Zigbee Hub

Build your local ZigBee network without Z2M/ZHA

- About Zigbee Hub mode
- MQTT API

About Zigbee Hub mode

4. About Zigbee Hub mode

> **It is highly recommended to use U series coordinators for this mode (SLZB-06xU / MRxU)**

4.1 What is Zigbee Hub in SLZB-OS?

The **Zigbee Hub** feature allows the SLZB device to run its Zigbee network **directly on the device** — without needing an external computer, Raspberry Pi, or NAS to host the Zigbee stack.
In Zigbee Hub mode, SLZB-OS launches an integrated Zigbee stack service that can connect directly to your smart home platform over MQTT.

This mode is ideal for:

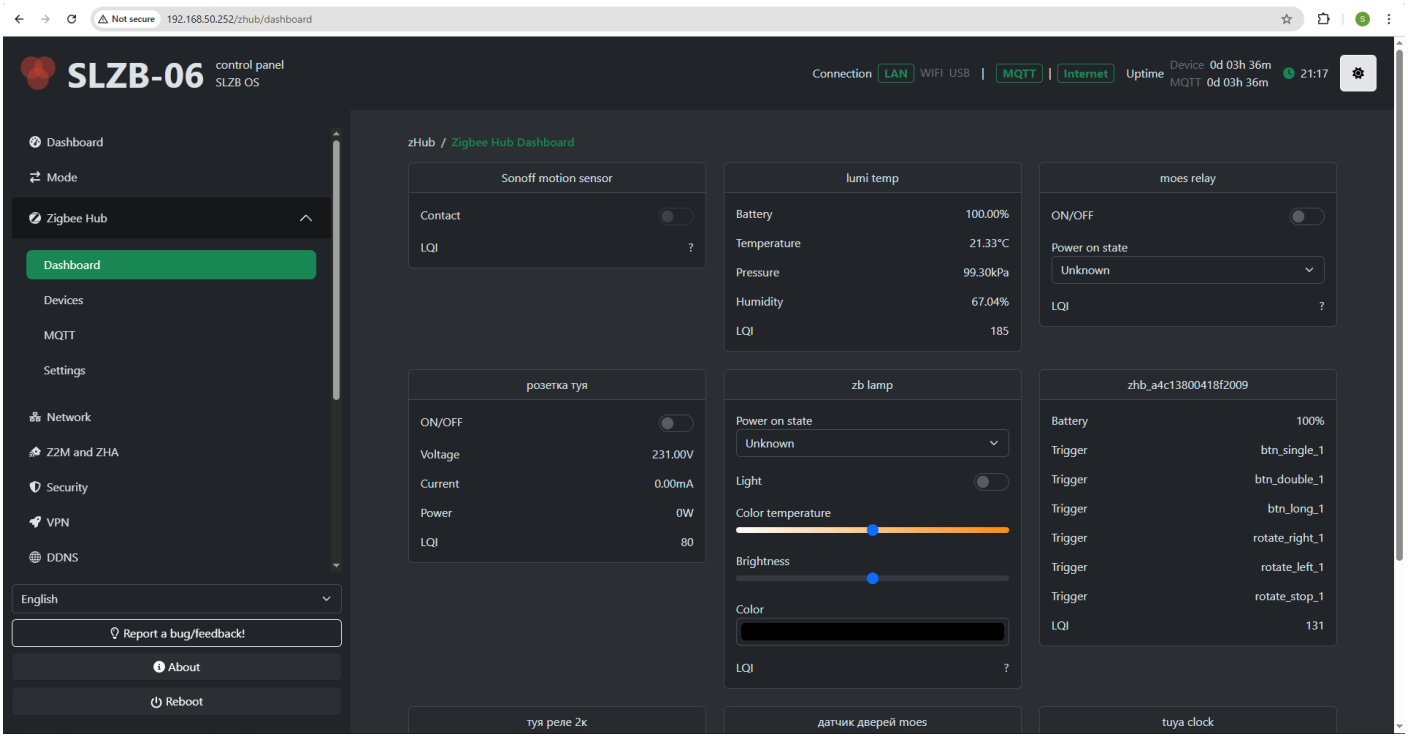
- **Self-contained setups** where the device acts as both the coordinator and the host.
- **Reducing complexity** by eliminating extra hardware.
- **PoE/Ethernet-based installations** for maximum stability.

When Zigbee Hub is active, a dedicated **Zigbee Hub** menu appears in the SLZB-OS interface, containing the following pages:

1. **Dashboard** – Live overview of Zigbee network status.
2. **Devices** – List and manage all paired Zigbee devices.
3. **MQTT** – Configure the MQTT broker connection.
4. **Settings** – Advanced Zigbee network and coordinator options.

