zencontrol Docs

Dec 17, 2021

Contents

MQTT Documentation

Overview

MQTT integration is designed to allow for flexible and realtime ingestion of DALI ECD and ECG analytics. A controller is configured to point at a MQTT broker with appropriate certification to allow communication between endpoints. To ensure secure communication between endpoints, TLS v1.2 is used to provide encryption throughout data transport.

To enable the MQTT integration module for zencontrol MQTT integration, contact the zencontrol support team for licensing information.

Supported Devices

Below is a list of supported controllers for MQTT integration:

1. Application Controller Pro
2. Field Controller
3. ACx3 Pro

Setup

To setup a control system to use MQTT it's required to setup a device and change from the default parameters. To do such, a user must configure the control system settings through cloud.zencontrol.com.

Cloud

All controller settings are modified through the use of cloud.zencontrol.com.

Changing MQTT Settings

1. Navigate to cloud.zencontrol.com
2. Open your site
3. `Dashboard` -> `Gridview`
4. On the top navigation menu, click on `Add-ons`
5. On the `Add-ons` navigation menu click on `MQTT`
6. Find your controller and modify settings

Adding Keys & Certificates

1. Navigate to cloud.zencontrol.com
2. Open your site
3. `Dashboard` -> `Gridview`
4. On the top navigation menu, click on `Site`
5. On the `Site` navigation menu go to `Key store`
6. On the line that says `+` double click on the `label` cell and a label for your key/certificate
7. Double click the `Upload` cell and upload your site key(s)
    - *Note: Currently only PEM is x509 is supported*
8. You certificate/key is now uploaded to the site Key Store

Server IP & Port

Option	Default	Optional
Server IP	0.0.0.0	Required
Server Port	1883	Required

The target server IP and Port of the MQTT broker.

Note: When using TLS it's recommended to use a port other than 1883, most brokers recommend to use port 8883, however, this will depend on the deployment setup for the target MQTT broker

Username & Password

Option	Default	Optional
Username	No Default	Required
Password	No Default	Required

Usernames and passwords may be required to connect to a server and can be set through the cloud. If unused, any value will suffice to allow for connection.

Modes

Option	Default	Optional
Mode	Normal	Required

There are currently two supported operating modes for MQTT, Normal and single-endpoint. Normal operating mode splits topic up based on the data being reported, whereas Single-Endpoint targets a single topic with payload descriptors.

TLS

Option	Default	Optional
CA	Not Defined	Required
Enable SSL	True	Required

See before modifying this setting *Adding Keys & Certificates* for a guide on how to add keys to your sites key store. Any keys within the key store will be selectable from the cell dropdown.

To enable TLS, a Certificate Authority Certificate must be uploaded and configured on cloud.zencontrol.com. If an invalid Certificate is uploaded, the device will fail to authenticate and will be unable to connect until a valid certificate is uploaded.

Not Recommended: TLS Can be disabled by setting `Enable SSL = False`.

Base Topic

The base topic of the device will publish with the following definition:

```
zencontrol/{schema_version}/{serial}_{EAN}
```

Example:

```
zencontrol/v1/06571626575E_000000000007A6BB
```

Topics

When sending telemetry to the MQTT Broker in normal operation, the control system will follow a publish string standard split into 5 main sections. Single-Endpoint only publishes to the one topic. Much like the control system, the device being described is identified through the topic using their GTIN, Serial and Logical Index. This is done to ensure uniqueness on every topic, and maintain concurrency through multiple sessions where DALI Identifying information could change, such as DALI Address.

ID Example:

```
{device_GTIN}_{device_serial}_{logical_index}
```

In use:

```
zencontrol/v1/06571626575E_000000000007A6BB/ecd/065716265220_0000000000000006_00
```

Controller

Details pertaining to the controller

```
'{base_topic}/{message_type}'
```

```
'zencontrol/v1/06571626575E_000000000007A6BB/profile'
```

Supported Messages:

- Description
- Error
- Profile

Group

Information regarding the status of groups

```
'{base_topic}/group/{group_id}/{message_type}'
```

```
'zencontrol/v1/06571626575E_000000000007A6BB/group/0/'
```

Supported Messages:

- Group Description
- Level
- Occupied
- Error

ECG

Information/details pertaining to Electronic Control Gears

```
'{base_topic}/ecg/{id}{message_type}'
```

```
'zencontrol/v1/06571626575E_000000000007A6BB/ecg/012345678910_0000000000000001_00/'
```

Supported Messages:

- Description
- Level Configuration

- Level
- Scene
- Group
- Fade
- Error

ECD

Information/details pertaining to Electronic Control Devices

```
'{base_topic}/ecd/{id}/{message_type}'
```

```
'zencontrol/v1/06571626575E_000000000007A6BB/ecd/012345678910_0000000000000001_00/'
```

Supported Messages:

- Description
- Lux
- Lux Configuration
- Motion Configuration
- Motion
- Absolute Button
- Button
- Error

