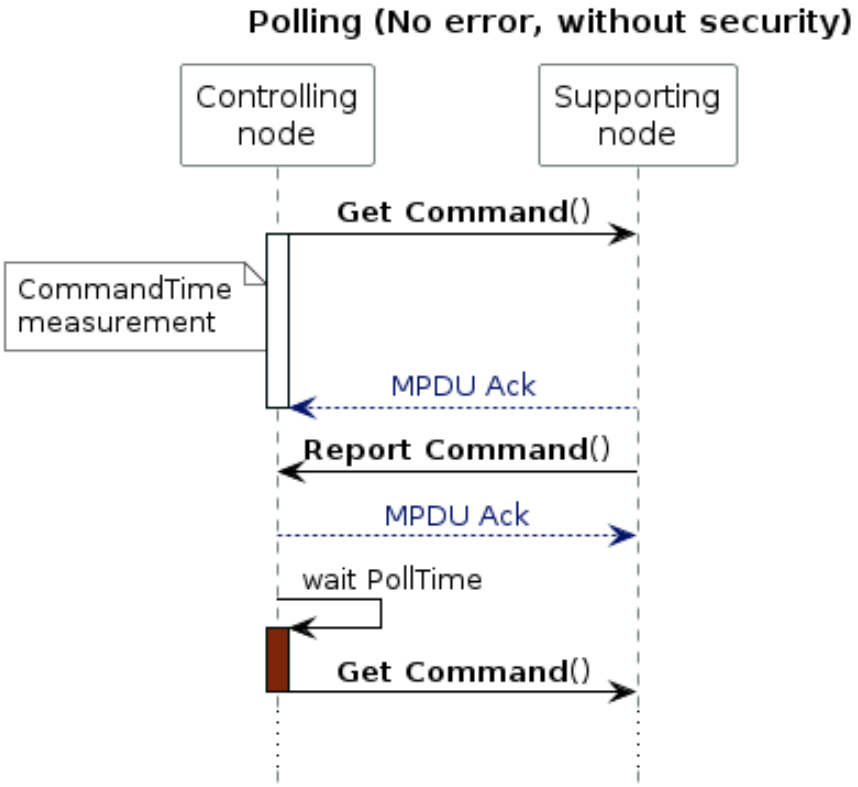


Figure 8.3: Polling (No errors, without security)

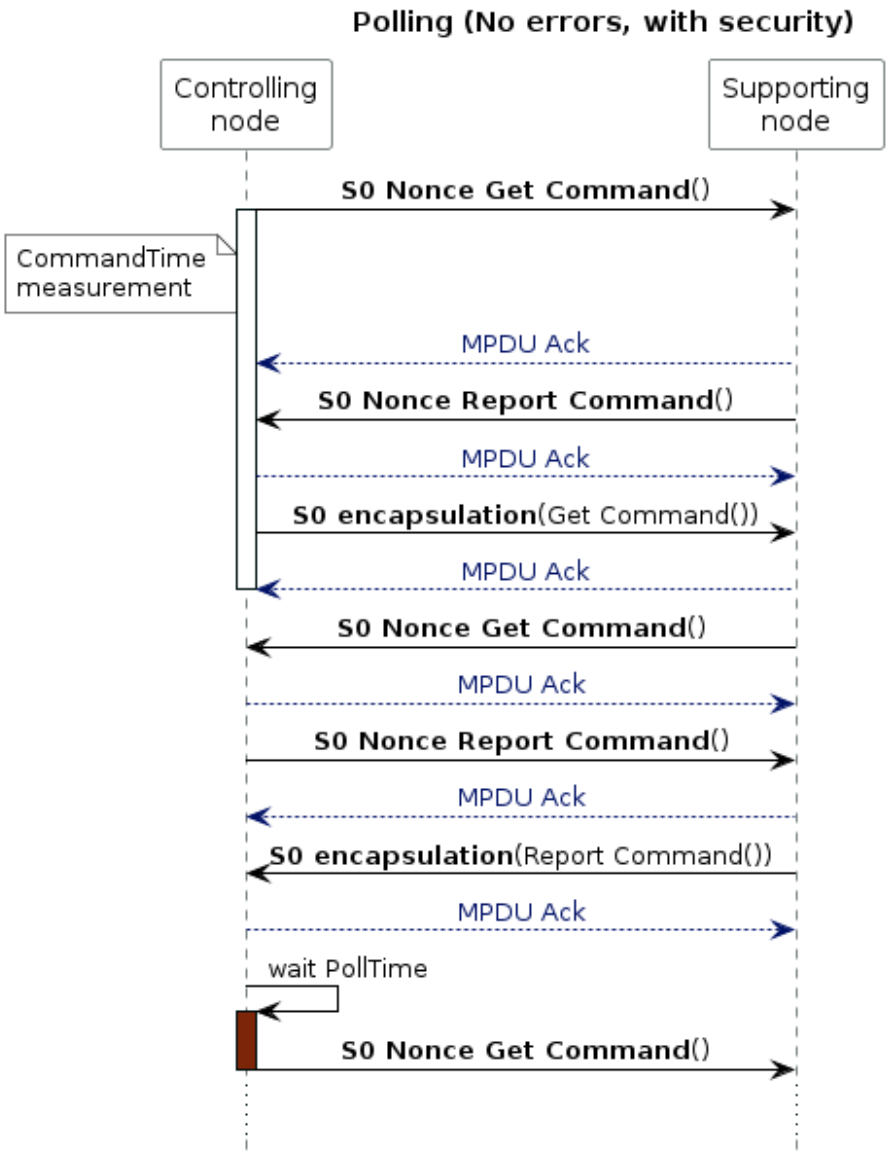


Figure 8.4: Polling (No errors, with security)

8.2.6.2 Polling with transmit error

Two timers named CommandTime and PollTime are used for polling requirements with transmission error. Illustrations are given for secure and non-secure cases in Figure 8.5 and Figure 8.6.

RT:00.11.0017.1 Note that in the case of a missing Ack, the Sending node MUST transmit the Get Command 3 times before considering the Ack to be missing. CommandTime is measured from the first Get Command transmission to the timeout.

The following requirements apply to the case where a polling request is not successful.

RT:00.11.0018.1
- If the transmission fails, the application MUST wait PollTime before polling any other node. PollTime MUST be 10 seconds + CommandTime or more

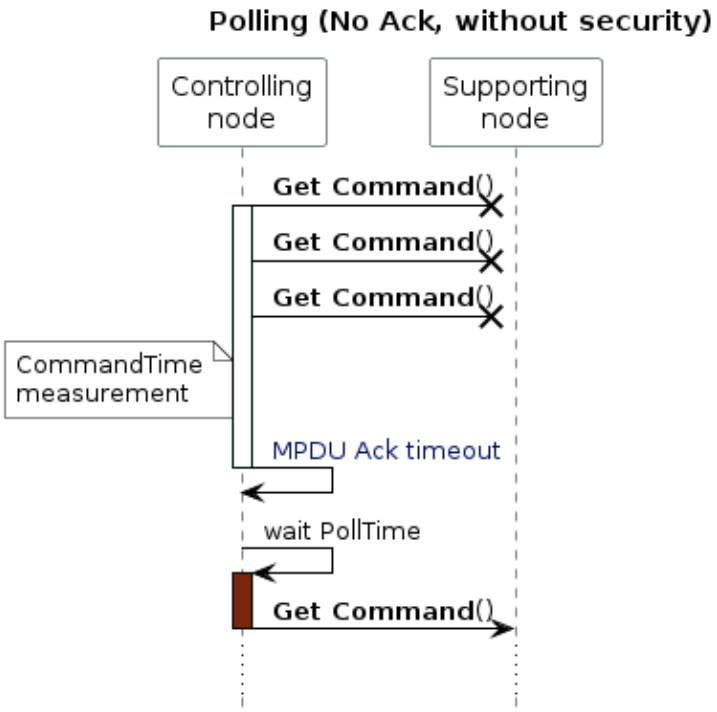


Figure 8.5: Polling (No Ack, without security)

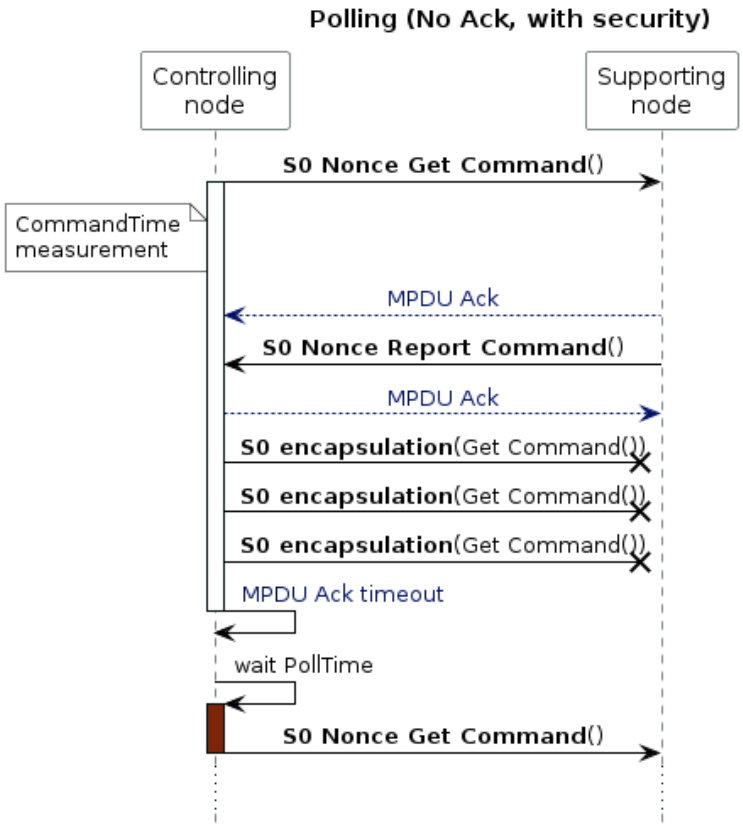


Figure 8.6: Polling (no ack with Security 0)

8.2.6.3 Polling with missing Report Frame

Two timers named CommandTime and ReportTime are used for polling requirements when transmission is successful but with missing report. Illustrations are given for secure and non-secure cases in Figure 8.7 and Figure 8.8.