4.2 Zigbee Hub ? Dashboard

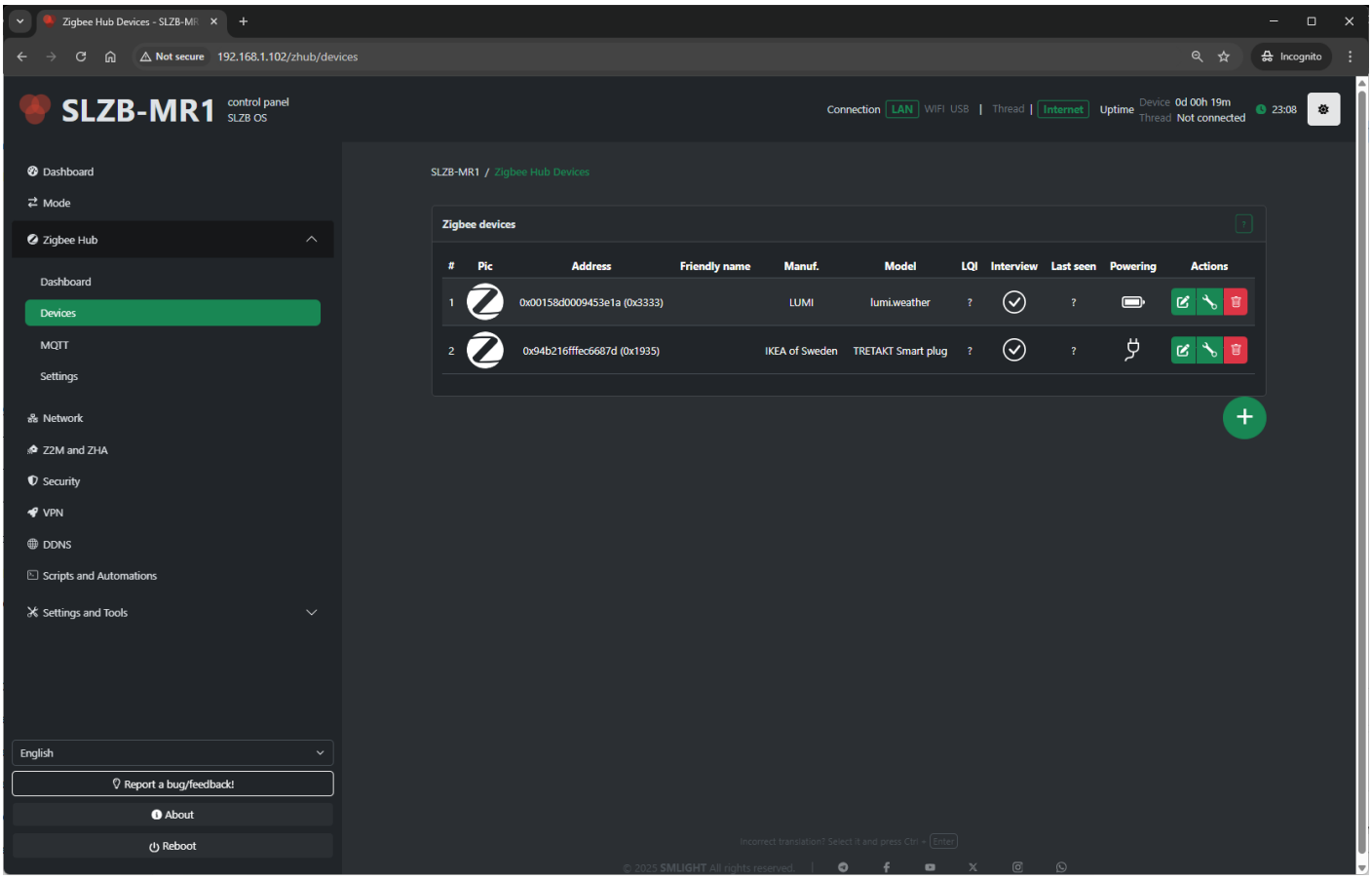
The **Dashboard** is the central monitoring page for your Zigbee network.



The dashboard contains cards of your ZigBee devices with the data they provide and controls. The dashboard is updated in real time via SSE.

4.3 Zigbee Hub ? Devices

The **Devices** page lists every Zigbee device paired to your coordinator.



Pairing control:

- **Permit Join** (+ button on the bottom right side of the device table) – Allow or deny new devices joining the network.

For each device, you'll see:

- **Name / Friendly Name** – Human-readable identifier.
- **IEEE Address** – Unique device ID.
- **Network Address** – Short Zigbee address assigned by the coordinator.
- **Last Seen** – Timestamp of the last communication.
- **Powering** – Device power source (AC/battery) or ? if the device does not provide information.
- **Link Quality (LQI)** – Signal strength indicator.
- **Actions:**
  - Rename device
  - Remove/unpair device
  - View device details (clusters, endpoints, bindings)
  - Bind/unbind devices (if supported)

Typical uses:

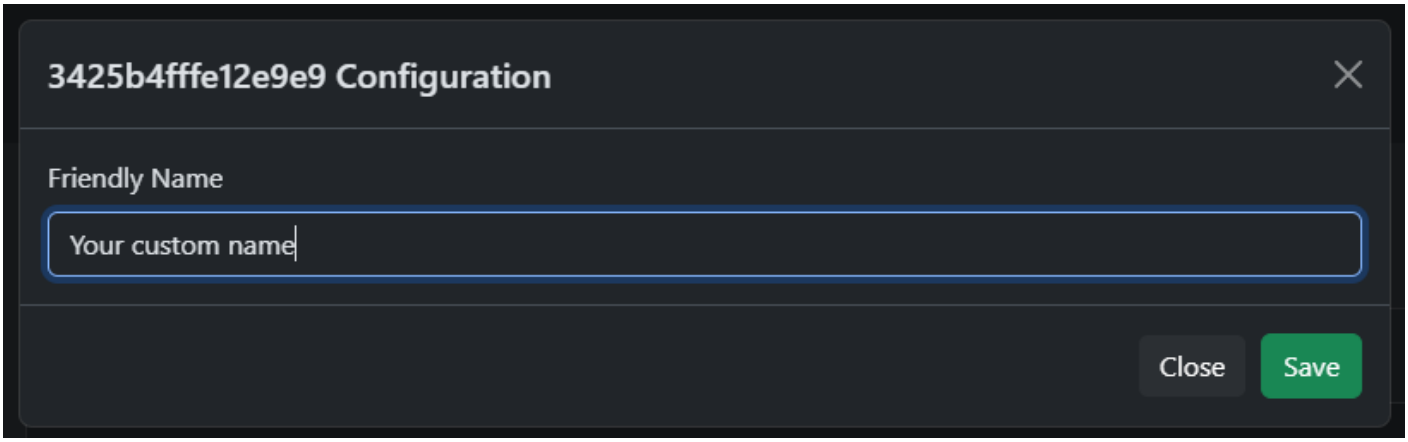
- Verify devices are online and responsive.
- Rename devices for easier identification in automations.

- Remove devices no longer in use.

