Virtual WAN Interface Specification

Version 1.0

17 October 2025

Documentation Alignment

ZigBee Smart Energy Profile

This document aligns with:

- ZigBee Smart Energy (ZSE) Profile Specification 1.4;
- ZigBee Cluster Library Specification, reference 07-5123-04;
- ZigBee Specification Revision 22, reference 05-3474-22; and
- ZigBee PRO/2007 Layer PICS and Stack Profiles – 08-0006-05.

These documents are available from http://zigbee.org/About/GBCSPartner.aspx or by contacting the ZigBee Alliance.

Message Queue Telemetry Transport (MQTT)

This document aligns with the MQTT 3.1.1 Specification. This document can be obtained from https://mqtt.org/mqtt-specification.

Great Britain Companion Specification

This document aligns with the GBCS version 4.4 or later.

Contents

1 Introduction – informative

The purpose of the Virtual WAN (VWAN) Solution is to provide Smart Metering in areas with no reliable Smart Metering WAN (SMWAN) connectivity. This requires Devices with additional functionality:

- Virtual WAN Communications Hub (VWCH)
    - A Communications Hub (CH) variant that contains functionality to allow communication with a VWD.
- Virtual WAN Device (VWD)
    - A PPMID or Type 2 Device (IHD/CAD) that has internet connectivity and contains functionality to route Messages between the VWCH and the VWP.
    - A VWD connects to a VWCH using a ZSE tunnel
    - A VWD connects to the VWP using an MQTT connection via the Consumer's internet.

The VWAN Solution also requires the following service:

- Virtual WAN Provider (VWP)
    - A service residing within DCC systems that communicates with VWDs.
    - The VWP and VWD in combination will convey Messages usually conveyed over the Smart Metering Wide Area Network (SMWAN) using the internet and the Smart Metering Home Area Network (SMHAN).

This document primarily defines the communications protocol between the Virtual WAN Provider (VWP) and a Virtual WAN Device (VWD) and forms part of a set of overall specification documents that comprise the Virtual WAN (VWAN) Solution. This document provides requirements and recommendations for the following parties:

- the implementer of the VWP; and
- VWD Manufacturers

Each section will indicate to which party it is relevant.

There are places in this document where functional behaviour for the implementer of the VWP is described at a high level without any detail. In these cases, it will be noted that the detailed behaviour is described elsewhere.

2 Normative Requirements

Some Sections of this document are informative and others normative. Unless Sections are marked ‘informative’ in the header, they shall be normative. Subsections of Sections marked informative shall also be informative.

3 System overview – informative

3.1 Existing Smart Metering system

Under the current Smart Metering arrangements, a DCC User connects to the DSP in the manner set out in the DCC User Interface Specification (DUIS). The DSP validates each Service Request from the DCC User and ultimately identifies which Communication Service Provider (CSP) that the target CH is connected to. The CSPs provide a wireless SMWAN to transport Messages (both GBCS and Device Manager) to and from the CHs.

3.2 Virtual WAN Solution

As described in Section 1, under the Smart Metering arrangements enhanced with the VWAN Solution, a VWCH is used, which is additionally connected to the VWP with the assistance of a VWD. The path used to transport Messages is determined by evaluating the respective link statuses of the SMWAN and the VWAN.

In the Smart Metering system enhanced by the VWAN Solution, a DCC User connects to the DSP in the manner set out in DUIS in the same way. The DSP validates each Service Request from the DCC User and the active communication path (SMWAN or VWAN) is ultimately identified for the target VWCH. Figure 3.2 illustrates the Smart Metering system enhanced by the VWAN Solution.

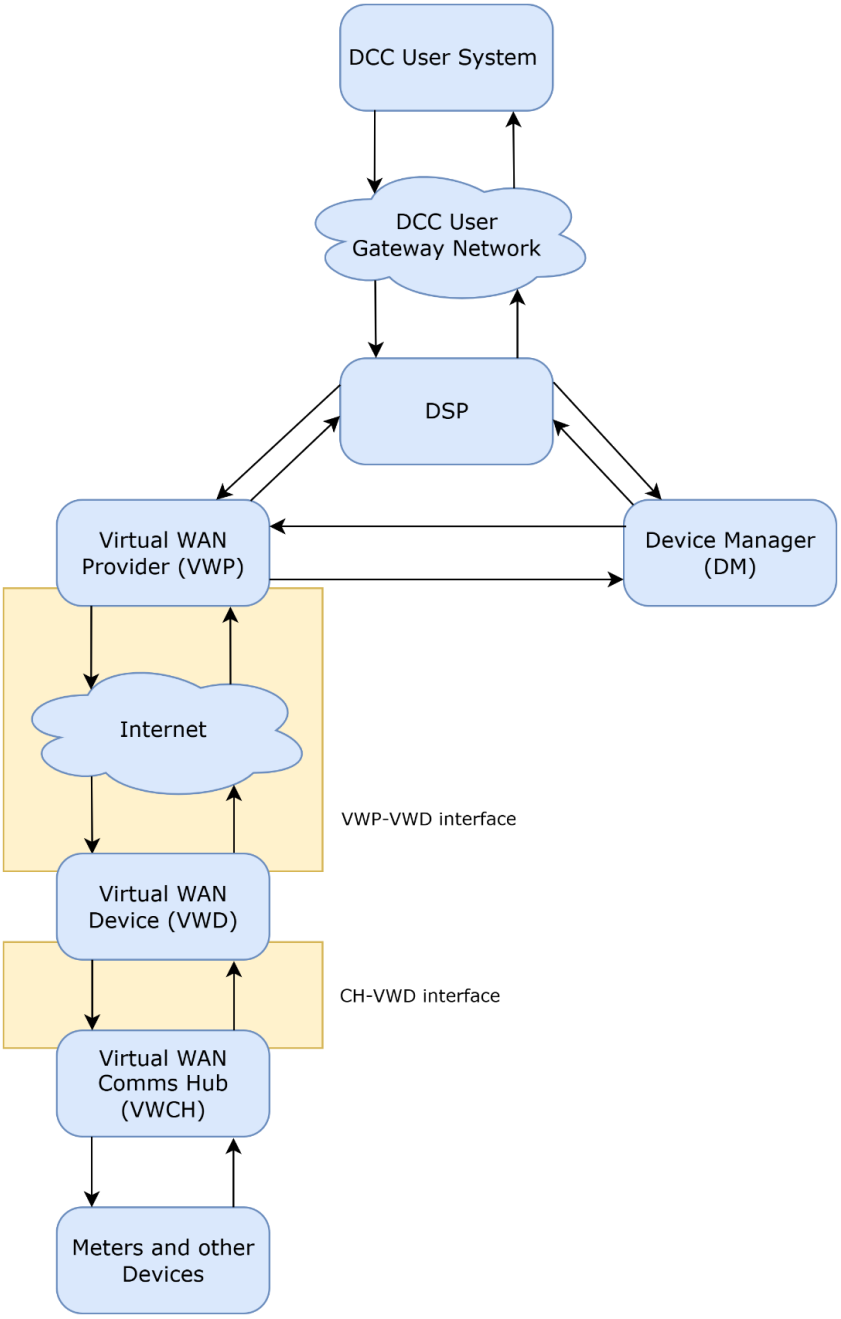

Figure: 3.2

3.3 Communications between VWCH and VWP

The VWCH has two physical network interfaces:

- a 4G cellular network interface for connecting to the SMWAN; and
- a Zigbee network interface to connect to other HAN Devices over the SMHAN.