The following requirements apply to the case where a polling request is successful but no Report frame is received.

RT:00.11.0019.1
- The application MUST wait ReportTime for the reply from node X before polling any other node

RT:00.11.001A.1
- ReportTime MUST be CommandTime + 10 seconds or more

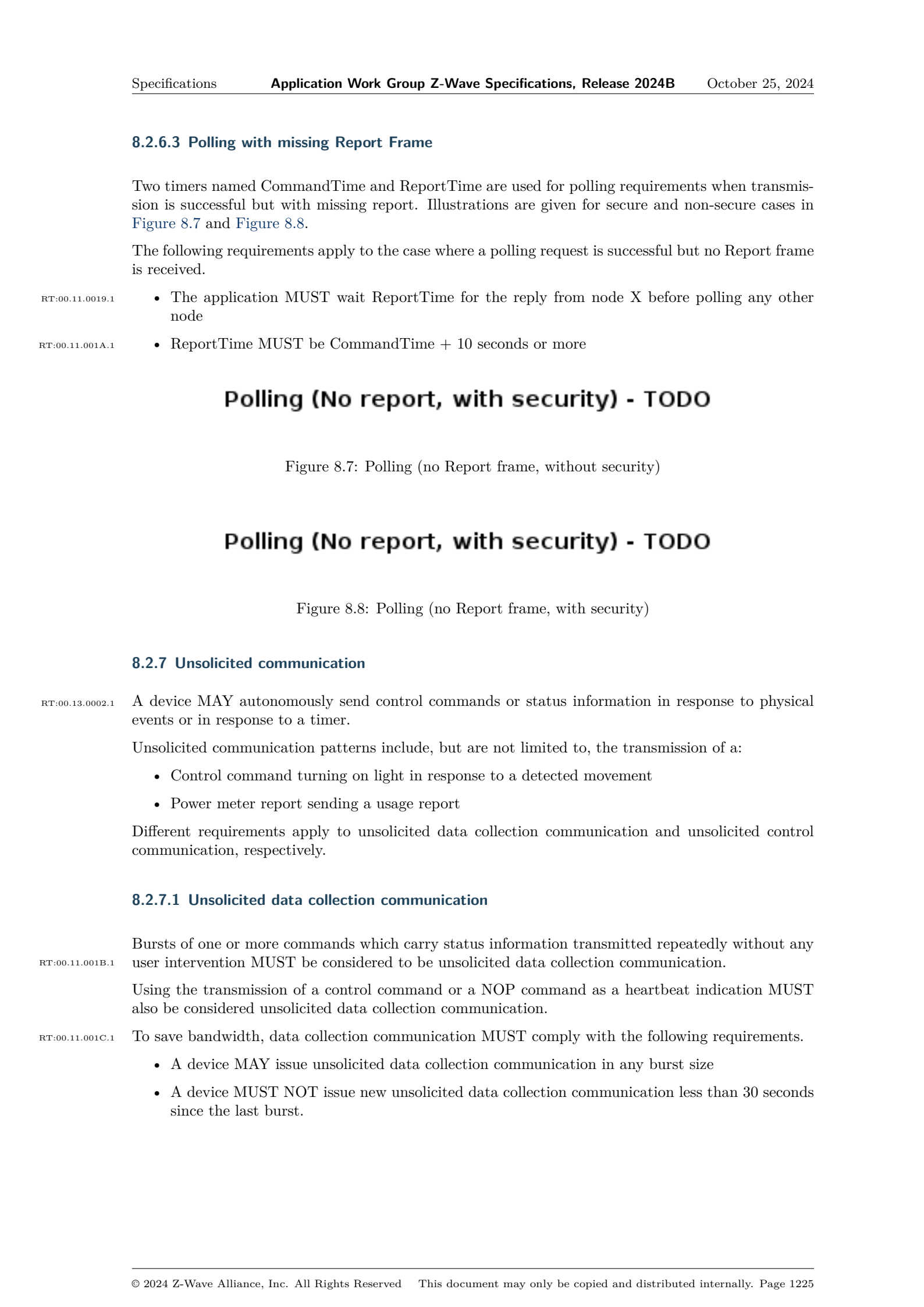


Figure 8.7: Polling (no Report frame, without security)

Figure 8.8: Polling (no Report frame, with security)

8.2.7 Unsolicited communication

RT:00.13.0002.1 A device MAY autonomously send control commands or status information in response to physical events or in response to a timer.

Unsolicited communication patterns include, but are not limited to, the transmission of a:

- Control command turning on light in response to a detected movement
- Power meter report sending a usage report

Different requirements apply to unsolicited data collection communication and unsolicited control communication, respectively.

8.2.7.1 Unsolicited data collection communication

RT:00.11.001B.1 Bursts of one or more commands which carry status information transmitted repeatedly without any user intervention MUST be considered to be unsolicited data collection communication.

Using the transmission of a control command or a NOP command as a heartbeat indication MUST also be considered unsolicited data collection communication.

RT:00.11.001C.1 To save bandwidth, data collection communication MUST comply with the following requirements.

- A device MAY issue unsolicited data collection communication in any burst size
- A device MUST NOT issue new unsolicited data collection communication less than 30 seconds since the last burst.

8.2.7.2 Unsolicited control communication

RT:00.11.001D.1 Bursts of one or more control commands initiated by a user action, a physical event or a time trigger MUST be considered control communication. Control communication MUST comply with the following requirements:

- A device MAY issue unsolicited control communication in any burst size.
- A device MAY issue unsolicited control communication at any interval since the last burst

8.2.8 Runtime communication

8.2.8.1 Routing

RT:00.11.001E.1 A Z-Wave Plus node MUST use by default the last working route to communicate with a target node. An illustration is given in Figure 8.9

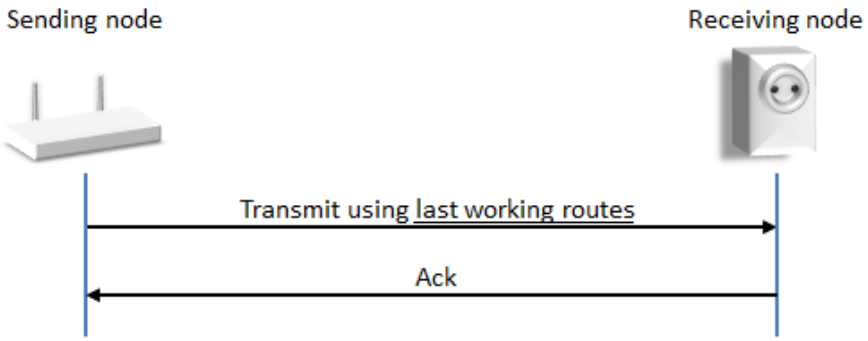


Figure 8.9: Successful transmission using last working routes

RT:00.11.001F.1 Over time, there is a risk that nodes are moved or stop working. To ensure that nodes adapt to changing network topology and failing repeaters, a Z-Wave Plus node MUST enable dynamic route resolution. Dynamic route resolution consists of trying the following routes:

- Last working routes
- Calculated routes
- Explorer Frame

Illustrations are given in Figure 8.10 and Figure 8.11.

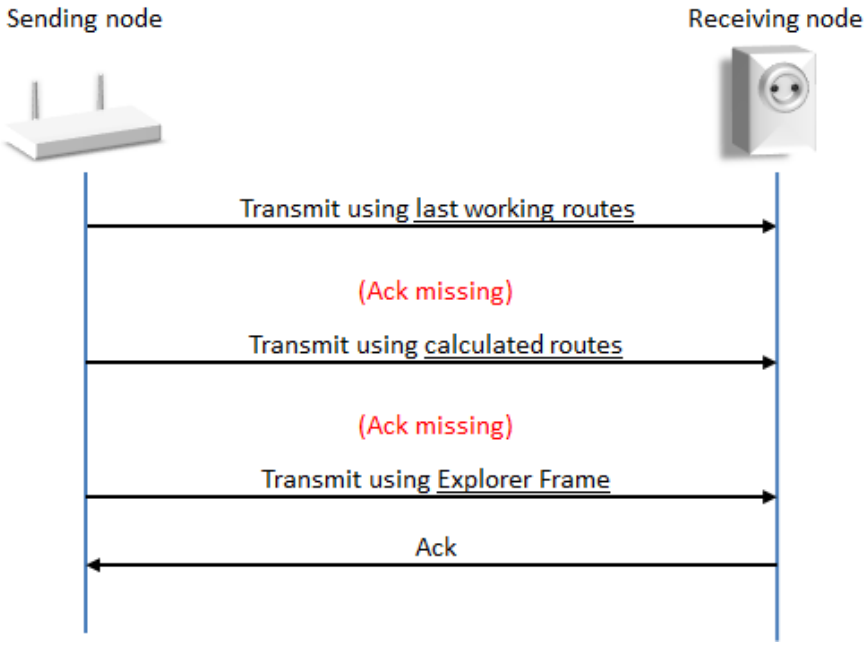


Figure 8.10: Successful transmission using Explorer Frame

RT:00.11.0020.1 A node MUST perform 3 routing attempts based on last working routes and/or calculated routes before sending an Explorer Frame. As outlined in Figure 8.10, controllers may calculate routes using the local neighbor map.