Device config

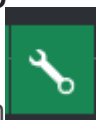
How to rename device

Select the pencil icon 

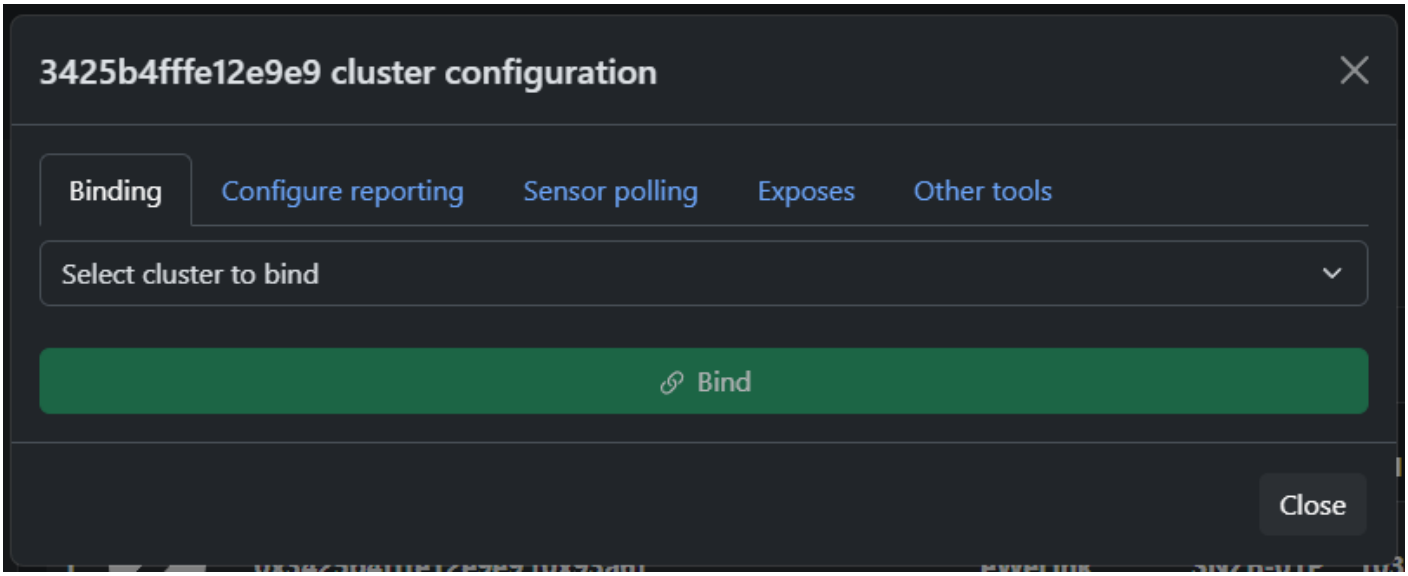


Enter new name and press "Save". Maximum length - 50 characters.