Normally, a CH communicates with DCC systems via the SMWAN. If SMWAN connectivity is not established or subsequently fails, the VWAN Solution provides a mechanism for establishing a 'virtual' WAN connection from a VWCH to DCC systems via an internet-connected VWD and the VWP, where:

- communication between the VWCH and the VWD is provided by the VWD joining the SMHAN and using a Zigbee tunnel; and

- communication between the VWD and the VWP is provided by the VWD connecting to the VWP over the internet via consumer broadband, using JSON messages over the MQTT protocol secured by mutually-authenticated TLS (v1.3 or v1.2).

Due to the use of two different types of networking protocols between VWCH and the VWP, a single end-to-end transport and security model is not achievable. To overcome this, a DLMS-based authenticated encryption model is adopted for all communication between the VWCH and the VWP. This is illustrated in Figure 3.3.

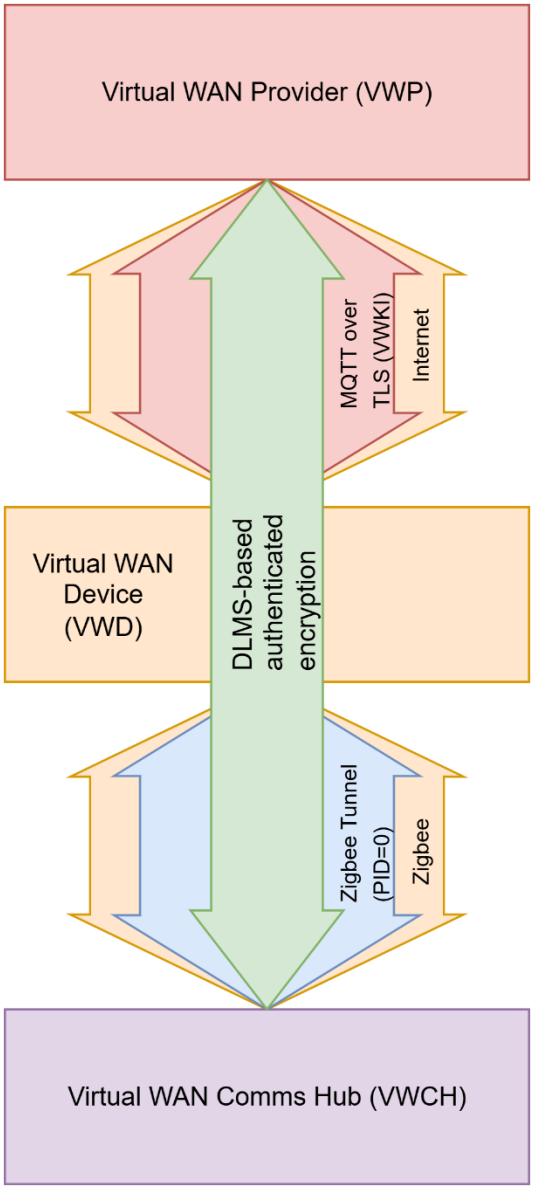

Figure: 3.3

Once VWAN functionality is enabled:

- All Messages exchanged between the VWP and the VWCH via the VWD are encrypted and integrity protected.
- For Messages from VWP to the VWCH, the VWD receives the encrypted and integrity protected DLMS content in a JSON wrapper over MQTT and sends the payload content to the VWCH over the VWAN Tunnel (see Section 3.5).

- For Messages from the VWCH to the VWP, the VWD receives the encrypted and integrity protected DLMS content from the VWAN Tunnel and sends it to the VWP in a JSON wrapper via MQTT.

The VWAN Solution will use the same message acknowledgement that already exists for the SMWAN for Messages sent from the VWCH to the VWP, i.e. each Message sent from the VWCH shall be acknowledged by the VWP using NEWAN ACK. The VWCH does not send an equivalent ACK for Messages from the VWP to the VWCH, but higher level retry functionality exists for these Messages.

3.4 Adding VWD to the SMHAN

A VWD can be added to the SMHAN in two ways:

- via the SMWAN; or
- via the Virtual WAN

3.4.1 Adding via SMWAN

If the VWCH is communicating via the SMWAN, a VWD can be added to the CHF Device Log of the VWCH by a DCC User issuing a SRV 8.11 (Update HAN Device Log – add) as IHD, CAD or PPMID Device type. This allows the VWD to Zigbee-join the SMHAN.

VWAN mode is enabled as described in Section 3.5; in this case, the EnableVWD Command is sent to the VWCH via the SMWAN.

3.4.2 Adding via the Virtual WAN

Where there is no SMWAN available, a VWD uses HHT functionality to Zigbee-join to the SMHAN. The VWD establishes an Enhanced Inter-PAN connection with the VWCH and requests a CCS01 Command from DCC systems. The CCS01 Command is then sent to the VWCH over the Enhanced Inter-PAN connection, which adds the VWD to the CHF Device Log of the VWCH. This allows the VWD to Zigbee-join the SMHAN.

VWAN mode is enabled as described in Section 3.5; in this case, the EnableVWD Command is requested and sent to the VWCH via the VWAN.

3.5 VWAN mode

To operate in VWAN mode, an EnableVWD Command is sent to the VWCH, which enables VWAN mode on the VWCH. The VWCH will then send a RequestTunnel ZSE command to the VWD specifying Protocol ID 0 to create the VWAN Tunnel (see Section 4.2.2). When the VWD has successfully processed the RequestTunnel ZSE command, it is VWAN mode.

3.6 Subsequent operation

Once the VWD has Zigbee-joined the SMHAN and has a viable WAN route (either SMWAN or Virtual WAN), the usual additional steps to fully commission the VWD may be required, for example, SRV 8.7.1 / 8.7.2 would be required to complete joining to other SMHAN Devices.

Note that once the VWCH and VWD are in VWAN mode and the SMWAN is active, there is only limited communication between the VWCH and the VWP primarily for maintenance purposes.

4 Zigbee interface

The basic Zigbee configuration for the VWD is shown in Table 4.

Property	Value
Zigbee frequency	2.4 GHz (868 MHz is not supported for VWD communication due to duty cycle concerns)
Zigbee device type	Zigbee Router or Zigbee End Device (RxOnWhenIdle = TRUE)

Table: 4

All data sent and received by the VWD is secured using Zigbee APS authenticated encryption based on Zigbee Certificate-based Key Establishment (CBKE) as described in GBCS.

4.1 Operating as an HHT

The VWD shall be able to use joining using Enhanced Inter-PAN (hereafter known as “inter-PAN join”) as described in GBCS section 10.9 to connect to the VWCH. If the VWD fails to inter-PAN join to the VWCH, the VWD should implement an automatic retry strategy.

If the VWD fails to inter-PAN join to the VWCH, it should notify the VWD user. The method used to notify the VWD user is not specified in this document.

4.2 VWD tunnel handling

The VWD and VWCH use up to two tunnels to communicate:

- a tunnel configured with Protocol ID 6; and
- a tunnel configured with Protocol ID 0

Both tunnels shall use Zigbee APS ACK (Application Support Sublayer Acknowledgement) as defined in the Zigbee Specification to ensure messages are delivered reliably.