RT:00.13.0003.1 Listening Sleeping End Nodes (LSEN) and Reporting Sleeping End Nodes (RSEN) MAY use return routes injected by a controller. The outlined sequence of transmission attempts is handled entirely by the routing protocol.

In case the destination is not reachable, all routed transmission attempts will fail and ultimately, the routing protocol will have to give up delivering the frame. After a failed transmission, the application
RT:00.13.0004.1 MAY try to transmit again in case a new event occurs, e.g. because the user issues a new button press.

The steps in Figure 8.11 involve at least three routing attempts. When all routing attempts are unsuccessful, it is very unlikely that any other transmission attempt to the same target will succeed.
RT:00.12.0003.1 The sending node SHOULD give up the frame transmission.

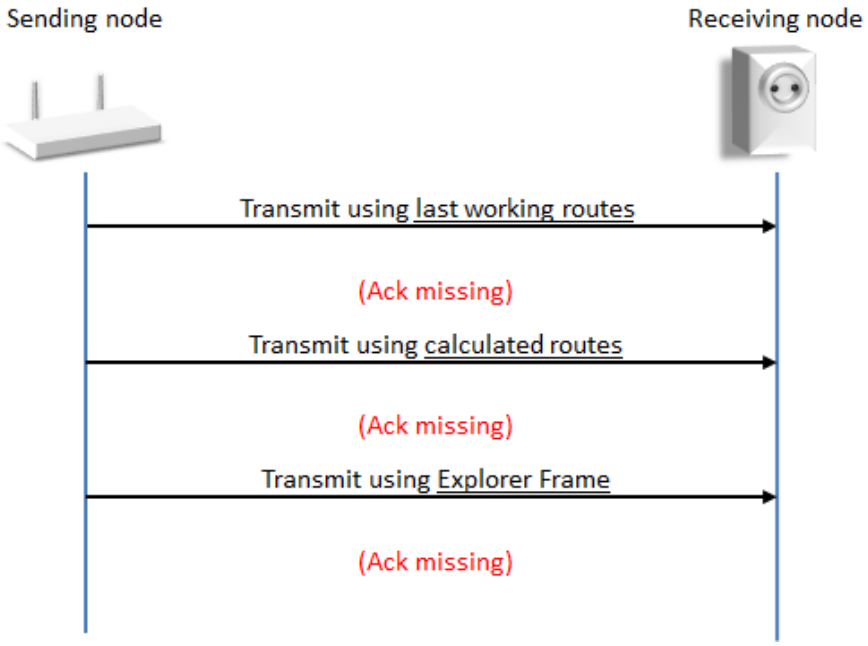


Figure 8.11: Unsuccessful transmission

RT:00.13.0005.1 Nodes based on a controller role type MAY skip transmission attempts if they are associated to a non-existing NodeID.

8.2.8.2 Wake-Up communication timeout protection

A battery powered node supporting Wake-Up communication sends a Wake Up Notification Command to get attention when it is awake and receives a Wake Up No More Information Command when it can safely return to sleep.

RT:00.12.0004.1 A battery powered Z-Wave Plus node supporting Wake-Up communication SHOULD implement a timeout mechanism which makes the node return to sleep if the node does not receive a Wake Up No More Information Command.

RT:00.11.0021.2 If no Wake Up No More Information Command is received from the Wake Up destination, the node MUST respond to the Wake Up destination until 10 seconds have elapsed since the last transmission or reception of an application frame supported by the node, a NOP frame or a Request Node Info Frame with the Wake Up destination.

An illustration is given in Figure 8.12.

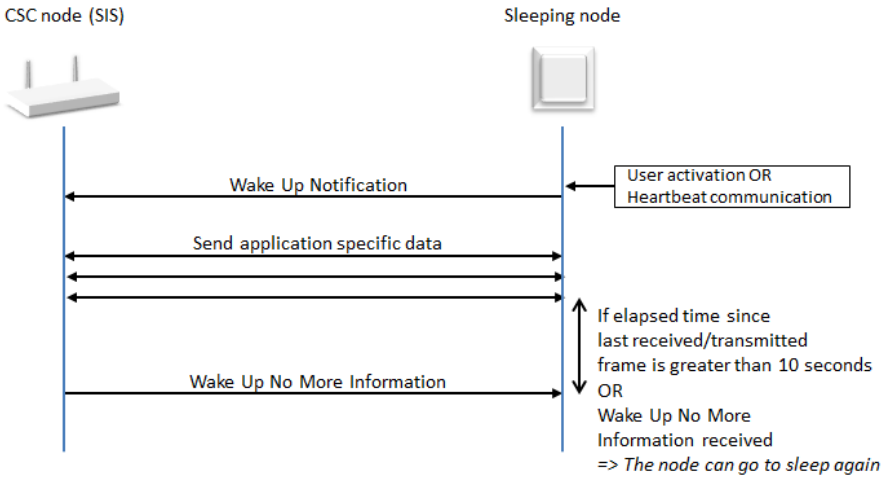


Figure 8.12: Wake Up Command Class

8.2.9 Network maintenance

RT:00.12.0005.1 The network rediscovery (Request neighbor update) feature SHOULD only be used as last resort in case the runtime communication fails.

8.2.10 SmartStart requirements

8.2.10.1 Support requirements

8.2.10.1.1 SmartStart learn mode activation

RT:00.11.0023.1 A node supporting SmartStart inclusion MUST enter SmartStart Learn Mode by default when ready after powering up, regardless of network inclusion status.

RT:00.11.0024.1 A node supporting SmartStart inclusion MUST fall back on SmartStart Learn Mode after deactivating Learn Mode.

8.2.10.1.2 Higher Inclusion Request Interval

If a very power-constrained battery node is designed to settle at a higher Max Inclusion Request Interval (*aNwkSmartStartMaxInclusionRequestInterval* in [35]) than the default 512 seconds, this value MUST be advertised in the node's provisioning information (QR Code, refer to [34] and [28]).

RT:00.11.0025.1

8.2.10.2 Control requirements

RT:00.11.002B.1 A controller providing control of the SmartStart functionality is NOT REQUIRED to support the SmartStart functionality and support being included in a network using SmartStart inclusion.

8.2.10.2.1 Command Class support

RT:00.11.002E.2 A Z/IP Gateway providing the SmartStart functionality MUST support the following Command Classes on the IP side:

- Network Management Inclusion Command Class, version 3 or newer

RT:00.12.0009.1 A Z/IP Gateway providing the SmartStart functionality SHOULD support the following Command Classes on the IP side:

- Node Provisioning Command Class

8.2.10.2.2 User interface

RT:00.11.0032.3 A controller providing control of the SmartStart functionality MUST:

- Provide a method for the end user to view the Node Provisioning List entries with their network inclusion status (included/ not included or failed).
- Provide a method for the end user to manually add and remove entries in the Node Provisioning List.
- Provide a method for the end user to edit available settings for each entry in the Node Provisioning List. (e.g., Inclusion setting, Advanced joining). A controller application MAY provide no available settings.
- Support S2 inclusion with authentication using the DSK PIN code.

RT:00.11.0033.1 If a user removes a node from the Node Provisioning List and the node is still included in the Z-Wave network, the controller MUST inform the end user that the node will stay in the network and requires to be excluded manually or reset to factory default in order to leave the Z-Wave Network.

RT:00.11.0034.1 A controller MUST inform the end user that S2 only (non-SmartStart) nodes present in the Provisioning List require to perform a classic inclusion to add them into the Z-Wave network.

8.2.10.2.3 QR Code scanning capability

RT:00.12.0006.1 A controller providing the SmartStart functionality SHOULD provide a QR Code scanning capability.

If the controller offers a QR Code scanning capability:

RT:00.11.0035.1 - It MUST support the addition of nodes using QR Code format defined in [29] in its Provisioning List when scanning the QR Code.

RT:00.12.0007.1 - It SHOULD support scanning of S2 only QR codes representing the DSK String prefixed with "zws2dsk:". (example: "zws2dsk:34028-23669-20938-46346-33746-07431-56821-14553") in order to simplify the S2 bootstrapping process.

8.3 Role Type Overview

Z-Wave Role Types is used as part of the Z-Wave Plus certification program. Role types define how battery and network functionalities must be implemented.

This is to provide better uniformity and hence ensuring better interoperability between Z-Wave Plus devices.

Role types are backwards compatible with Z-Wave products certified under earlier certification programs. The Role Types are device specific and hence the Device Type will define which Role Type(s) a given device can support.

Table 8.1 shows an overview of Role Types which are described in details in Chapter 5.