Device config

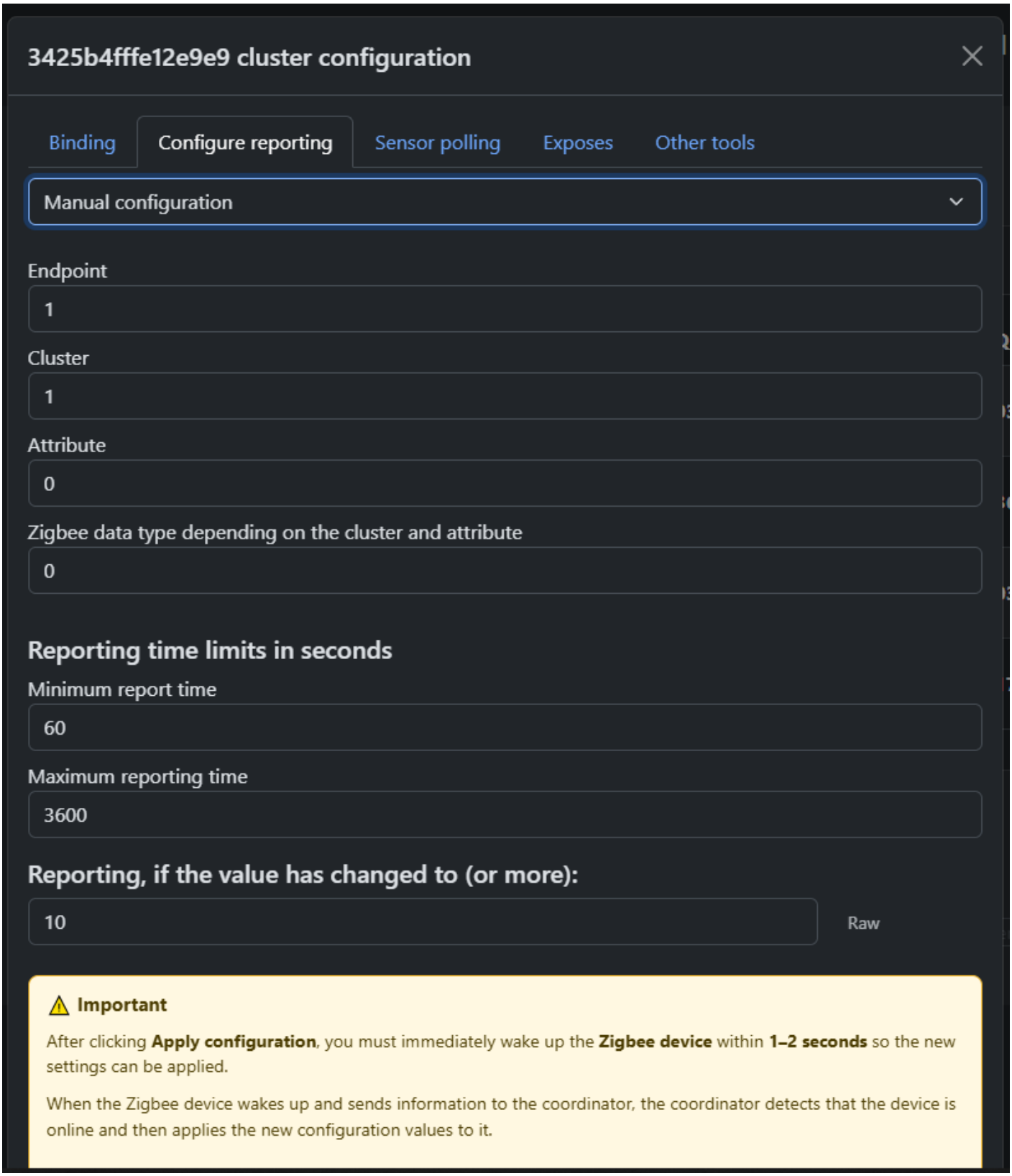
Click on the wrench 

Binding

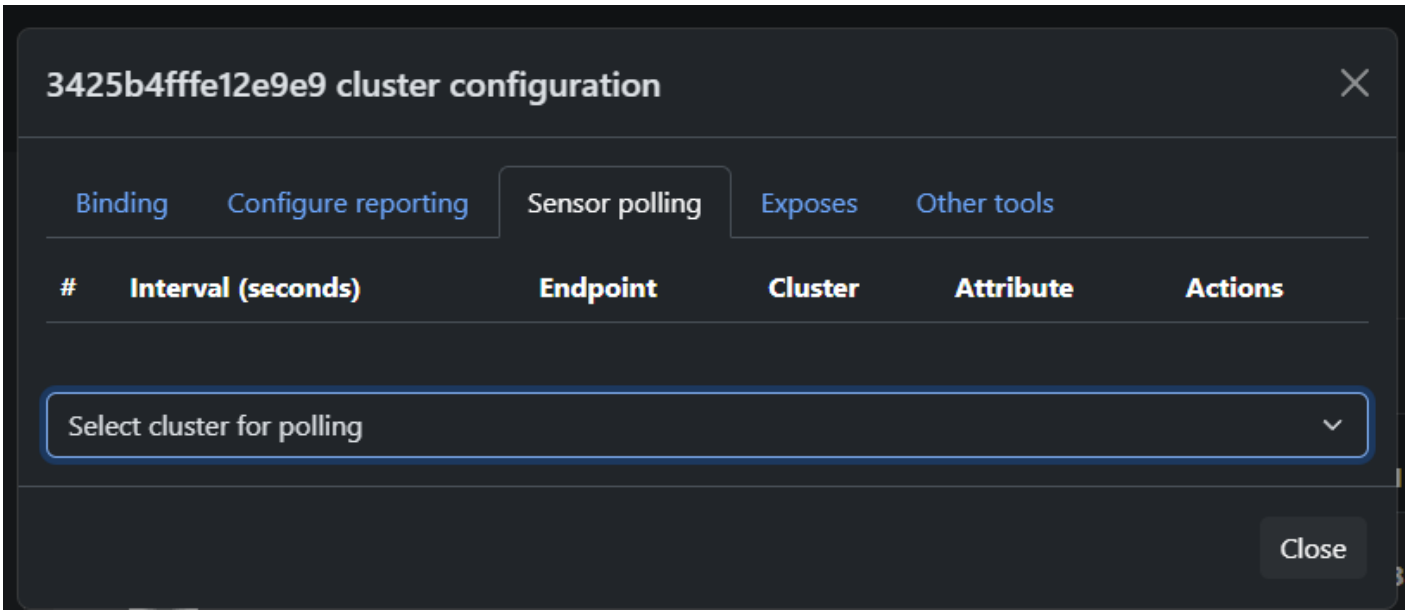


This menu sends the device a bind request **to the coordinator**, the device must be active to accept this. If it is a battery-powered device you need to wake it up.

Configure reporting



Polling



Allows you to configure polling of the selected attribute after a certain time interval

Exposes

3425b4fffe12e9e9 cluster configuration

Binding | Configure reporting | Sensor polling | Exposes | Other tools

Battery

MQTT status topic: `zhub/data/3425b4fffe12e9e9/1/0001/0021`
Values: `100`

HTTP GET sample: `http://192.168.50.252/api2?param=6&action=9&topic=zhub/data/3425b4fffe12e9e9/1/0001/0021`

Berry samples:

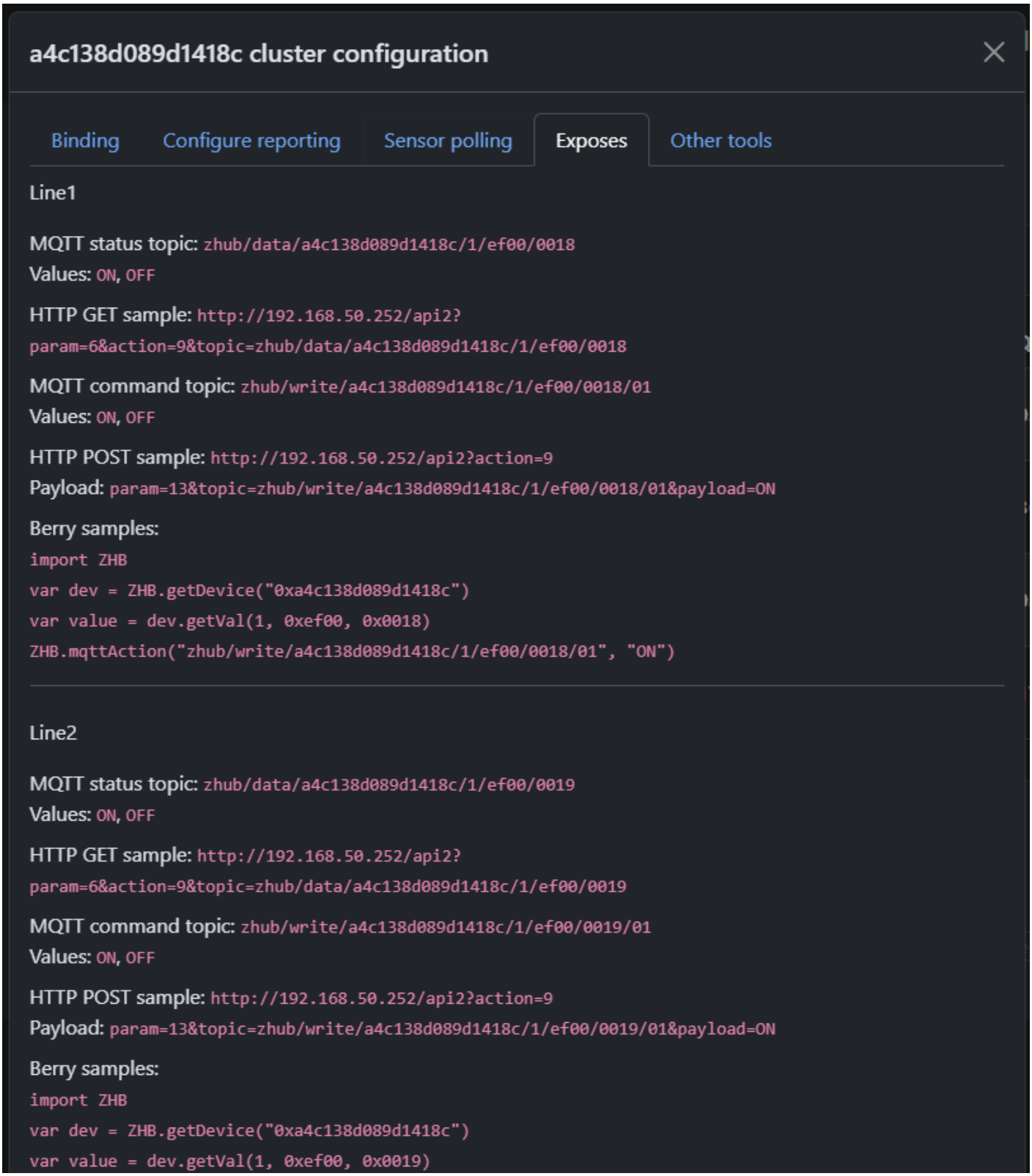
```
import ZHB
var dev = ZHB.getDevice("0x3425b4fffe12e9e9")
var value = dev.getVal(1, 0x0001, 0x0021)
```

Actions

MQTT topic: zhub/trigger/3425b4fffe12e9e9
Payload: `single`, `double`, `long`
Berry example:

```
import ZHB
ZHB.on_action(
def(action, device)
SLZB.log(action) # will log single, double, long
SLZB.log(device.getName()) # will log device name
end
)
```