The VWD and VWCH shall support up to 10 messages queued per tunnel. These are typically configuration parameters for the Zigbee stack, e.g.

```
FRAGMENTATION_MAX_INCOMING_PACKETS=10,
FRAGMENTATION_MAX_OUTCOMING_PACKETS=10,
TUNNELING_CLIENT_MAXIMUM_INCOMING_TRANSFER_SIZE=1500,
FRAGMENTATION_BUFFER_SIZE=1500
```

4.2.1 Tunnel configured with Protocol ID 6

A tunnel configured with Protocol ID 6 has up to two mutually exclusive distinct purposes:

- An HHT Tunnel opened after inter-PAN joining to convey Messages to and from the VWCH that is being used to convey Messages to and from the VWD is described in GBCS Section 10.5.3.4 when it is operating in HHT mode.
    - The VWD is operating in HHT mode after it has successfully inter-PAN joined to a VWCH and subsequently Zigbee-joined the SMHAN.
- A PPMID Tunnel that is being used to convey PPMID-GSME messages to and from the VWD when it is not operating in HHT mode.
    - Only applicable when the VWD is based on a PPMID. PPMID Tunnel usage is not described in this document.

- A VWD shall leave HHT mode (and enter VWAN mode) after it has successfully processed a RequestTunnel ZSE command from the VWCH for a tunnel configured with Protocol ID 6.
- Any HHT Tunnel opened after inter-PAN joining shall be converted to a PPMID tunnel when the VWD is enabled in VWAN mode.
- Note that a VWD that Zigbee-joins via SMWAN is never in HHT mode, therefore any tunnel configured with Protocol ID 6 is always a PPMID Tunnel

4.2.2 Tunnel configured with Protocol ID 0

A tunnel configured with Protocol ID 0 has a single purpose:

- The VWAN Tunnel to convey Messages between the VWCH and the VWP.

The VWAN Tunnel is initially used to convey Messages that perform a key agreement process between the VWCH and the VWP to establish a shared secret that is the basis for encryption and integrity protection of all subsequent Messages being relayed by the VWD using DLMS general-glo-ciphering authenticated encryption. This provides additional security over and above the protection already present on the Messages and ensures the VWD cannot eavesdrop on or manipulate any of the relayed Messages. The key agreement process and additional security is not described in this document.

5 MQTT interface

5.1 Introduction – informative

The connection between a VWD and the VWP uses the MQTT protocol. The VWP acts as MQTT Server (Broker) and a VWD acts as MQTT Client. The MQTT topology is a many-to-one configuration, where the MQTT Server handles point-to-point MQTT connections with many MQTT Clients. The MQTT Server does not typically act as a broker in this configuration, as a MQTT Client publishes and subscribes to its own set of topics and has no involvement with any other MQTT Client.

5.2 MQTT Server configuration

This section is only relevant to the implementer of the VWP.

The VWP shall be the MQTT Server.

5.2.1 TLS configuration

The MQTT Server shall use client Certificate authentication.

During the TLS handshake, the MQTT Server shall transfer its end-entity Certificate (VWP Certificate) alone.

On initial connection, the VWD Certificate and identity presented by the MQTT Client shall be validated against the Smart Metering Inventory. The validation method used by the VWP is not described in this document.

5.2.2 Access Control List (ACL) configuration

A VWD as MQTT Client shall only be able to subscribe to topics relevant to itself that are published on by the MQTT Server and shall not be able to subscribe to topics that are published on by other VWDs. Topics for a particular VWD are identified by the VWD's EUI64 in the topic name, which allows that VWD to subscribe to topics only relevant to itself. This enables ACLs to be set up that restrict traffic to between the MQTT Server and the MQTT Client alone.

A VWD as MQTT Client may use its EUI64 as its single username. This allows an ACL entry based on username for that VWD to be created. For example, for a VWD whose EUI64 is "acde4800feedc0de", an ACL entry for a Mosquitto MQTT Server would be as follows:

```
user acde4800feedc0de
pattern read sb/%u/install
pattern read sb/%u/message/high
pattern read sb/%u/message/medium
pattern read sb/%u/message/low
pattern read sb/%u/error

pattern write nb/%u/message
pattern write nb/%u/control
pattern write nb/%u/logs
pattern write nb/%u/error
```

5.2.3 Additional settings

5.2.3.1 Quality of Service (QoS)

The MQTT Server shall use QoS level of 0.

5.3 MQTT Client configuration

This section is only relevant to a VWP Manufacturer.

A VWD shall be an MQTT Client.

5.3.1 TLS configuration

The MQTT Client shall use server Certificate authentication. The MQTT Client shall perform certification path validation using the Root VWKICA Certificate and Issuing VWP CA Certificate.

During the TLS handshake, the MQTT Client shall transfer its end-entity Certificate (VWD Certificate) alone.

On initial connection, VWD Certificate validation at the VWP may result in TLS timeout failure. Therefore:

- the MQTT Client should implement an automatic reconnect strategy when it is disconnected due to TLS error; and
- the MQTT Client should implement a back-off strategy (that is increasing the time between the retries) to avoid unnecessary network traffic.

If the VWD fails to connect to the MQTT Server, it should notify the VWD user. The method used to notify the VWD user is not specified in this document.

5.3.2 FQDN

The VWD shall be pre-configured with the FQDN of the VWP MQTT Server address. The DCC uses a custom domain name for the FQDN.

5.3.3 SNI

The Server Name Indication (SNI) TLS extension shall be used to connect to the VWP MQTT Server.

5.3.4 Persistent certificate storage

The VWD shall store the following data persistently:

- Root VWKICA Certificate
- Issuing VWP CA Certificate
- VWD Certificate

The VWD shall store the following data persistently and securely:

- VWD private key

5.4 Protocol level

The MQTT Server and MQTT Client shall use the MQTT protocol with QoS=1 for all MQTT messages except MQTT LWT. The QoS level 1 provides “at least once” message transmission assurance, and this is achieved as follows:

- The message publisher sends a message and expects a PUBACK from the message subscriber.
- If PUBACK is not received, the publisher retries sending the message until it gets an acknowledgment.
- This ensures the message is delivered at least once, but duplicates may occur.

5.5 VWP to VWD JSON messages

This section defines the JSON messages sent from the VWP to the VWD using the MQTT protocol. The schema is defined in Section 7.2. These are generally known as “southbound” messages.

All members are mandatory unless explicitly stated as being optional.

5.5.1 MQTT Topics

In the following topic descriptions, `<VWD EUI64>` refers to the EUI64 used by the VWD to communicate with the VWP. The EUI64 is the same value as used in the VWD Certificate subject field.

Topic	Purpose
`sb/<VWD EUI64>/install`	Messages from VWP during installation
`sb/<VWD EUI64>/message/high`	High priority messages to be routed to VWCH
`sb/<VWD EUI64>/message/medium`	Medium priority messages to be routed to VWCH
`sb/<VWD EUI64>/message/low`	Low priority messages to be routed to VWCH
`sb/<VWD EUI64>/error`	VWP error and exception messages

Table: 5.5.1

5.5.2 Allowlist request

5.5.2.1 VWP processing

This section is only relevant to the implementer of the VWP.

5.5.2.1.1 Trigger

VWP processing shall be triggered by the VWP receiving either:

- a message from DCC systems containing the CCS01 Command to add the VWD to the CHF Device Log of the VWCH; or
- an error message from DCC systems indicating that the VWD is not authorised.

The format of both messages is not described in this document.

5.5.2.1.2 Successful processing

The VWP shall construct an `allowlist_request` JSON message including the members shown in Table 5.5.2.1.2 and send it to the `sb/<VWD EUI64>/install` topic.