Table 8.1: Role Type Overview

Role Type	Abbreviation	Repeater	Power source	Can be SIS	Network Setup	Lifeline Setup	Report Through Lifeline	Direct controllable	Heart beat communication
Central Static Controller	CSC	✓	Mains	✓	✓	✓	✓	✓	✗
Sub Static Controller	SSC	✓	Mains	✗	✓	✗	✓	✓	✗
Portable Controller	PC	✗	Battery	✗	✓	✗	✓	✗	✗
Reporting Portable Controller	RPC	✗	Battery	✗	✓	✗	✓	✗	✓
Portable End Node	PEN	✗	Battery	✗	✗	✗	✓	✗	✗
Always On End Node	AOEN	✓	Mains	✗	✗	✗	✓	✓	✗
Listening Sleeping End Node	LSEN	✗	Battery	✗	✗	✗	✓	✓	✗
Reporting Sleeping End Node	RSEN	✗	Battery	✗	✗	✗	✓	✗	✓
Network Aware End Node	NAEN	✓	Mains	✗	✗	✗	✓	✓	✗

The following functionalities depend on the actual Role Type:

Repeater: Indicates whether the device can act as repeater in the network. This requires an always listening device, which can accommodate any routing requests immediately.

Power source: Mains powered devices are accessible immediately and are always listening devices. Battery powered devices focus on battery lifetime extension as one of the primary objectives.

Can be SIS: The node supports the Static Update Controller (SUC) and SUC node ID Server (SIS) functions. When SIS functionality is enabled, the controller also takes the Primary Controller role. All other controllers operate as Inclusion Controllers, i.e. they can request that nodes are included/excluded. If a SIS is present in the network, it is RECOMMENDED that all other devices update their network topology once a day and before configuring associations.

Network setup: The node is capable of managing the network and inclusion/exclusion of nodes.

Setup lifeline: The node is able to configure lifeline associations.

Report through lifeline: The node MUST be able to report events via a lifeline association to a central home control application.

Direct controllable: Mains powered devices and battery devices configured as Frequently Listening (FL) nodes can be controlled at any time.

Heart beat communication: Operating as a sleeping device, the node is able to connect at given intervals to a central home control application to allow delivery of messages from other devices. Such a node supports the Wake Up Command Class.

8.3.1 Detecting the Role Type of a device

RT:00.11.0036.1 The Role Type of a node can be requested via the Z-Wave Plus Info Command Class, which MUST be listed as the first supported Command Class in the Node Information Frame (NIF) by all Z-Wave Plus nodes. For details about Z-Wave Plus Info Command Class, refer to Section 3.

Table 8.2: Role Type identifiers

Role Type	Value	Identifier
Central Static Controller (CSC)	0x00	*ROLE_TYPE_CONTROLLER_CENTRAL_STATIC*
Sub Static Controller (SSC)	0x01	*ROLE_TYPE_CONTROLLER_SUB_STATIC*
Portable Controller (PC)	0x02	*ROLE_TYPE_CONTROLLER_PORTABLE*
Reporting Portable Controller (RPC)	0x03	*ROLE_TYPE_CONTROLLER_PORTABLE_REPORTING*
Portable End Node (PEN)	0x04	*ROLE_TYPE_END_NODE_PORTABLE*
Always On End Node (AOEN)	0x05	*ROLE_TYPE_END_NODE_ALWAYS_ON*
Reporting Sleeping End Node (RSEN)	0x06	*ROLE_TYPE_END_NODE_SLEEPING_REPORTING*
Listening Sleeping End Node (LSEN)	0x07	*ROLE_TYPE_END_NODE_SLEEPING_LISTENING*
Network Aware End Node (NAEN)	0x08	*ROLE_TYPE_END_NODE_NETWORK_AWARE*

8.4 Role Type Definitions

The following sections describe requirements for individual Role Types. Each Role Type has requirements categorized in the following subsections:

1. Protocol Requirements
2. Setup
3. Runtime Configuration
4. Runtime Communication

The Setup subsection describes the specific requirements for a given Role Type during and after a network inclusion. Figure 8.13 shows the different steps of a node setup / commissioning.

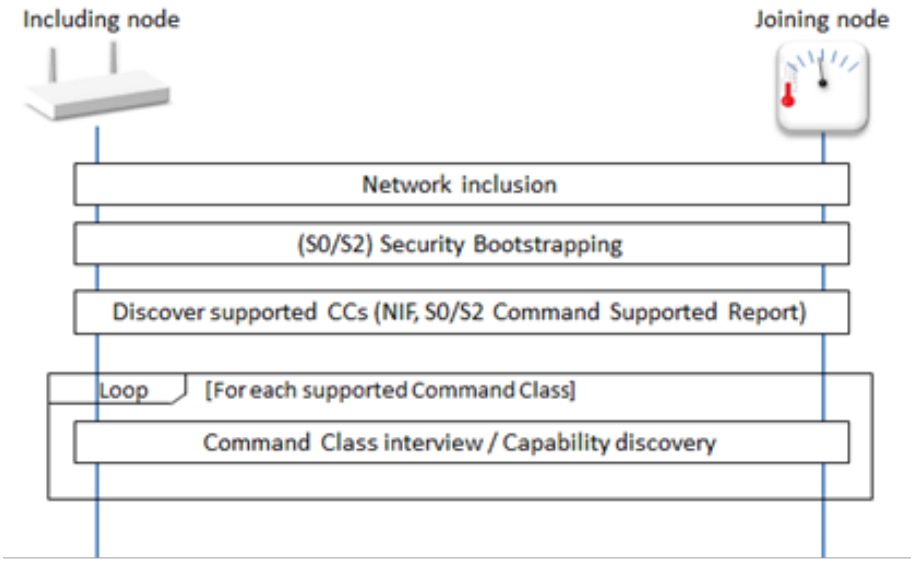


Figure 8.13: Node Setup/Commissioning

Network inclusion

The network inclusion process is described in *Inclusion Process*. Additional recommendations are given for the different Role Types.

(S0/S2) Security bootstrapping

The security (Security 0 or Security 2) bootstrapping takes place immediately after the network inclusion. Refer to Section 4.

Discover supported Command Classes

The controlling node reads the supported command classes before interviewing each of them.

Command Class interview

Each role type specifies some requirements that must be observed during the Command Class interview:

- **Lifeline configuration:**:

  When interviewing the Association or Multi Channel Association Command Class, the including controller sets up the lifeline association if it is the SIS. If a SIS is present in the network, the destination NodeID of the Lifeline group MUST be the SIS NodeID. Requirements are detailed for each Role Type in the following sections. Refer to [34] and Section 7 for Lifeline group definition

RT:00.11.0037.1

- **Battery considerations:**

  Some requirements apply for battery powered nodes, supporting the Wake Up Command Class. Details are given for each Role Type.

Commissioning and runtime phases

The commissioning phase is defined as the period after a node's inclusion during which the Security bootstrapping, Lifeline configuration, Wake Up configuration and initial device interview is made by a controller.

RT:00.12.0008.1 It is RECOMMENDED that a controller does not display a newly included node as ready to be operated during the commissioning phase.

The commissioning phase is considered over when the initial interview is completed or latest 10 minutes after the network inclusion.

Once the commissioning phase is over, a node is said to be in the runtime phase.

8.4.1 Central Static Controller (CSC)

The Central Static Controller Role Type is intended for always powered devices which are capable of operating as a central controller. The CSC will be the central device for most network communications and other devices will rely on it for unsolicited information via the lifeline association to the CSC (which is also the SIS). This will enable the user to receive key information without having to perform major network configuration tasks.

The CSC is typically a router, central gateway or some sort of central communication panel.

8.4.1.1 CSC Protocol Requirements

RT:01.11.0001.1 The CSC MUST respect requirements described in Section 8.2.

RT:01.11.0002.1 The CSC MUST support the Static Update Controller (SUC) and SUC node ID Server (SIS) functions.

RT:01.11.0003.1 The CSC MUST be mains powered and MAY have a battery back-up.

RT:01.11.0004.1 The CSC MUST set the listening flag to 1 in its NIF.

RT:01.11.0005.1 The CSC MUST support and control the S0 and S2 Command Classes.

RT:01.11.0006.2 The CSC MUST support the following network roles:

- SIS
- Secondary controller (if Learn Mode is supported)
- Inclusion controller (if Learn Mode is supported)

8.4.1.1.1 If first node in the network

RT:01.11.0007.3 If the CSC is the first node in the network, it MUST set itself the SIS role and MUST support the following network functions:

- Include new nodes (“Add mode”)
- Exclude nodes
- Remove failing nodes

RT:01.13.0001.2 Additionally, it MAY support the following network function:

- Replace failing nodes
- Learn Mode

RT:01.11.0008.1 It MUST NOT be possible to activate Learn Mode if the CSC is the SIS and other nodes are included in the network.

8.4.1.2 CSC setup

8.4.1.2.1 Inclusion process

RT:01.12.0001.1 It is RECOMMENDED to use soft buttons for activating learn mode and add mode on a CSC Role Type.

8.4.1.2.2 Lifeline configuration

When including a node,

RT:01.11.0009.1
- if the CSC is the SIS: it MUST set itself as the Association group ID 1 (Lifeline) destination.

RT:01.13.0003.1
- if the CSC is not the SIS: it MAY set the SIS' NodeID as the Association group ID 1 (Lifeline) destination.

RT:01.11.0016.1 The CSC MUST assign a return route for the SIS after setting the lifeline of End Node Role Types.

Details and requirements about establishing the Lifeline are provided in *Association Group Information (AGI) Command Class, version 1-3* and *Multi Channel Association Command Class, version 2-5*.