Close

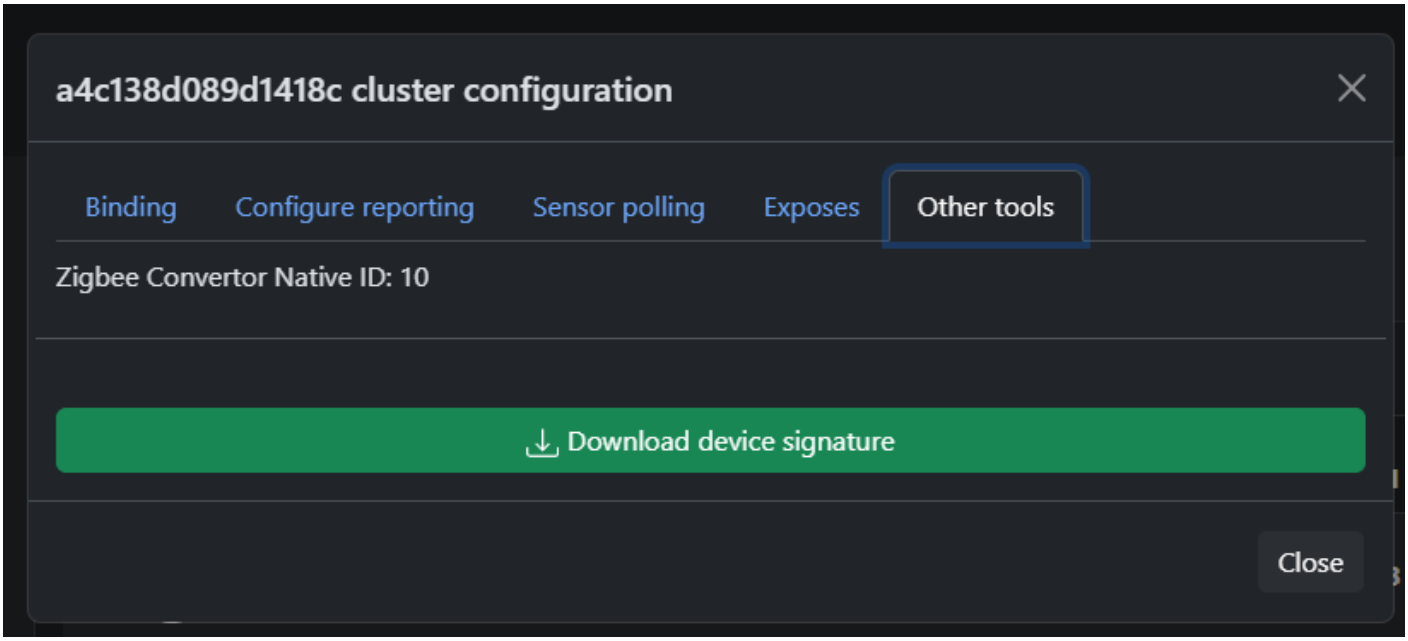


Provides information about **expected** data from the device and examples of using MQTT, HTTP, and Berry API to get or set device state.

> **IMPORTANT!**
> This section provides information about **EXPECTED** data from the device, based on information about the device clusters or converter (if it exists).
> This information may not match the actual behavior of the device if it uses non-standard

clusters!
If this is a Tuya DP device, then information will be displayed here only if a converter exists for the device!

Other tools

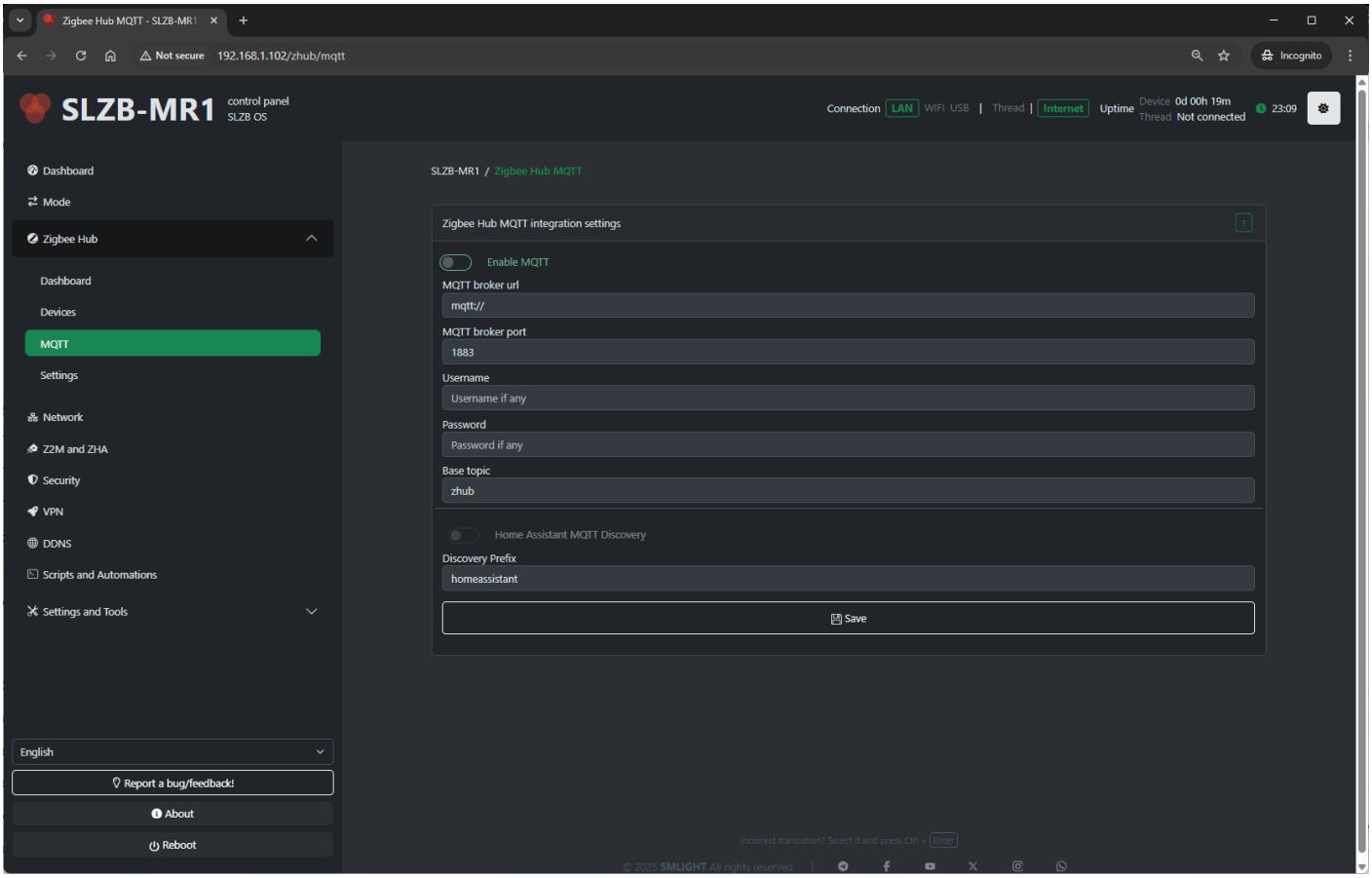


Here you can find the ZCN converter number (if used) and download device information

---

4.4 Zigbee Hub ? MQTT

This page configures the MQTT connection that Zigbee2MQTT (running on SLZB-OS) uses to communicate with your smart home platform.