Member	Description
`response_code`	A code indicating successful authorisation of the VWD. The code shall assume one of the 2xx values in Table 6b.
`payload`	The base64-encoded CCS01 Command.

Table: 5.5.2.1.2

Example JSON message:

```
{
  "allowlist_request": {
```

```
    "response_code": "200",
    "payload": "ZGVmYXVsdA=="
  }
}
```

5.5.2.1.3 Error processing

The VWP shall construct an `allowlist_request` JSON message including the members shown in Table 5.5.2.1.3 and send it to the `sb/<VWD EUI64>/install` topic.

Member	Description
`response_code`	A code reflecting the error encountered during the VWD authorisation processing. The code shall assume one of the values in Table 6b. There will be no payload member.

Table: 5.5.2.1.3

Example JSON message:

```
{
  "allowlist_request": {
    "response_code": "501",
  }
}
```

5.5.2.2 VWD processing

This section is only relevant to a VWD Manufacturer.

5.5.2.2.1 Trigger

VWD processing shall be triggered by receipt of an `allowlist_request` JSON message.

5.5.2.2.2 Successful processing

On receipt of an `allowlist_request` JSON message with `response_code` indicating success, the JSON message will also include the base64-encoded CCS01 Command payload.

The VWD shall base64-decode the payload and send the decoded CCS01 Command to the VWCH using Enhanced Inter-PAN (see GBCS Section 10.5) for the purpose of adding the VWD to the VWCH's CHF Device Log.

5.5.2.2.3 Error processing

On receipt of an `allowlist_request` JSON message with `response_code` indicating an error, the JSON message will not include any payload and the VWD shall undertake no further processing. The VWD should notify the VWD user. The method used to notify the VWD user is not specified in this document.

Refer to Section 6 for error scenarios.

5.5.3 Allowlist response outcome

5.5.3.1 VWP processing

This section is only relevant to the implementer of the VWP.

5.5.3.1.1 Trigger

VWP processing shall be triggered by the VWP receiving the outcome of the CCS01 Response verification from DCC systems. The format of the message conveying the outcome is not described in this document.

5.5.3.1.2 Processing

The VWP shall construct an `allowlist_response_outcome` JSON message including the members shown in Table 5.5.3.1.2 and send it to the `sb/<VWD EUI64>/install` topic.

Member	Description
`response_code`	A code indicating the outcome of the CCS01 Response verification. The code shall assume one of the values in Table 6b.

Table: 5.5.3.1.2

Example JSON message:

```
{
  "allowlist_response_outcome": {
    "response_code": "200"
  }
}
```

5.5.3.2 VWD processing

This section is only relevant to a VWD Manufacturer.

5.5.3.2.1 Trigger

VWD processing shall be triggered by receipt of an `allowlist_response_outcome` JSON message.

5.5.3.2.2 Successful processing

On receipt of an `allowlist_response_outcome` JSON message with `response_code` indicating success, the VWD shall initiate Zigbee-joining with the VWCH (see GBCS section 10.9).

5.5.3.2.3 Error processing

On receipt of an `allowlist_response_outcome` JSON message with `response_code` indicating an error, the VWD shall undertake no further processing. The VWD should notify the VWD user and the VWD may implement an automatic retry strategy. The method used to notify the VWD user is not specified in this document and no retry strategy is specified in this document.

Refer to Section 6 for error scenarios.

5.5.4 EnableVWD request outcome

5.5.4.1 VWP processing

This section is only relevant to the implementer of the VWP.

5.5.4.1.1 Trigger

VWP processing shall be triggered by the VWP receiving either:

- a message from DCC systems containing the EnableVWD Command; or
- an error message indicating that the VWD is not authorised.

The format of either message is not described in this document.

5.5.4.1.2 Successful processing

On receipt of the EnableVWD Command, the VWP shall construct an `enablevwd_request_outcome` JSON message including the members shown in Table 5.5.4.1.2 and send it to the `sb/<VWD EUI64>/install` topic.

Member	Description
response_code	A code indicating successful CCS01 Response processing. The code shall assume one of the 2xx values in Table 6b.
payload	The base64-encoded EnableVWD Command.

Table: 5.5.4.1.2

Example JSON message:

```
{
  "enablevwd_request_outcome": {
    "response_code": "200",
    "payload": "ZGVmYXVsdA=="
  }
}
```

5.5.4.1.3 Error processing

On receipt of an error message indicating that the VWD is not authorised, the VWP shall construct an `enablevwd_request_outcome` JSON message including the members shown in Table 5.5.4.1.3 and send it to the `sb/<VWD EUI64>/install` topic.

Member	Description
response_code	A code reflecting the error encountered during the CCS01 Response verification processing. The code shall assume one of the values in Table 6b. There will be no payload member.

Table: 5.5.4.1.3

Example JSON message:

```
{
  "enablevwd_request_outcome": {
    "response_code": "501"
  }
}
```

5.5.4.2 VWD processing

This section is only relevant to a VWD Manufacturer.

5.5.4.2.1 Trigger

VWD processing shall be triggered by receipt of an `enablevwd_request_outcome` JSON message.

5.5.4.2.2 Successful processing

On receipt of an `enablevwd_request_outcome` JSON message with `response_code` indicating success, the JSON message also includes the base64-encoded EnableVWD Command payload.

The VWD shall base64-decode the payload and send the decoded EnableVWD Command to the VWCH using the HHT Tunnel (see GBCS section 10.9) for the purpose of enabling Virtual WAN mode between the VWD and the VWCH.

5.5.4.2.3 Error processing

On receipt of an `enablevwd_request_outcome` JSON message with `response_code` indicating an error, the JSON message will not include any payload and the VWD shall

undertake no further processing. The VWD should notify the VWD user and the VWD may implement an automatic retry strategy. The method used to notify the VWD user is not specified in this document and no retry strategy is specified in this document.

Refer to Section 6 for error scenarios.

5.5.5 EnableVWD response outcome

5.5.5.1 VWP processing

This section is only relevant to the implementer of the VWP.

5.5.5.1.1 Trigger

VWP processing shall be triggered by the VWP receiving the outcome of the EnableVWD Response verification from DCC systems. The format of the message conveying the outcome is not described in this document.

5.5.5.1.2 Processing

The VWP shall construct an `enablevwd_response_outcome` JSON message including the members shown in Table 5.5.5.1.2 and send it to the `sb/<VWD EUI64>/install` topic.

Member	Description
`response_code`	A code indicating the outcome of the EnableVWD Response verification. The code shall assume one of the values in Table 6b.

Table: 5.5.5.1.2

Example JSON message:

```
{
  "enablevwd_response_outcome": {
    "response_code": "200",
  }
}
```

5.5.5.2 VWD processing

This section is only relevant to a VWD Manufacturer.

5.5.5.2.1 Trigger

VWD processing shall be triggered by receipt of an `enablevwd_response_outcome` JSON message.

5.5.5.2.2 Successful processing

On receipt of an `enablevwd_response_outcome` JSON message with `response_code` indicating success, the VWD shall await the RequestTunnel ZSE command.

5.5.5.2.3 Error processing

On receipt of an `enablevwd_response_outcome` JSON message with `response_code` indicating an error, the VWD shall undertake no further processing. The VWD should notify the VWD user and the VWD may implement an automatic retry strategy. The method used to notify the VWD user is not specified in this document and no retry strategy is specified in this document.