8.4.1.2.3 CSC including a SSC, PC, RPC or NAEN

Battery considerations

If the CSC is the SIS and the included node is of Role Type RPC:

RT:01.11.000A.1
- The CSC MUST configure the Wake Up Interval Set Command destination NodeID to its NodeID.

RT:01.11.000B.1
- The CSC MUST send a Wake Up No More Information Command when the CSC has no more command to transmit.

8.4.1.2.4 CSC including a EN, LSEN or RSEN

Battery considerations

RT:01.11.000D.1 If the CSC is the SIS and the included node is of Role Type PEN or RSEN, the CSC MUST:

- configure the Wake Up Interval Set Command destination NodeID to its NodeID.
- send a Wake Up No More Information Command when the CSC has no more command to transmit.

If the CSC is the SIS and the included node is of Role Type PEN:

RT:01.11.000E.1
- If the node advertises Wake-Up Capabilities (Wake-Up Command Class, version 2 or newer), the Wake Up Interval Set Command Seconds field MUST be within the allowed range

RT:01.12.0002.1 If the CSC is not the SIS, it SHOULD NOT send a Wake Up Interval Set Command to the included node.

RT:01.11.0010.1 If the CSC is not the SIS and sends a Wake Up Interval Set Command, the destination NodeID MUST be the SIS' NodeID.

8.4.1.2.5 CSC including an AOEN

Battery considerations

None.

8.4.1.2.6 CSC including another CSC

RT:01.11.0013.1 If the CSC is included by another CSC, the included CSC MUST take the Inclusion Controller role and MUST support the following network functions:

- Include new nodes ("Add mode")
- Exclude nodes
- Learn mode
- Remove failing node

RT:01.13.0002.1 Additionally, it MAY support the following network function:

- Replace failing node

Battery considerations

None.

8.4.1.2.7 CSC included by a PC, RPC, SSC

RT:01.11.0014.1 The CSC MUST accept to take the SIS role when a PC, RPC or SSC assigns it to the included CSC.

Lifeline configuration

RT:01.12.0003.1 If the CSC was assigned the SIS role, previously added nodes may have no lifeline associations. The CSC SHOULD create lifeline associations in all existing nodes that are directly reachable.

Battery considerations

None.

8.4.1.3 CSC Runtime Configuration

RT:01.11.0015.1 The CSC MUST instruct a reporting node (RPC, RSEN, PEN) to return to sleep after application data has been delivered to the node. This is done by sending a Wake Up No More Information Command. An illustration is given in Figure 8.12.

8.4.1.4 CSC Runtime Communication

No requirements

8.4.2 Sub Static Controller (SSC)

The Sub Static Controller Role Type is intended for static controllers which are not suitable as central controllers. It is aimed at applications that require a static controller to manage a subset of nodes. It is typically offered as a bundled package with e.g. sensors.

8.4.2.1 SSC Protocol Requirements

RT:02.11.0001.1 The SSC MUST respect requirements described in Section 8.2

RT:02.11.0002.1 The SSC MUST be mains powered and MAY have battery back-up.

RT:02.11.0003.1 The SSC MUST set the listening flag to 1 in its NIF.

RT:02.11.0004.1 The SSC MUST NOT support the SIS functionality.

RT:02.12.0001.1 The SSC SHOULD NOT configure lifeline associations.

RT:02.11.0005.1 The SSC MUST support the following network roles:

- Primary controller
- Secondary controller
- Inclusion controller

8.4.2.1.1 If first node in the network

RT:02.11.0006.1 If the SSC is the first node in the network, it MUST take the Primary Controller role and MUST support the following network functions:

- Include new nodes (“Add mode”)
- Exclude nodes
- Learn mode

RT:02.11.0007.1 It MUST NOT be possible to activate Learn Mode if the SSC is the Primary Controller and other nodes are included in the network.

8.4.2.2 SSC Setup

8.4.2.2.1 Inclusion process

RT:02.12.0002.1 It is RECOMMENDED to use a physical push button for activating learn mode and a soft button for activating add mode on a SSC Role Type.

8.4.2.2.2 Lifeline configuration

No requirement

8.4.2.2.3 SSC including a CSC

RT:02.11.0008.1 If the SSC is the Primary Controller and a CSC is added to the network, the SSC MUST assign the SIS role to the CSC.

RT:02.13.0001.1 If the SSC is the Primary Controller and has previously included some Wake Up nodes, it MAY re-assign the Wake Up destination NodeID to the CSC/SIS for the previously included Wake Up nodes at the next Wake Up Notification.

RT:02.11.0009.1 The SSC becomes an inclusion controller and MUST support the following network functions:

- Include new modes ("Add mode")
- Exclude nodes
- Learn mode

Battery considerations

None.

8.4.2.2.4 SSC including an RPC, PEN or RSEN

RT:02.12.0003.1 If there is a SIS in the network, the SSC SHOULD NOT send a Wake Up Interval Set Command to the included node. If there is a SIS in a network and the SSC sends a Wake Up Interval Set Command, RT:02.11.000A.1 the destination NodeID MUST be the SIS' NodeID.

RT:02.12.0004.1 If there is no SIS present in the network, the SSC SHOULD send a Wake Up Interval Set Command RT:02.11.000B.1 with its own NodeID as destination. If issuing a Wake Up Interval Set Command, the SSC MUST respect the following rules:

- If the included node is of Role Type RPC, PEN or RSEN:

RT:02.12.0005.1

  - The SSC SHOULD set the Wake Up Interval Set Command Seconds field to the default Wake Up time advertised by the included node.

- If the included node is of Role Type PEN:

  - RT:02.11.000C.1 If the node advertises Wake-Up Capabilities (Wake-Up Command Class, version 2 or newer), the Wake Up Interval Set Command Seconds field MUST be within the allowed range.

8.4.2.2.5 SSC including an SSC, PC, AOEN, LSEN, or NAEN

Battery considerations

None.

8.4.2.3 SSC Runtime Configuration

No requirements.

8.4.2.4 SSC Runtime communication

No requirements.

8.4.3 Portable Controller (PC)

The Portable Controller Role Type is intended for portable controllers that can setup and maintain a Z-Wave network but do not require unsolicited reporting. It is typically used by home control remotes that control a few lights.

8.4.3.1 PC Protocol Requirements

RT:03.11.0001.1 The PC MUST respect requirements described in Section 8.2

RT:03.11.0002.1 The PC MUST be battery powered and support the Battery Command Class.

RT:03.11.0003.1 The PC MUST set the listening flat to 0 in its NIF.

RT:03.12.0001.1 The PC SHOULD NOT configure lifeline associations when adding nodes to the network.

RT:03.11.0004.1 The PC MUST support the following network roles:

- Primary controller
- Secondary controller
- Inclusion controller

8.4.3.1.1 If first node in the network

RT:03.11.0005.1 If the PC is the first node in the network, it MUST take the Primary Controller role and MUST support the following network functions:

- Include new nodes (“Add mode”)
- Exclude nodes
- Learn mode

RT:03.11.0006.1 It MUST NOT be possible to activate Learn Mode if the PC is the Primary Controller and other nodes are included in the network.

8.4.3.2 PC Setup

8.4.3.2.1 Inclusion process

RT:03.12.0002.1 It is RECOMMENDED to use physical push buttons for activating Learn Mode and Add Mode on a PC Role Type.

8.4.3.2.2 Lifeline configuration

No requirement.

8.4.3.2.3 PC including a CSC

RT:03.11.0007.1 If the PC is the Primary Controller and a CSC is added to the network, the PC MUST assign the SIS role to the CSC.

RT:03.11.0008.1 The PC becomes an inclusion controller and MUST support the following network functions:

- Include new nodes ("Add mode")
- Exclude nodes
- Learn mode

Battery considerations

None.

8.4.3.2.4 PC including an RPC, PEN, or RSEN

Battery considerations

None.

8.4.3.2.5 PC including an SSC, PC, AOEN, LSEN or NAEN

Battery considerations

None.

8.4.3.3 PC Runtime Configuration

No requirements.

8.4.3.4 PC Runtime communication

No requirements.

8.4.4 Reporting Portable Controller (RPC)

The Reporting Portable Controller Role Type is intended for portable reporting controllers, which need to setup a Z-Wave network and also send unsolicited messages.

The RPC Role Type may for instance be used for a battery powered thermostat which can include and exclude nodes in a small network. In addition, the thermostat may be configured remotely.

8.4.4.1 RPC protocol requirements

RT:04.11.0001.1 The RPC MUST respect requirements described in Section 8.2.

RT:04.11.0002.1 The RPC MUST be battery powered and support the following Command Classes: * Battery Command Class * Wake Up Command Class, version 2 or newer

RT:04.11.0003.1 The RPC MUST set the listening flag to 0 in its NIF.

RT:04.12.0001.1 The RPC SHOULD NOT configure lifeline associations when adding nodes to the network.

RT:04.11.0004.1 The RPC MUST support the following network roles:

- Primary controller
- Secondary controller
- Inclusion controller

8.4.4.1.1 If first node in the network

RT:04.11.0005.1 If the RPC is the first node in the network, it MUST take the Primary Controller role and MUST support the following network functions: * Include new nodes (“Add mode”) * Exclude nodes * Learn mode

RT:04.12.0006.1 It MUST NOT be possible to activate Learn Mode if the RPC is the Primary Controller and other nodes are included in the network.

8.4.4.2 RPC Setup

8.4.4.2.1 Inclusion process

RT:04.12.0002.1 It is RECOMMENDED to use physical push buttons for activating learn mode and add mode on an RPC Role Type.