Configuration fields include:

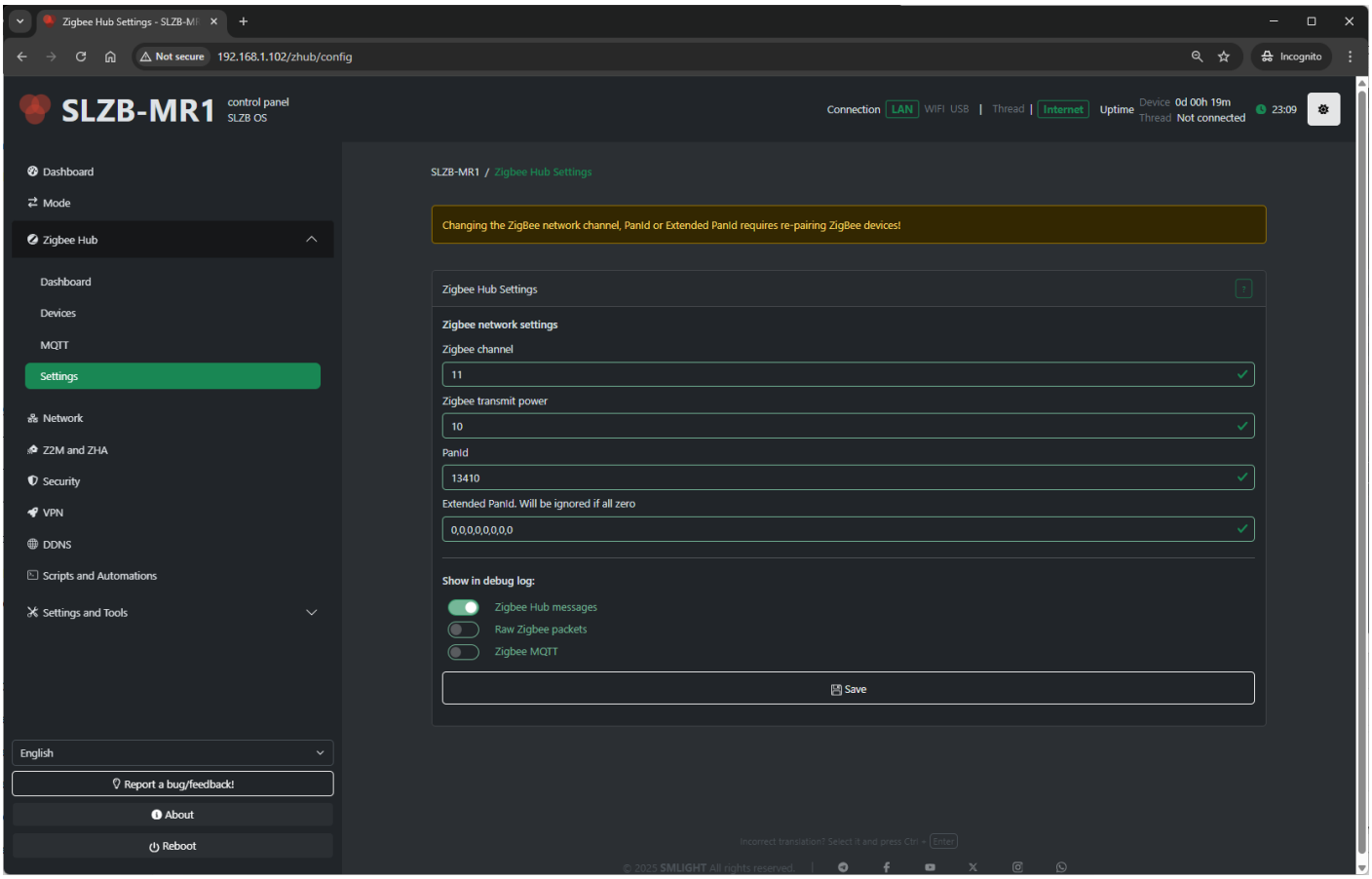
- **MQTT Server Address** – IP or hostname of your broker (e.g., `mqtt://192.168.1.100`).
- **Port** – Default 1883 for MQTT, 8883 for MQTT over TLS.
- **Username / Password** – Broker authentication (if required).
- **Base Topic** – Topic prefix for Zigbee messages (default: `zigbee2mqtt`).
- **Discovery Prefix** - Home assistant main topic name.

Tips:

- For Home Assistant with Mosquitto add-on, use the HA IP and port `1883`.
- Always use a unique base topic if you run multiple Zigbee networks.
- Save and restart Zigbee Hub after making MQTT changes.

4.5 Zigbee Hub ? Settings

The **Settings** page contains deeper configuration for the Zigbee coordinator and Zigbee2MQTT service.



Typical settings available:

- **Network Parameters**:
  - **PAN ID** – Zigbee network identifier.
  - **Channel** – RF channel (11–26; avoid Wi-Fi overlap if possible).
  - **Extended PAN ID** – Long network identifier.
- **Transmit Power** – Radio TX power in dBm (higher = longer range, more power draw).
- **Show in debug log -** Select which categories of Zigbee-related information are recorded in the Log & Debug page.
  - **Zigbee Hub messages** – Logs high-level events from the Zigbee Hub service (e.g., device joins, status updates).
  - **Raw Zigbee packets** – Logs low-level Zigbee frame data; useful for deep protocol debugging.
  - **Zigbee MQTT** – Logs MQTT messages related to Zigbee communication, including publishes and subscriptions.

Best practices:

- Change the Zigbee channel only on a fresh network (re-pair required after change).
- Keep `Permit Join` disabled most of the time for security.
- Adjust transmit power to match your coverage needs and regulatory limits.