Refer to Section 6 for error scenarios.

5.5.6 VWD key agreement outcome

5.5.6.1 VWP processing

This section is only relevant to the implementer of the VWP.

5.5.6.1.1 Trigger

VWP processing shall be triggered by the VWP receiving the outcome of the key agreement verification from DCC systems. The format of the message conveying the outcome is not described in this document.

5.5.6.1.2 Processing

The VWP shall construct a `vwd_ka_outcome` JSON message including the members shown in Table 5.5.6.1.2 and send it to the `sb/<VWD EUI64>/message/high` topic.

Member	Description
`response_code`	A code indicating the outcome of the key agreement verification. The code shall assume one of the values in Table 6b.

Table: 5.5.6.1.2

Example JSON message:

```
{
  "vwd_ka_outcome": {
    "response_code": "200"
  }
}
```

5.5.6.2 VWD processing

This section is only relevant to a VWD Manufacturer.

5.5.6.2.1 Trigger

VWD processing shall be triggered by receipt of a `vwd_ka_outcome` JSON message.

5.5.6.2.2 Successful processing

On receipt of a `vwd_ka_outcome` JSON message with `response_code` indicating success, the VWD may indicate the outcome to the VWD user for the purposes of informing the user that the VWD will start to relay Messages between the VWCH and the VWP.

5.5.6.2.3 Error processing

On receipt of a `vwd_ka_outcome` JSON message with `response_code` indicating an error, the VWD may indicate the outcome to the VWD user for the purposes of informing the user that there will be no Messages to be relayed between the VWCH and the VWP. The VWD should notify the VWD user and the VWD may implement an automatic retry strategy. The method used to notify the VWD user is not specified in this document and no retry strategy is specified in this document.

Refer to Section 6 for error scenarios.

5.5.7 Error reporting

5.5.7.1 VWP processing

This section is only relevant to the implementer of the VWP.

5.5.7.1.1 Trigger

VWP processing shall be triggered by the VWP encountering an error that it needs to convey to the VWD. The circumstances under which the VWP encounters an error are not described in this document.

5.5.7.1.2 Processing

The VWP shall construct an `error` JSON message including the members shown in Table 5.5.7.1.2 and send it to the `sb/<VWD EUI64>/error` topic:

Member	Description
`code`	The error code using an arbitrary string format.
`desc`	A description of the error using an arbitrary string format (maximum 255 characters).
`timestamp`	Timestamp in ISO 8601 format.

Table: 5.5.7.1.2

Example JSON message:

```
{
  "error": {
    "code": "nnnnnnnn",
    "desc": " xxxx....xxxx ",
    "timestamp": "2025-01-26T22:10:00Z"
  }
}
```

5.5.7.2 VWD processing

This section is only relevant to a VWD Manufacturer.

5.5.7.2.1 Trigger

VWD processing shall be triggered by receipt of an `error` JSON message.

5.5.7.2.2 Processing

On receipt of an `error` JSON message, the VWD may indicate the contents of the message to the VWD user for information purposes. No other processing of the `error` JSON message is required.

5.5.8 General Message

5.5.8.1 VWP processing

This section is only relevant to the implementer of the VWP.

5.5.8.1.1 Trigger

VWP processing shall be triggered by the VWP receiving a protected Message to be forwarded to the VWCH. The Message will have an associated priority, which determines to which topic it is sent.

5.5.8.1.2 Processing

The VWP shall construct a `sb_message` JSON message including the members shown in Table 5.5.8.1.2

and send it to one of the following topics depending on its priority:

- `sb/<VWD EUI64>/message/high`

- `sb/<VWD EUI64>/message/medium`
- `sb/<VWD EUI64>/message/low`

The prioritisation rules that the VWP assigns to the Message are not described in this document.

Member	Description
`payload`	The base64-encoded protected Message received from the VWP. The length of the payload may be up to 2000 bytes.

Table: 5.5.8.1.2

Example JSON message:

```
{
  "payload": "ZGVmYXVsdA=="
}
```

5.5.8.2 VWD processing

This section is only relevant to a VWD Manufacturer.

5.5.8.2.1 Trigger

VWD processing shall be triggered by receipt of a `sb_message` JSON message.

5.5.8.2.2 Processing

The VWD shall base64-decode the `payload` member and send the decoded Message to the VWCH using the VWAN Tunnel. A prioritisation scheme shall be used.

The VWD shall be able to buffer up to 10 Messages and send them when the VWAN Tunnel is available. This is illustrated in Figure 5.5.8.2.2 where high priority Messages (e.g. GBCS Messages) are indicated in green and low priority Messages (e.g. firmware transfer Messages) are in red.

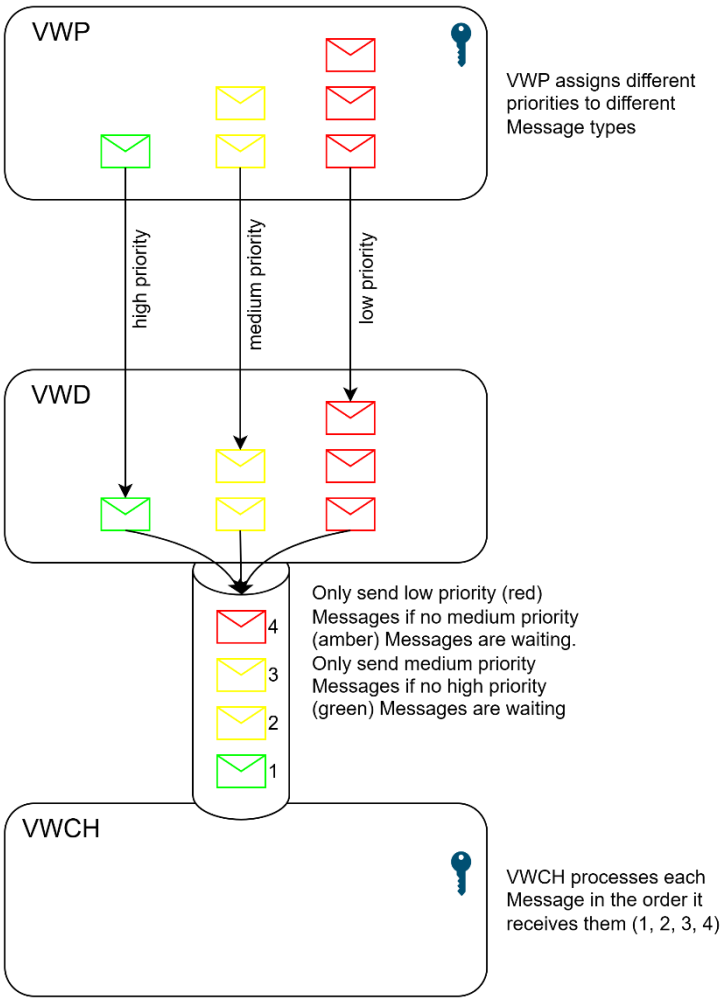

Figure: 5.5.8.2.2

5.6 VWD to VWP JSON messages

This section defines the JSON messages sent from the VWD to the VWP using the MQTT protocol. The schema is defined in the Section 7.1. These are generally known as “northbound” messages.

All JSON messages are effectively notifications. The different VWD to VWP message types reflect the state of the VWD. Any payload that originates from:

- Enhanced Inter-PAN connection with the VWCH
- HHT Tunnel
- VWAN Tunnel

will be base64-encoded verbatim and placed in the payload of the JSON message.