8.4.4.2.2 Lifeline configuration

RT:04.13.0001.1 When including a node, if a SIS is present in the network, the RPC MAY set the SIS’ NodeID as the Association group ID 1 (Lifeline) destination.

Details and requirements about establishing the Lifeline are provided in the Association and Multi Channel Association control specifications *Command Class Control*.

8.4.4.2.3 RPC including a CSC

RT:04.11.0007.1 If the RPC is the Primary Controller and a CSC is added to the network, the RPC MUST assign the SIS role to the CSC.

RT:04.11.0008.1 The RPC becomes an inclusion controller and MUST support the following network functions: * Include new nodes ("Add mode") * Exclude nodes * Learn mode

Battery considerations None

8.4.4.2.4 RPC including an RPC, PEN or RSEN

Battery considerations None

8.4.4.2.5 RPC including an SSC, PC, AOEN, LSEN or NAEN

Battery considerations None

8.4.4.3 RPC runtime configuration

RT:04.11.0009.1 The RPC MUST support the Wake Up Command Class as described in 0.

RT:04.12.0005.1 The RPC SHOULD have a physical push button for waking up the device for expedited communication. This enables interactive delivery of new configuration parameters or firmware updates.

RT:04.11.000A.1 The RPC MUST implement a Minimum Wake Up Interval in the range 0 ..4200 (i.e. between 0 second and 70 minutes).

RT:04.11.000B.1 If the RPC's Minimum Wake Up Interval is 0, the RPC MUST implement a Maximum Wake Up Interval greater than 0.

8.4.4.4 RPC runtime communication

RT:04.11.000C.1 The RPC MUST communicate via the lifeline association if any lifeline association exists. Refer to [34] for more details.

8.4.4.4.1 Portable End Node (PEN)

The Portable End Node Role Type is intended for battery powered devices that aim for the lowest possible power consumption. The PEN only wakes up in response to a physical event such as a button press. The PEN allows for optimal cost, as no EEPROM is required.

8.4.5 Portable End Node (PEN)

8.4.5.1 PEN protocol requirements

RT:05.11.0001.1 The PEN MUST respect requirements described in Section 8.2.

RT:05.11.0002.1 The PEN MUST be battery powered and support the following Command Classes:

- Battery Command Class
- Wake Up Command Class, version 2 or newer

RT:05.11.0003.1 The PEN MUST set the listening flag to 0 in its NIF.

The PEN can only be added to a network and has no network role requirement.

8.4.5.2 PEN setup

The setups of a PEN by a CSC, SSC, PC or RPC are respectively described in Section 8.4.1.2.4, Section 8.4.2.2.4, Section 8.4.3.2.4 or Section 8.4.4.2.5. The PEN has no additional requirement when being included.

8.4.5.2.1 Inclusion process

RT:05.12.0001.1 It is RECOMMENDED to use a physical push button for activating learn mode on a PEN Role Type.

8.4.5.3 PEN Runtime configuration

RT:05.11.0004.1 The PEN MUST support the Wake Up Command Class as described in Section 8.2.8.2.

RT:05.12.0002.1 The PEN SHOULD use a default Wake-Up interval of 0.

RT:05.12.0003.1 The PEN SHOULD have a physical push button for waking up the device for expedited communication. This enables interactive delivery of new configuration parameters or firmware updates.

8.4.5.4 PEN Runtime communication

RT:05.11.0005.1 The PEN MUST communicate via the lifeline association if any lifeline association exists. Refer to [34] for more details.

8.4.6 Always On End Node (AOEN)

The Always On End Node Role Type is intended for mains powered devices that are always reachable. One example of such a device is a light switch.

8.4.6.1 AOEN protocol requirements

RT:06.11.0001.1 The AOEN MUST respect requirements described in Section 8.2.

RT:06.11.0002.1 The AOEN MUST be mains powered and MAY have a battery back-up.

RT:06.11.0003.1 The AOEN MUST set the listening flag to 1 in its NIF. The AOEN can only be added to a network and has no network role requirement.

8.4.6.2 AOEN setup

The setups of an AOEN by a CSC, SSC, PC or RPC are respectively described in Section 8.4.1.2.5, Section 8.4.2.2.5, Section 8.4.3.2.5 or Section 8.4.4.2.5. The AOEN has no additional requirement when being included.

8.4.6.2.1 Inclusion process

RT:06.12.0001.1 It is RECOMMENDED to use a physical push button for activating learn mode on an AOEN Role Type.

8.4.6.3 AOEN runtime configuration

AOEN can always be configured, as it is always listening.

8.4.6.4 AOEN runtime communication

RT:06.11.0004.1 The AOEN MUST communicate via the lifeline association if any lifeline association exists. Refer to [34] for more details.

8.4.7 Reporting Sleeping End Node (RSEN)

The Reporting Sleeping End Node Role Type is intended for battery-powered devices that only wake up and communicates when an event has occurred. This allows to reconfigure the device remotely. Examples include sensors, wall controllers etc.

8.4.7.1 RSEN protocol requirements

RT:07.11.0001.1 The RSEN MUST respect requirements described in Section 8.2.

RT:07.11.0002.1 The RSEN MUST be battery powered and support the following Command Classes: * Battery Command Class * Wake Up Command Class, version 2 or newer

RT:07.11.0003.1 The RSEN MUST set the listening flag to 0 in its NIF.

The RSEN can only be added to a network and has no network role requirement.

8.4.7.2 RSEN setup

The setups of an RSEN by a CSC, SSC, PC or RPC are respectively described in Section 8.4.1.2.4, Section 8.4.2.2.4, Section 8.4.3.2.4 or Section 8.4.4.2.4.

The RSEN has no additional requirement when being included.

8.4.7.2.1 Inclusion process

RT:07.12.0001.1 It is RECOMMENDED to use a physical push button for activating learn mode on an RSEN Role Type.

8.4.7.2.2 RSEN runtime configuration

RT:07.11.0004.1 The RSEN MUST support the Wake Up Command Class as described in Section 8.2.8.2 and in Figure 8.12.

RT:07.12.0002.1 The device SHOULD have a physical push button for waking up the device for expedited communication. This enables interactive delivery of new configuration parameters or firmware updates.

RT:07.11.0005.1 The RSEN MUST implement a Minimum Wake Up Interval in the range 0 .. 4200 (i.e. between 0 second and 70 minutes).

RT:07.11.0006.1 If the RSEN's Minimum Wake Up Interval is 0, the RSEN MUST implement a Maximum Wake Up Interval greater than 0.

8.4.7.2.3 RSEN runtime communication

RT:07.11.0007.1 The RSEN MUST communicate via the lifeline association if any lifeline association exists. Refer to [34] for details.

8.4.8 Listening Sleeping End Node (LSEN)

The Listening Sleeping End Node Role Type is intended for battery-operated devices that can be reached even though they are sleeping thanks to Beaming (FL nodes). Examples include Door Locks and Battery operated Thermostats.

8.4.8.1 LSEN Protocol Requirements

RT:08.11.0001.1 The LSEN MUST respect requirements described in Section 8.2.

RT:08.11.0002.1 The LSEN MUST be battery powered and support the Battery Command Class.

RT:08.11.0003.1 The LSEN MUST set the listening flag to 0 in its NIF.

The LSEN can only be added to a network and has no network role requirement.

8.4.8.2 LSEN Setup

The setups of an LSEN by a CSC, SSC, PC or RPC are respectively described in Section 8.4.1.2.4, Section 8.4.2.2.4, Section 8.4.3.2.4 or Section 8.4.4.2.4. The LSEN has no additional requirement when being included.

8.4.8.2.1 Inclusion Process

RT:08.12.0001.1 It is RECOMMENDED to use a physical push button for activating learn mode on a LSEN Role Type.

8.4.8.3 LSEN Runtime Configuration

A LSEN can always be configured, as it is reachable via FLiRS communication.

8.4.8.4 LSEN Runtime Communication

RT:08.11.0004.1 The LSEN MUST communicate via the lifeline association if any lifeline association exists. Refer to [34] for details.

RT:08.11.0005.1 The LSEN MUST stay awake for at least 2 seconds after communicating.

8.4.9 Network Aware End Node (NAEN)

The Network Aware End Node Role Type is intended for end nodes with application controlling capabilities, which are leveraging controller functionalities to be aware of the network topology and nodes capabilities.

The SIS (or primary controller) will consider a NAEN as a controller, but the NAEN will not be able to include new nodes in the network.

8.4.9.1 NAEN Protocol Requirements

RT:09.11.0001.1 The NAEN MUST respect requirements described in Section 8.2.

RT:09.11.0002.1 The NAEN MUST be mains powered and MAY have a battery back-up.

RT:09.11.0003.1 The NAEN MUST set the listening flag to 1 in its NIF.

RT:09.11.0006.1 The NAEN can only be added to a network and MUST take the inclusion controller or the secondary controller role when added to a network.

RT:09.11.0004.1 The NAEN MUST NOT provide the following network functions:

- Include new nodes
- Exclude nodes
- Remove failing node
- Replace failing node

8.4.9.2 NAEN Setup

The setups of an NAEN by a CSC, SSC, PC or RPC are respectively described in Section 8.4.1.2.3, Section 8.4.2.2.5, Section 8.4.3.2.5 or Section 8.4.4.2.5. The NAEN has no additional requirement when being included.