4.6 Troubleshooting

Problems when starting a Zigbee network

> OS v3.0.9 update is breaking. If you updated OS to v3.0.9 and started getting this error then follow the instructions below

Network commissioning timed out - most likely network with the same panId or extendedPanId already exists nearby.
Network formation refused there is too much RF interference or network with the same panId or extendedPanId already exists.
If you got this error after updating Zigbee chip:
- turn off coordinator
- turn off ALL Zigbee routers that were connected to Zigbee Hub. **This is important, it will not work without this**
- turn on coordinator. Zigbee Hub should start now but zigbee devices will be unavailable
- remove all devices (click on the red trash can)
- download and run berry script below, this script will keep permit join enabled as long as it is running

```
#META {"start":0}
#Insert your code below
import ZHB

ZHB.waitForStart(0xFF)

while 1
  ZHB.permitJoin(254)
  SLZB.delay(255 * 1000)
end
```

- turn back on previously disabled zigbee devices
- repair all devices
- after devices repaired you can stop and delete script

Other cases:
- move the coordinator away from the wifi router
- make sure there is no other coordinator nearby with the same zigbee network settings

MQTT API

Zigbee Hub mode has the ability to connect to local or remote MQTT brokers. Currently only TCP connections are supported.
This document describes MQTT API for SLZB-OS 3.0.9 or higher

Topics are divided into **IN** and **OUT**.
IN - you can send messages to these topics.
OUT - Zigbee Hub sends messages to these topics.

zHub topics format

Data topic (OUT)

General `data` topic format below:
`base topic` / `data` / `zigbee device ieee` / `zigbee endpoint` / `zigbee cluster` / `zigbee attribute`

`base topic` - global prefix, so you can have few Zigbee Hubs connected to one broker, you just have to set different base topics.
`data` - static text (topic type)
`zigbee device ieee` - sender IEEE address in HEX format.
`zigbee endpoint` - zigbee endpoint from which this message is coming, DEC format.

`zigbee cluster` - zigbee cluster from which this message is coming, HEX format.
`zigbee attribute` - zigbee attribute from which this message is coming, HEX format.

Data topic example: `zhub/data/a4c1383439bf5cc9/1/0000/0001`

Read topic (IN) (available from v3.3.0)

This topic accepts requests to read attributes of ZigBee devices.
When you send a message to this topic, the coordinator generates and sends a request for the specified attribute to the target ZigBee device. The device must be online to accept the request, so this is usually used for AC-powered devices.

Topic format: `base topic` / `read` / `zigbee device ieee` / `zigbee endpoint` / `zigbee cluster` / `zigbee attribute`

`base topic` - global prefix, so you can have few Zigbee Hubs connected to one broker, you just have to set different base topics.
`read` - static text (topic type)
`zigbee device ieee` - sender IEEE address in HEX format.
`zigbee endpoint` - zigbee endpoint from which this message is coming, DEC format.
`zigbee cluster` - zigbee cluster from which this message is coming, HEX format.
`zigbee attribute` - zigbee attribute from which this message is coming, HEX format.

Read topic example: `zhub/read/a4c1383439bf5cc9/1/0006/0000`
payload: `any text`

> Attribute update will be sent to `data` topic!

> **PLEASE NOTE!**
> **You must add any text content to the payload!**
> **Do not send an empty payload to this topic!**
> An empty payload will only delete this topic (if it exists). The coordinator will not respond to empty payloads.

Command topic (IN)

This topic is intended for sending ZCL commands to a Zigbee device.
General `cmd` topic format below:
`base topic` / `cmd` / `zigbee device ieee` / `zigbee endpoint` / `zigbee cluster` / `zigbee command`

`base topic` - global prefix, so you can have few Zigbee Hubs connected to one broker, you just have to set different base topics.
`cmd` - static text (topic type)

`zigbee device ieee` - target zigbee device IEEE address in HEX format.
`zigbee endpoint` - target zigbee device endpoint to which this message is coming, DEC format.
`zigbee cluster` - target zigbee device cluster to which this message is coming, HEX format.
`zigbee command` - zigbee command to send, HEX format.

Some clusters have special handlers for input commands, such as the ON/OFF or Light cluster. You can find their formats below:

ON/OFF cluster payload format for command topic

`ON` - send ON command
`OFF` - send OFF command

Example: `zhub/cmd/a4c1383439bf5cc9/1/0006` payload: `ON`
Will send command to enable relay or light device.

Level control for Light cluster payload format

`0000` command payload is a number in DEC from 1 to 254, the larger the number, the brighter the lamp will be.

Color control cluster payload format

`0007` command payload is a **color** in **HEX** or **RGB** format. For example: `255,29,0` or `#FFFFFF`

`000a` command payload is a **light temperature** in **mired**. For example: 200

Other clusters

If no built-in clusters or ZCN converters has overridden the processing of this command, if the command contains a payload, it must be a HEX string of the following format:
- Bytes without spaces, uppercase, multiple of two. Example: `010203FF`
- Bytes with spaces, uppercase, grouped by two. Example: `01 02 03 FF`

Write topic (IN)

This topic is intended for writing ZigBee device ZCL attributes.
General `write` topic format below:
`base topic` / `write` / `zigbee device ieee` / `zigbee endpoint` / `zigbee cluster` / `zigbee attribute`

`base topic` - global prefix, so you can have few Zigbee Hubs connected to one broker, you just have to set different base topics.
`write` - static text (topic type)
`zigbee device ieee` - target zigbee device IEEE address in HEX format.
`zigbee endpoint` - target zigbee device endpoint to which this message is coming, DEC format.
`zigbee cluster` - target zigbee device cluster to which this message is coming, HEX format.
`zigbee attribute` - target zigbee device attribute to which this message is coming, HEX format.

Cluster 0006, attribute 4003 payload format

`Last state` - device will remember its state
`ON` - device will be on after power loss
`OFF` - device will be off after power loss

Cluster EF00 (Tuya DP) payload format

> **Please note that the write topic format for Tuya is different!**
> `base topic` / `write` / `zigbee device ieee` / `1` / `ef00` / `data point` / `data type`

`data point` - Tuya data point id. You can read more about it here:
https://www.zigbee2mqtt.io/advanced/support-new-devices/03_find_tuya_data_points.html
`data type` - a number that represents the type of data being sent.
0 - RAW, **1** - BOOL, **2** - INT, **3** - STRING, **4** - ENUM, **5** - BITMAP

Examples:
topic: zhub/write/a4c138d089d1418c/1/ef00/0018/1
payload: 1

Any other clusters/attributes payload format

Payload should contain **JSON**: {"type": <zigbee data type>, "data":<data to be sent>}
`type` - a number that represents the type of data being sent.