All members are mandatory unless explicitly stated as being optional.

5.6.1 MQTT Topics

In the following topic descriptions, `<VWD EUI64>` refers to the EUI64 used by the VWD to communicate with the VWP. The EUI64 is the same value as used in the VWD Certificate subject field.

Topic	Purpose
nb/<VWD EUI64>/message	Messages to be routed from VWCH
nb/<VWD EUI64>/control	VWD LWT messages
nb/<VWD EUI64>/logs	VWD log messages
nb/<VWD EUI64>/error	VWD error and exception messages

Table: 5.6.1

5.6.2 Interpan Notification

5.6.2.1 VWD processing

This section is only relevant to a VWD Manufacturer.

5.6.2.1.1 Trigger

VWD processing shall be triggered by the VWD completing CBKE with the VWCH successfully.

5.6.2.1.2 Processing

The VWD shall construct an `interpan_notification` JSON message including the members shown in Table 5.6.2.1.2 and send it to the `nb/<VWD EUI64>/message` topic.

Member	Description
gpf_eui64	The GPF EUI64 (obtained from the Zigbee beacon sent by the VWCH).
vwd_eui64	The VWD EUI64.
install_code	A dummy install code (e.g. "000102030405060708090a0b0c0d0e0f"). An install code is needed to mimic the 8.11 SRV that would usually be received from the DCC User. The actual VWD install code shall NOT be used. A CBKE-derived install code is used in inter-PAN joining (see GBCS Section 10.5), which means any install code sent in a CCS01 Command is ignored anyway. This prevents the real install code being sent over an unsecured Enhanced Inter-PAN link.

Table: 5.6.2.1.2

Example JSON message:

```
{
  "gpf_eui64": "0be2cf4d6af3d1be",
  "vwd_eui64": "c45f15dcffa8d55f",
  "interpan_notification": {
    "install_code": "000102030405060708090a0b0c0d0e0f"
  }
}
```

5.6.2.2 VWP processing

This section is only relevant to the implementer of the VWP.

5.6.2.2.1 Trigger

VWP processing shall be triggered by receipt of an `interpan_notification` JSON message.

5.6.2.2.2 Processing

On receipt of an `interpan_notification` JSON message, the VWP shall check the VWD is authorised and, if authorised, request the CCS01 Command from DCC systems. The GPF EUI64 (obtained from the Zigbee beacon) is used to look up the corresponding CHF EUI64, which is the target of the CCS01 Command. The method used by the VWP for checking VWD authorisation is not described in this document.

5.6.3 Allowlist Command Response

5.6.3.1 VWD processing

This section is only relevant to a VWD Manufacturer.

5.6.3.1.1 Trigger

VWD processing shall be triggered by the VWD receiving the CCS01 Response from the VWCH through the Enhanced Inter-PAN connection.

5.6.3.1.2 Processing

The VWD shall construct an `allowlist_command_response` JSON message including the members shown in Table 5.6.3.1.2 and send it to the `nb/<VWD EUI64>/message` topic.

Member	Description
`gpf_eui64`	The GPF EUI64 (obtained from the Zigbee beacon sent by the VWCH).
`vwd_eui64`	The VWD EUI64.
`payload`	The base64-encoded CCS01 Response received from the VWCH through the Enhanced Inter-PAN connection.

Table: 5.6.3.1.2

Example JSON message:

```
{
  "gpf_eui64": "0be2cf4d6af3d1be",
  "vwd_eui64": "c45f15dcffa8d55f",
  "allowlist_command_response": {
    "payload": "ZGVmYXVsdA=="
  }
}
```

5.6.3.2 VWP processing

This section is only relevant to the implementer of the VWP.

5.6.3.2.1 Trigger

VWP processing shall be triggered by receipt of an `allowlist_command_response` JSON message.

5.6.3.2.2 Processing

On receipt of an `allowlist_command_response` JSON message, the VWP shall forward the CCS01 Response to DCC systems for verification. The method used by the VWP for forwarding the CCS01 Response is not described in this document.

5.6.4 EnableVWD request

5.6.4.1 VWD processing

This section is only relevant to a VWD Manufacturer.

5.6.4.1.1 Trigger

VWD processing shall be triggered by the VWD either:

- Joining the SMHAN and requesting the HHT Tunnel successfully.
  - This is indicated by receipt of a RequestTunnelResponse ZSE command with a TunnelStatus value of 0x00 (SUCCESS)
  - The VWD is now effectively operating as an HHT.
- Receiving a Message from the VWCH through the HHT Tunnel.

5.6.4.1.2 RequestTunnelResponse processing

If the VWD has received a RequestTunnelResponse ZSE command with a TunnelStatus value of 0x00 (SUCCESS), the VWD shall construct an `enablevwd_request` JSON message including the members shown in Table 5.6.4.1.2 and send it to the `nb/<VWD EUI64>/message` topic.

Member	Description
`gpf_eui64`	The GPF EUI64 (obtained from the Zigbee beacon sent by the VWCH).
`vwd_eui64`	The VWD EUI64.

Table: 5.6.4.1.2

Note that there is no payload in this case.

Example JSON message:

```
{
  "gpf_eui_64": "0be2cf4d6af3d1be",
  "vwd_eui_64": "c45f15dcffa8d55f"
  "enablevwd_request": {
  }
}
```

5.6.4.1.3 Received Message processing

If the VWD has received a Message from the VWCH, the VWD shall construct an `enablevwd_request` JSON message including the members shown in Table 5.6.4.1.3 and send it to the `nb/<VWD EUI64>/message` topic.

Member	Description
`gpf_eui64`	The GPF EUI64 (obtained from the Zigbee beacon sent by the VWCH).
`vwd_eui64`	The VWD EUI64.
`payload`	The base64-encoded Message received from the VWCH through the VWAN Tunnel.

Table: 5.6.4.1.3

Example JSON message:

```
{
  "gpf_eui_64": "0be2cf4d6af3d1be",
  "vwd_eui_64": "c45f15dcffa8d55f"
  "enablevwd_request": {
    "payload": "ZGVmYXVsdA=="
  }
}
```

5.6.4.2 VWP processing

This section is only relevant to the implementer of the VWP.

5.6.4.2.1 Trigger

VWP processing shall be triggered by receipt of an `enablevwd_request` JSON message.

5.6.4.2.2 Processing

On receipt of an `enablevwd_request` JSON message, the VWP shall check the VWD is authorised.

If the VWD is authorised and the VWP has not already requested the EnableVWD Command for the VWD from DCC systems, the VWP shall request the EnableVWD Command from DCC systems and record that the EnableVWD Command has been requested for the corresponding VWD. The method used by the VWP for requesting the EnableVWD Command is not described in this document.

Note that the GPF EUI64 is used by DCC systems to look up the corresponding CHF EUI64, which is the target of the EnableVWD Command.

5.6.5 EnableVWD command response

5.6.5.1 VWD processing

This section is only relevant to a VWD Manufacturer.

5.6.5.1.1 Trigger

VWD processing shall be triggered by the VWD receiving the EnableVWD Response or any other Alert from the VWCH. Note that the VWD cannot distinguish between Message types from the VWCH so will forward any Message it receives.

5.6.5.1.2 Processing

The VWD shall construct an `enablevwd_command_response` JSON message including the members shown in Table 5.6.5.1.2 and send it to the `nb/<VWD EUI64>/message` topic.