8.4.9.2.1 Inclusion process

RT:09.12.0001.1 It is RECOMMENDED to use a physical push button for activating learn mode on an NAEN Role Type.

8.4.9.3 NAEN Runtime Configuration

The NAEN can always be configured, as it is always listening.

8.4.9.4 NAEN Runtime communication

RT:09.11.0005.1 The NAEN MUST communicate via the lifeline association if any lifeline association exists. Refer to [34] for more details.

9 Appendices

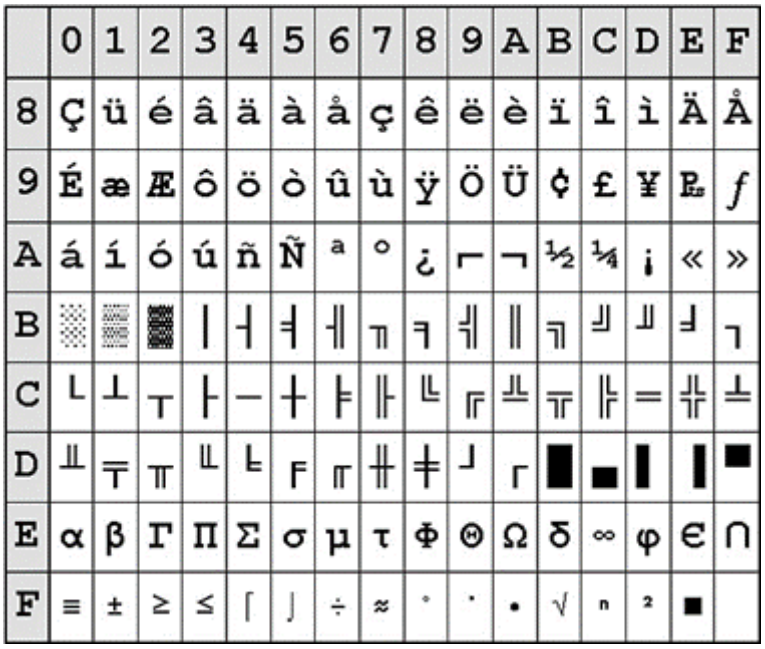
9.1 ASCII Codes

The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes (non-printable), and the remaining 96 character codes are printable characters. Figure 9.1 shows the hexadecimal values of the ASCII character codes, e.g. the ASCII code for the capital letter "A" is equal to 0x41:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	U
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

Figure 9.1: The Standard ASCII Table

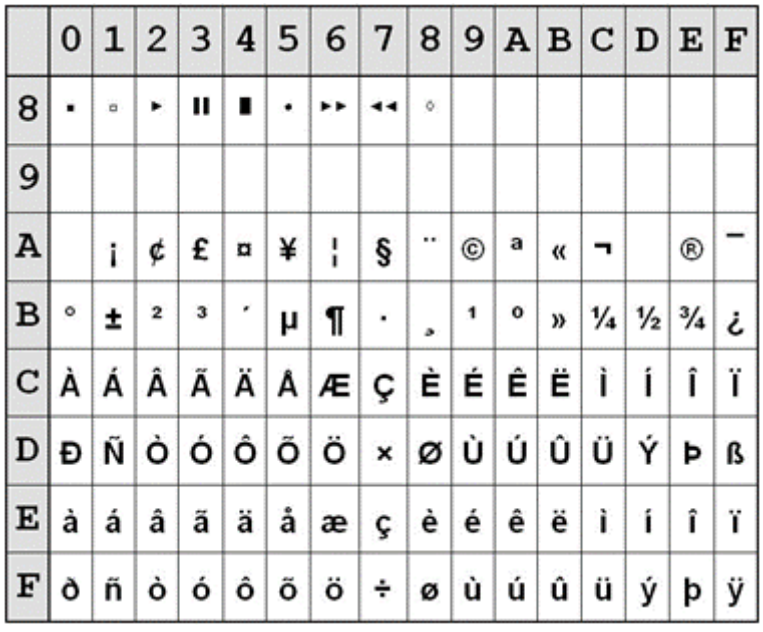
In addition to the 128 standard ASCII codes (the ones listed above ranging from 0 to 127), most systems have another 128 extra codes which form what is known as extended ASCII (with ranges from 128 to 255). The OEM Extended ASCII character set is included in all PC-compatible computers as the default character set when the system boots before loading any operating system and under MS-DOS. It includes some foreign signs, some marked characters and also pieces to draw simple panels. Figure 9.2 shows the hexadecimal values of the OEM Extended ASCII character codes, e.g. the ASCII code for the capital letter "Æ" is equal to 0x92:



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
9	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	₧	ƒ
A	á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐	¬	½	¼	¡	«	»
B	▒	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	▉	▄	▌	▐	▀
E	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	φ	∈	∩
F	≡	±	≥	≤	⌠	⌡	÷	≈	°	∙	·	√	ⁿ	²	■	

Figure 9.2: OEM Extended ASCII Table

Below are listed codes for players, radios etc. as an alternative to the OEM Extended ASCII codes. Undefined values MUST be ignored.



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	▪	▫	▸	⏸	▮	•	▸▸	◂◂	◊							
9																
A		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
B	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Figure 9.3: Players Table

9.2 CRC-CCITT Source Code

The checksum algorithm implements a CRC-CCITT using initialization values equal to 0x1D0F and 0x1021 (normal representation) as the poly.

9.2.1 Header file

```
/**
 * @file
 * Functions for calculation of CRC.
 * @copyright 2018 Silicon Laboratories Inc.
 */
#ifndef _CRC_H_
#define _CRC_H_

/****************************************************************************/
/*                              INCLUDE FILES                               */
/****************************************************************************/

#include <stdint.h>

/****************************************************************************/
/*                     EXPORTED TYPES and DEFINITIONS                       */
/****************************************************************************/

/****************************************************************************/
/*                              EXPORTED DATA                               */
/****************************************************************************/

/****************************************************************************/
/*                            EXPORTED FUNCTIONS                            */
/****************************************************************************/

#define CRC_INITAL_VALUE                    0x1D0Fu

/**
 * Returns a CRC calculation using the CCITT polynomial (0x1021).
 *
 * @param crc Initial value set to CRC_INITAL_VALUE unless calculating multiple parts of a␣
↪frame. In that case
 *            the value should be set to the result of the previous calculation.
 * @param pDataAddr Pointer to the array of data.
 * @param bDataLen Length of the data.
 * @return CRC value
 */
uint16_t CRC_CheckCrc16(
  uint16_t crc,
  uint8_t *pDataAddr,
  uint16_t bDataLen
);

#endif /* _CRC_H_ */
```


9.2.2 Implementation

```c
/**
 * @file
 * Functions for calculation of CRC.
 * @copyright 2018 Silicon Laboratories Inc.
 */
#include <CRC.h>

#define POLY 0x1021           /* crc-ccitt mask */

uint16_t CRC_CheckCrc16(
  uint16_t crc,
  uint8_t *pDataAddr,
  uint16_t bDataLen)
{
  uint8_t WorkData;
  uint8_t bitMask;
  uint8_t NewBit;

  while(bDataLen--)
  {
    WorkData = *pDataAddr;
    pDataAddr++;
    for (bitMask = 0x80; bitMask != 0; bitMask >>= 1)
    {
      /* Align test bit with next bit of the message byte, starting with msb. */
      NewBit = ((WorkData & bitMask) != 0) ^ ((crc & 0x8000) != 0);
      crc <<= 1;
      if (NewBit)
      {
        crc ^= POLY;
      }
    } /* for (bitMask = 0x80; bitMask != 0; bitMask >>= 1) */
  }
  return crc;
}
```


9.3 Inclusion Process

This section outlines the recommended inclusion process that all Role Types should follow.

The processes for both node including and being included are covered.