Zigbee Data Types and Data Type IDs

Data Class	Data Type	Data Type ID
Null	No data Reserved	0x00 0x01—0x07
General Data	8-bit data 16-bit data 24-bit data 32-bit data 40-bit data 48-bit data 56-bit data 64-bit data	0x08 0x09 0x0a 0x0b 0x0c 0x0d 0x0e 0x0f
Logical	Boolean Reserved	0x10 0x11—0x17

Data Class	Data Type	Data Type ID
Bitmap	8-bit data 16-bit data 24-bit data 32-bit data 40-bit data 48-bit data 56-bit data 64-bit data	0x18 0x19 0x1a 0x1b 0x1c 0x1d 0x1e 0x1f
Unsigned integer	Unsigned 8-bit integer Unsigned 16-bit integer Unsigned 24-bit integer Unsigned 32-bit integer Unsigned 40-bit integer Unsigned 48-bit integer Unsigned 56-bit integer Unsigned 64-bit integer	0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27
Signed integer	Signed 8-bit integer Signed 16-bit integer Signed 24-bit integer Signed 32-bit integer Signed 40-bit integer Signed 48-bit integer Signed 56-bit integer Signed 64-bit integer	0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f
Enumeration	8-bit enumeration 16-bit enumeration Reserved	0x30 0x31 0x32—0x37
Floating point	Semi-precision Single precision Double precision Reserved	0x38 0x39 0x3a 0x3b—0x3f
String	Reserved Octet string Character string Long octet string Long character string Reserved	0x40 0x41 0x42 0x43 0x44 0x45—0x47
Ordered sequence	Array Reserved Structure Reserved	0x48 0x49—0x4b 0x4c 0x4d—0x4f
Collection	Set Bag Reserved	0x50 0x51 0x52—0x57
Reserved	-	0x58—0xdf

Data Class	Data Type	Data Type ID
Time	Time of day Date UTC Time Reserved	0xe0 0xe1 0xe2 0xe3—0xe7
Identifier	Cluster ID Attribute ID BACnet ID Reserved	0xe8 0xe9 0xea 0xeb—0xef
Miscellaneous	IEEE Address 128-bit security key Reserved	0xf0 0xf1 0xf2—0xfe
Unknown	Unknown	0xff

`data` - a HEX string of the following format:
- Bytes without spaces, uppercase, multiple of two. Example: `010203FF`
- Bytes with spaces, uppercase, grouped by two. Example: `01 02 03 FF`

Please note that the number of bytes in the payload **strictly** depends on the data type!
For example, if you send data with type 16-bit integer, the number of bytes in the payload should be 2.
Example: `{"type": 33, "data":"0000"}`
33 - is 0x21 converted to DEC

System control topic (IN) (available from v3.3.0)

This topic allows you to control main functions of the Zigbee Hub.

Topic format: `base topic` / `system_control`

Payload format: {"action": "<system control actions>", <additional parameters>}

Permit join action

Allows you to open a ZigBee network to add new devices.

Payload format: {"action": "permit_join", "time": <time to open the network in seconds, from 1 to 254>, "addr": <network address of the device on which to open the network. Can be omitted if the network needs to be opened on all devices>}

Examples:
`{"action": "permit_join", "time": 60}` - open the entire network for 60 seconds
`{"action": "permit_join", "time": 0}` - close network
`{"action": "permit_join", "time": 60, "addr": 64580}` - open the network for 60 seconds only on device with network address 64580 (DEC number, not HEX)

Configure reporting action

Configures reporting for the selected device. If the device is battery-powered, it must be woken up to accept this command.

Payload format: {"action": "configure_reporting", "ieee": <IEEE address of the target device, HEX string>, "ep": <target endpoint, DEC>, "cl": <target cluster, DEC>, "attr": <target attribute, DEC>, "minRep": <minimum time for reporting in seconds, DEC>, "maxRep": <max time for reporting in seconds, DEC>, "dType": <zigbee reporting data type for this attribute, DEC>, "change": <how much the attribute value must change for reporting to occur. Should be omitted for discrete attributes>}

Examples:
`{"action": "configure_reporting", "ieee": "3425b4fffe12e9e9", "ep": 1, "cl": 6, "attr": 0, "minRep": 1, "maxRep": 3600, "dType": 0}` - configure ON/OFF attribute repotring to minimum 1s and max 3600s report time. "change" field is omitted because it is a discrete attribute.
`{"action": "configure_reporting", "ieee": "3425b4fffe12e9e9", "ep": 1, "cl": 2820, "attr": 1285, "minRep": 30, "maxRep": 3600, "dType": 33, "change": 200}` - configure RMS Voltage attribute repotring to minimum 1s and max 3600s report time. Reporting will occur no earlier than 30s if the value has changed by 200 or more (2v) or after a 3600s timeout if value does not changed.

Binding action

Binds the target device to the coordinator or to another device.

> If one device bound to another, the coordinator will stop receiving reports from the bound device cluster.

Payload format: {"action": "binding", "mode": "<binding mode>", "scrIeee": "<IEEE of the device to which the binding request will be sent, HEX string>", ""scrEp": <endpoint number that needs to be bound, DEC number>, "scrCl": <cluster number that needs to be bound, DEC number>, <additional parameters>}

To device mode

Binds one device to another.

Payload format: {"action": "binding", "mode": "to_device", "scrIeee": "<IEEE of the device to which the binding request will be sent, HEX string>", ""scrEp": <endpoint number that needs to be bound, DEC number>, "scrCl": <cluster number that needs to be bound, DEC number>, "dstEp": <endpoint number **to which** the binding will be done, DEC number>, "dstIeee": "<IEEE device to which binding will be performed, HEX string>"}

Example: `{"action": "binding", "mode": "to_device", "scrIeee": "3425b4fffe12e9e9", ""scrEp": 1, "scrCl": 6, "dstEp": 1, "dstIeee": "a4c1389ffb198304"}` - binding a Zigbee button (ON/OFF cluster) to a Zigbee relay.

To coordinator mode

Binds to the coordinator so that it can receive reports from this cluster.

> Zigbee Hub will attempt to bind automatically when interviewing a device, but this does not always work perfectly.

Payload format: {"action": "binding", "mode": "to_device", "scrIeee": "<IEEE of the device to which the binding request will be sent, HEX string>", ""scrEp": <endpoint number that needs to be bound, DEC number>, "scrCl": <cluster number that needs to be bound, DEC number>}

Example: `{"action": "binding", "mode": "to_coordinator", "scrIeee": "3425b4fffe12e9e9", ""scrEp": 1, "scrCl": 6}` - binding a Zigbee button (ON/OFF cluster) to the coordinator.