Member	Description
`gpf_eui64`	The GPF EUI64 (obtained from the Zigbee beacon sent by the VWCH).
`vwd_eui64`	The VWD EUI64.
`payload`	The base64-encoded EnableVWD Response or Alert received from the VWCH through the HHT tunnel.

Table: 5.6.5.1.2

Example JSON message:

```
{
  "gpf_eui_64": "0be2cf4d6af3d1be",
  "vwd_eui_64": "c45f15dcffa8d55f",
  "enablevwd_command_response": {
    "payload": "ZGVmYXVsdA=="
  }
}
```

5.6.5.2 VWP processing

This section is only relevant to the implementer of the VWP.

5.6.5.2.1 Trigger

VWP processing shall be triggered by receipt of an `enablevwd_command_response` JSON message.

5.6.5.2.2 Processing

On receipt of an `enablevwd_command_response` JSON message, the VWP shall forward the EnableVWD Response or Alert to DCC systems for verification. The method used by the VWP for forwarding the EnableVWD Response or Alert is not described in this document.

5.6.6 Error reporting

5.6.6.1 VWD processing

This section is only relevant to a VWD Manufacturer.

5.6.6.1.1 Trigger

VWD processing shall be triggered by the VWD encountering an error/exception to report to the VWP. The error/exception encountered is manufacturer-specific and there is no requirement to report specific errors.

5.6.6.1.2 Processing

The VWD shall construct an `error` JSON message including the members shown in Tables 5.6.6.1.2 and send it to the `nb/<VWD EUI64>/error` topic.

Member	Description
`gpf_eui64`	The GPF EUI64 (obtained from the Zigbee beacon sent by the VWCH).
`vwd_eui64`	The VWD EUI64.
`error`	The error entry.

The error entry shall contain the following members:

Member	Description
`code`	The error code using an arbitrary string format. The data values are not described in this document.
`desc`	A description of the error using an arbitrary string format (maximum 255 characters). The data values are not described in this document.
`timestamp`	Timestamp using ISO 8601 format.

Tables: 5.6.6.1.2

The values that can be used in a log entry are not described in this document.

Example JSON message:

```
{
  "gpf_eui_64": "0be2cf4d6af3d1be",
  "vwd_eui_64": "c45f15dcffa8d55f",
  "error": {
    "code": "nnnnnnnn",
    "desc": "xxxx....xxxx",
    "timestamp": "2025-01-26T22:10:00Z"
  }
}
```

5.6.6.2 VWP processing

This section is only relevant to the implementer of the VWP.

5.6.6.2.1 Trigger

VWP processing shall be triggered by receipt of an `error` JSON message.

5.6.6.2.2 Processing

The VWP shall forward the error to DCC systems for further processing. The method used by the VWP for forwarding the error is not described in this document.

5.6.7 Log reporting

Up to 100 log entries may be included in a single `log` message.

5.6.7.1 VWD processing

This section is only relevant to a VWD Manufacturer. There is no requirement to perform any specific logging.

5.6.7.1.1 Trigger

VWD processing shall be triggered by the VWD having one or more log entries to report to the VWP.

5.6.7.1.2 Processing

The VWD shall construct a `log` JSON message including the members shown in Tables 5.6.7.1.2 and send it to the `nb/<VWD EUI64>/logs` topic:

Member	Description
`gpf_eui64`	The GPF EUI64 (obtained from the Zigbee beacon sent by the VWCH).
`vwd_eui64`	The VWD EUI64.
`log`	An array of log entries.

Each log entry shall contain the following members.

Member	Description
`level`	Log level (“info”, “warning” or “error”). The data values are not described in this document.
`message`	A message using an arbitrary string format (maximum 255 characters). The data values are not described in this document.
`timestamp`	Timestamp using ISO 8601 format.
`metadata`	Optional metadata using an arbitrary string format. The data values are not described in this document.
`tag`	Optional tag using an arbitrary string format. The data values are not described in this document.

Table: 5.6.7.1.2

Example JSON message:

```
{
  "gpf_eui_64": "0be2cf4d6af3d1be",
  "vwd_eui_64": "c45f15dcffa8d55f",
  "log": [
    {
      "level": "info",
      "message": "Log 1",
      "timestamp": "2025-01-26T22:10:00Z"
    },
    {
      "level": "info",
      "message": "Log 2",
      "timestamp": "2025-01-26T22:10:00Z",
      "metadata": "metadata 1",
      "tag": "tag 1"
    },
    {
      "level": "info",
      "message": "Log File sent",
      "timestamp": "2025-01-26T22:10:00Z"
    }
  ]
}
```

5.6.7.2 VWP processing

This section is only relevant to the implementer of the VWP.

5.6.7.2.1 Trigger

VWP processing shall be triggered by receipt of a `log` JSON message.

5.6.7.2.2 Processing

The VWP shall forward the logs to DCC systems for further processing. The method used by the VWP for forwarding the logs is not described in this document.

5.6.8 General Message

5.6.8.1 VWD processing

This section is only relevant to a VWD Manufacturer.

5.6.8.1.1 Trigger

VWD processing shall be triggered by the VWD receiving a protected Message over the VWAN tunnel to be forwarded to the VWP.

5.6.8.1.2 Processing

The VWD shall construct a `nb_message` JSON message including the members shown in Table 5.6.8.1.2 and send it to the `nb/<VWD EUI64>/message` topic:

Member	Description
`payload`	The base64-encoded protected Message received from the VWCH through the VWAN Tunnel. The length of the payload may be up to 2000 bytes.

Table: 5.6.8.1.2

Example JSON message:

```
{
  "payload": "ZGVmYXVsdA=="
}
```

5.6.8.2 VWP processing

This section is only relevant to the implementer of the VWP.

5.6.8.2.1 Trigger

VWP processing shall be triggered by receipt of a `message` JSON message.

5.6.8.2.2 Processing

The VWP shall base64-decode, decrypt and verify the protected Message before forwarding to DCC systems for further processing. The method used by the VWP for decrypting, verifying and forwarding the Message is not described in this document.

6 Error and status codes

This section lists error and status codes generated by the VWP.

6.1 Error and status code grouping

Error and status code range	Error or status type
1nn	Information/Warning message
2nn	Success messages
3nn	Device-side error
4nn	Server-side error
5nn	Communication/network/system error

Table: 6a

6.2 Specifc error and status codes

Error or status code	Error or status description
200	Success
401	VWD Device ID does not exist in the SMI
402	CHF/GPF Device ID does not exist in the SMI
403	VWD is not pre-notified by the correct user role
404	CCS01 Command generation error
405	Cryptographic error in Command generation
406	Device status in the SMI is incorrect for the requested operation
410	Cryptographic error in Response verification
411	CCS01 Response content is invalid
412	CCS01 Response status code is not success
413	CCS01 Response verification failure
415	EnableVWD Command generation error
419	EnableVWD Response content is invalid

420	EnableVWD Response status code is not success
421	EnableVWD Response verification failure
425	Key agreement failure
500	Invalid request, due to malformed JSON
501	Other server exceptions
502	System busy
503	Unrecoverable system error

Table: 6b

7 JSON schema

JSON schemas are maintained by the DCC as part of this interface specification, however the VWP and VWD manufacturers are under no obligation to use the schema as part of their implementation provided they have an equivalent means of generating and validating JSON objects that conform to this interface specification.