9.3.1 Being Included

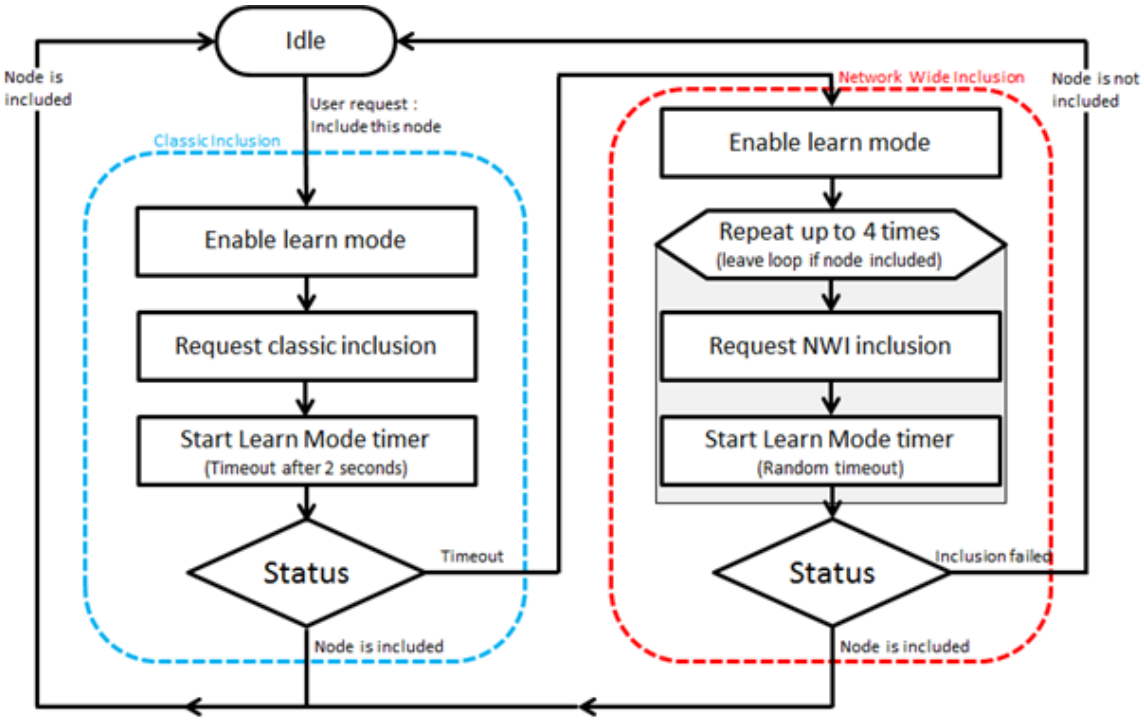


Figure 9.4: Inclusion Process for the Node being Included

9.3.2 Including a Node

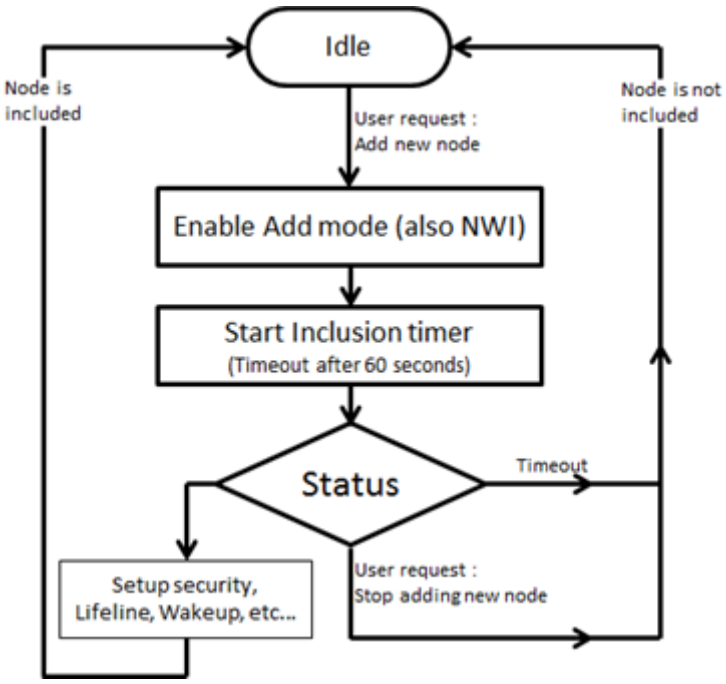


Figure 9.5: Inclusion Process for the Including Node

References

[1] D. J. Bernstein. A state-of-the-art Diffie-Hellman function. 2022. URL: http://cr.yp.to/ecdh.html.

[2] Barak Boaz and Shai Halevi. A Model and Architecture for Pseudo-Random Generation with Applications to /dev/random. In *Proceedings of the 12th ACM Conference on Computer and Communications Security*, CCS 2005, 203–212. New York, NY, USA, November 2005. Association for Computing Machinery. doi:10.1145/1102120.1102148.

[3] Scott O. Bradner. Key words for use in RFCs to Indicate Requirement Levels. RFC 2119, March 1997. URL: https://www.rfc-editor.org/info/rfc2119, doi:10.17487/RFC2119.

[4] Anders Brandt and Jakob Buron. Transmission of IPv6 Packets over ITU-T G.9959 Networks. RFC 7428, February 2015. URL: https://www.rfc-editor.org/info/rfc7428, doi:10.17487/RFC7428.

[5] Alex Conta. Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification. RFC 3122, June 2001. URL: https://www.rfc-editor.org/info/rfc3122, doi:10.17487/RFC3122.

[6] Dr. Steve E. Deering and Bob Hinden. IP Version 6 Addressing Architecture. RFC 4291, February 2006. URL: https://www.rfc-editor.org/info/rfc4291, doi:10.17487/RFC4291.

[7] Niels Ferguson, Bruce Schneier, and Tadayoshi Kohno. *Cryptography Engineering: Design Principles and Practical Applications*. Wiley Publishing, Inc., Indianapolis, IN, 2010. ISBN 9780470474242.

[8] Bob Hinden and Dr. Steve E. Deering. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, December 1998. URL: https://www.rfc-editor.org/info/rfc2460, doi:10.17487/RFC2460.

[9] S. Matyas, C. Meyer, and J. Oseas. Generating strong one-way functions with cryptographic algorithm. *IBM Technical Disclosure Bulletin*, 27:5658–5695, 1985.

[10] Robert Moskowitz and Rene Hummen. HIP Diet EXchange (DEX). Internet Draft draft-moskowitz-hip-dex-02, Internet Engineering Task Force (IETF), June 2005. Work in Progress. URL: https://datatracker.ietf.org/doc/html/draft-moskowitz-hip-dex-02.

[11] National Institute of Standards and Technology (NIST). Advanced Encryption Standard (AES). Technical Report Federal Information Procesing Standards Publications (FIPS PUBS) 197, U.S. Department of Commerce, Washington, D.C., November 2001. doi:10.6028/NIST.FIPS.197.

[12] National Institute of Standards and Technology (NIST). Recommendation for Block Cipher Modes of Operation: Methods and Techniques. Technical Report Special Publication (NIST SP) 800-38A, U.S. Department of Commerce, Washington, D.C., January 2001. doi:10.6028/NIST.SP.800-38A.

[13] National Institute of Standards and Technology (NIST). Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. Technical Report Special Publication (NIST SP) 800-38C, U.S. Department of Commerce, Washington, D.C., November 2007. doi:10.6028/NIST.SP.800-38C.

[14] National Institute of Standards and Technology (NIST). Recommendation for Random Number Generation Using Deterministic Random Bit Generators. Technical Report Special Publication (NIST SP) 800-90A, U.S. Department of Commerce, Washington, D.C., January 2012. doi:10.6028/NIST.SP.800-90A.

[15] National Institute of Standards and Technology (NIST). Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. Technical Report Special Publication (NIST SP) 800-38B, U.S. Department of Commerce, Washington, D.C., October 2016. doi:10.6028/NIST.SP.800-38B.

[16] William A. Simpson, Dr. Thomas Narten, Erik Nordmark, and Hesham Soliman. Neighbor Discovery for IP version 6 (IPv6). RFC 4861, September 2007. URL: https://www.rfc-editor.org/info/rfc4861, doi:10.17487/RFC4861.

[17] Doug Whiting, Russ Housley, and Niels Ferguson. Counter with CBC-MAC (CCM). RFC 3610, September 2003. URL: https://www.rfc-editor.org/info/rfc3610, doi:10.17487/RFC3610.

[18] Biometrics Institute. Types of Biometrics. [Online]. URL: https://www.biometricsinstitute.org/what-is-biometrics/types-of-biometrics/.

[19] Internet Assigned Numbers Authority (IANA). Service Name and Transport Protocol Port Number Registry. April 2022. [Online]. URL: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.

[20] ITU Telecommunication Standardization Sector (ITU-T). Short range narrow-band digital radiocommunication transceivers - PHY, MAC, SAR and LLC layer specifications. January 2015. [Online]. URL: https://www.itu.int/rec/T-REC-G.9959-201501-I.

[21] Wikipedia contributors. List of graphical user interface elements — Wikipedia, the free encyclopedia. 2022. [Online]. URL: https://en.wikipedia.org/w/index.php?title=List_of_graphical_user_interface_elements&oldid=1080399297.

[22] Z-Wave Alliance, Inc. Anti-Theft Command Class, list of assigned Locking Entity IDs.

[23] Z-Wave Alliance, Inc. Association Command Class, list of mandatory Commands for the Lifeline Association Group. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/635.

[24] Z-Wave Alliance, Inc. Indicator Command Class, list of defined Indicator and Property IDs. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/633.

[25] Z-Wave Alliance, Inc. List of Defined Z-Wave Command Classes. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/823.

[26] Z-Wave Alliance, Inc. Meter Table Monitor Command Class, list of assigned Types, Scales and Datasets. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/650.

[27] Z-Wave Alliance, Inc. Multilevel Sensor Command Class, list of assigned Multilevel Sensor Types and Scales. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/640.

[28] Z-Wave Alliance, Inc. Node Provisioning Information Type Registry (QR Code, Z/IP Gateway, SmartStart. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/644.

[29] Z-Wave Alliance, Inc. Node Provisioning QR Code Format (S2, SmartStart. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/647.

[30] Z-Wave Alliance, Inc. Security 2 (S2) and SmartStart Z-Wave Product Labeling Requirements. September 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/162.

[31] Z-Wave Alliance, Inc. Simple AV Command Class, list of assigned AV Control Codes. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/648.

[32] Z-Wave Alliance, Inc. Z-Wave Assigned Manufacturer IDs. September 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/641.

[33] Z-Wave Alliance, Inc. Z-Wave Plus Assigned Icon Types. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/646.

[34] Z-Wave Alliance, Inc. Z-Wave Plus Device Type Specification. October 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/642.

[35] Z-Wave Alliance, Inc. Z-Wave and Z-Wave Long Range Network Layer Specification. August 2021. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/897.

[36] Z-Wave Alliance, Inc. Notification Command Class, list of assigned Notifications. February 2022. URL: https://sdomembers.z-wavealliance.org/wg/Members/document/639.