7.1 Northbound schema

```
{
  "$comment": "Filename : VPIS nb schema.json | Schema Version : 0.2 | Aligned with
VPIS version : 0.4 | Date : February 2025 | Minor revision : 0.0",
  "$schema": "https://json-schema.org/draft-07/schema",
  "title": "VPIS Schema for northbound VWAN messages",
  "description": "Schema for Virtual WAN messages from Virtual WAN Device to
Virtual WAN Provider",
  "type": "object",
  "required": [
    "gpf_eui_64",
    "vwd_eui_64"
  ],
  "oneOf": [
    {
      "required": ["interpan_notification"]
    },
    {
      "required": ["allowlist_command_response"]
    },
    {
      "required": ["enablevwd_request"]
    },
    {
      "required": ["enablevwd_command_response"]
    },
    {
      "required": ["payload"]
    },
    {
      "required": ["error"]
    },
    {
      "required": ["log"]
    }
  ],
  "properties": {
    "gpf_eui_64": {
      "type": "string",
      "pattern": "^[0-9a-f]{16}$",
      "description": "EUI-64 unique identifier for the CH zigbee interface"
    },
    "vwd_eui_64": {
      "type": "string",
      "pattern": "^[0-9a-f]{16}$",
      "description": "EUI-64 unique identifier for the VWD zigbee interface"
    },
    "interpan_notification": {
      "type": "object",
      "properties": {
        "install_code": {
          "type": "string",
          "pattern": "^[0-9a-f]{32}$",
          "description": "Install code of the VWD"
        }
      },
      "required": ["install_code"]
    },
```

```
"allowlist_command_response": {
  "type": "object",
  "properties": {
    "payload": {
      "type": "string",
      "contentEncoding": "base64",
      "description": "The GB Companion Specification formatted message"
    }
  },
  "required": ["payload"]
},
"enablevwd_request": {
  "type": "object",
  "additionalProperties": false
},
"enablevwd_command_response": {
  "type": "object",
  "properties": {
    "payload": {
      "type": "string",
      "contentEncoding": "base64",
      "description": "The GB Companion Specification formatted message"
    }
  },
  "required": ["payload"]
},
"payload": {
  "type": "string",
  "contentEncoding": "base64",
  "description": "The GB Companion Specification formatted message"
},
"error": {
  "type": "object",
  "properties": {
    "code": {
      "type": "string",
      "description": "Error code.",
      "maxLength": 8
    },
    "desc": {
      "type": "string",
      "maxLength": 255,
      "description": "Error description"
    },
    "timestamp": {
      "type": "string",
      "format": "date-time",
      "description": "Date and time when error is reported"
    }
  },
  "required": [
    "code",
    "timestamp"
  ]
},
"log": {
  "type": "array",
  "items": {
    "type": "object",
    "properties": {
      "level": {
        "type": "string",
        "enum": [
          "debug",
          "info",
          "error",
          "warn",
          "critical"
```

```
            ],
            "description": "Log level"
          },
          "message": {
            "type": "string",
            "maxLength": 255,
            "description": "Log message"
          },
          "timestamp": {
            "type": "string",
            "format": "date-time",
            "description": "Date and time when error is reported"
          },
          "metadata": {
            "type": "string",
            "maxLength": 255,
            "description": "Additional context"
          },
          "tag": {
            "type": "string",
            "maxLength": 255,
            "description": "For easier search"
          }
        }
      },
      "required": [
        "level",
        "message",
        "timestamp"
      ],
      "minItems": 1,
      "maxItems": 100
    }
  },
  "additionalProperties": false
}
```

7.2 Southbound schema

```
{
  "$comment": "Filename : VPIS sb schema.json | Schema Version : 0.2 | Aligned with
VPIS version : 0.4 | Date : February 2025 | Minor revision : 0.0",
  "$schema": "https://json-schema.org/draft-07/schema",
  "title": "VPIS Schema for southbound VWAN messages",
  "description": "Schema for Virtual WAN messages from Virtual WAN Provider to
Virtual WAN Device",
  "type": "object",
  "oneOf": [
    {
      "required": ["allowlist_request"]
    },
    {
      "required": ["allowlist_response_outcome"]
    },
    {
      "required": ["enablevwd_request_outcome"]
    },
    {
      "required": ["enablevwd_response_outcome"]
    },
    {
      "required": ["vwd_ka_outcome"]
    },
    {
      "required": ["payload"]
    },
    {
      "required": ["error"]
```

```
  }
],
"properties": {
  "allowlist_request": {
    "type": "object",
    "properties": {
      "response_code": {
        "type": "string",
        "description": "Success or error code",
        "maxLength": 8
      },
      "payload": {
        "type": "string",
        "contentEncoding": "base64",
        "description": "The GB Companion Specification formatted message"
      }
    },
    "required": ["response_code"]
  },
  "allowlist_response_outcome": {
    "type": "object",
    "properties": {
      "response_code": {
        "type": "string",
        "description": "Success or error code",
        "maxLength": 8
      }
    },
    "required": ["response_code"]
  },
  "enablevwd_request_outcome": {
    "type": "object",
    "properties": {
      "response_code": {
        "type": "string",
        "description": "Success or error code",
        "maxLength": 8
      },
      "payload": {
        "type": "string",
        "contentEncoding": "base64",
        "description": "The GB Companion Specification formatted message"
      }
    },
    "required": ["response_code"]
  },
  "enablevwd_response_outcome": {
    "type": "object",
    "properties": {
      "response_code": {
        "type": "string",
        "description": "Success or error code",
        "maxLength": 8
      }
    },
    "required": ["response_code"]
  },
  "vwd_ka_outcome": {
    "type": "object",
    "properties": {
      "response_code": {
        "type": "string",
        "description": "Success or error code",
        "maxLength": 8
      }
    },
    "required": ["response_code"]
  },
```

```
    "payload": {
      "type": "string",
      "contentEncoding": "base64",
      "description": "The GB Companion Specification formatted message"
    },
    "error": {
      "type": "object",
      "properties": {
        "code": {
          "type": "string",
          "description": "Error code",
          "maxLength": 8
        },
        "desc": {
          "type": "string",
          "maxLength": 255,
          "description": "Error description"
        },
        "timestamp": {
          "type": "string",
          "format": "date-time",
          "description": "Date and time when error is reported"
        }
      },
      "required": [
        "code",
        "timestamp"
      ]
    }
  },
  "additionalProperties": false
}
```

8 Glossary

8.1 Abbreviations

Abbreviation	Meaning
CAD	Consumer Access Device
CBKE	Certificate-based Key Establishment
CSP	Communications Service Provider
DCC	Data Communications Company
DLMS	Device Language Message Specifications
DSP	Data Service Provider
DUIS	DCC User Interface Specification
FQDN	Fully Qualified Domain Name
GBCS	Great Britain Companion Specification
GSME	Gas Smart Metering Equipment
HAN	Home Area Network
HHT	Hand-held Terminal
IHD	In-home Display
JSON	Javascript Object Notation
LWT	MQTT Last Will and Testament
MQTT	Message Queuing Telemetry Transport
PPMID	Prepayment Metering Interface Device
SMHAN	Smart Metering Home Area Network
SMI	Smart Metering Inventory
SMWAN	Smart Metering Wide Area Network
VWAN	Virtual Wide Area Network
VWCH	Virtual WAN Comms Hub
VWD	Virtual WAN Device
VWKI	Virtual WAN Key Infrastructure

VWP	Virtual WAN Provider
WAN	Wide Area Network
ZSE	Zigbee Smart Energy