Emergency

Information regarding the status of emergency fittings

```
'{base_topic}/em/{id}/{message_type}'
```

```
'zencontrol/v1/06571626575E_000000000007A6BB/em/012345678910_0000000000000001_00/'
```

Messages

- Non Specific telemetry information from the controller*

```
'{base_topic}/messages/{message_type}'
```

```
'zencontrol/v1/06571626575E_000000000007A6BB/messages/outbound'
```

Single-Endpoint

Single-Endpoint publish topic.

```
'{base_topic}/messages/outbound'
```

```
'zencontrol/v1/06571626575E_000000000007A6BB/messages/outbound'
```

Outbound Messages

Telemetry sent to the MQTT broker is sent using JSON. All messages will contain a session ID that changes on every initial connection to the broker. Below is an example JSON block sent to the broker.

```
{
    "session_id": 123,

    "{variable_1}": "xyz",
    "{variable_2}": "abc",

    ...
}
```

Payloads

Description

The description message type is used to describe the device currently sitting on the topic.

Payload Topic *Controller*

`{base_topic}`

ECG, ECD, Emergency

`{base_topic}/ecg/{device}`

`{base_topic}/ecd/{device}`

`{base_topic}/emergency/{device}`

Payload JSON

```
{
    "session_id": {session_id: int},

    "id": {device_id: string},
    "label": {device_label: string},
    "type": {device_type: int},
    "dali_address": {device_dali_address: int},
    "serial_number": [{device_serial_number: string}],
    "firmware_v_maj": {device_firmware_major_version: int},
    "firmware_v_min": {device_firmware_minor_version: int},
    "device_id": {device_id: int},
    "EAN": [{device_EAN: string}]
}
```

Payload Variables

Name	Type	Description
id	string	Device ID
label	string	Current Device Label
type	string	Device Type
dali_address	int	Current DALI Address of the device
serial_number	string	Hex string of serial number
firmware_v_maj	int	Device Firmware Major Version
firmware_v_min	int	Device Firmware Minor Version
device_id	int	Device ID for logical index
EAN	string	Hex string of ean

Payload Definitions

Variable	Value	Meaning
type	0	Control Gear
type	1	Control Device

Level Configuration

Provides information to describe current level settings of the device

Payload Topic

```
{base_topic}/ecg/{device}/level
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "max": {arc_max: int},
    "min": {arc_min: int},
    "last_heard": {last_heard_scene: int}
}
```

Payload Values

Name	Type	Description
max	int	Max ARC Level
min	int	Min ARC Level
last_heard	int	Last heard scene

Level

Provides information to describe the current lighting level of the device.

Payload Topic

```
{base_topic}/ecg/{device}/level
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "arc": {arc: int}
}
```

Payload Values

Name	Type	Description
arc	int	Current arc level

Scene

Provides information to describe the scene settings of device.

Payload Topic

```
{base_topic}/ecg/{device}/scene
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "scenes": [
        {
            "number": {scene_number},
            "arc": {scene_arc},
            "temp": {scene_temperature}
        },

        {
            "number": {scene_number},
            "arc": {scene_arc},
            "temp": {scene_temperature}
        },

        ...
    ]
}
```

Payload Values

Name	Type	Description
scenes	list	List of scene information
scenes[n].number	int	Scene Number
scenes[n].arc	int	Scene ARC level
scenes[n].Temperature	int	Scene temperature value

Group

Provides group membership information of device

Payload Topic

`{base_topic}/ecg/{device}/group`

Payload JSON

```
{
    "session_id": {session_id: int},
    "membership": [{group_0}, {group_1}, {group_[n]}, ... {group_15}]
}
```

Payload Values

Name	Type	Description
`membership`	list	List of group membership status
`membership[n]`	int	Current status of membership for group (0 or 1)

Fade

Information used to describe a DALI-2 Fade.

Payload Topic

`{base_topic}/ecg/{device}/fade`

Payload JSON

```
    {
        "session_id": {session_id: int},

        "fade_time": {fade_time: int},
        "fade_rate": {fade_rate: int},
        "number_of_steps": {number_of_steps: int},
        "dimming_curve": {dimming_curve_type: int}
    }
```

Payload Variables

Name	Type	Description
`fade_time`	int	DALI-2 Fade Time
`fade_rate`	int	DALI-2 Fade Rate
`number_of_steps`	int	DALI-2 Fade Steps
`dimming_curve`	int	Current dimming curve in use

Payload Definitions

Variable	Value	Meaning
`Dimming Curve`	0	Linear Curve
`Dimming Curve`	1	Log Curve

Error

Provides information to describe an error state reported by the control module or device being reported on.

Payload Topic The payload topic for errors is slightly different to most other topics. As each error is an individual point, the error message is used as part of the topic string.

```
{base_topic}/{type}/{device}/error/{error_message}/
```

Error Type	Topic	Description (When error_value set)
Short Address Mask	“short_address_is_mask”	Device has MASK (0xFF) as short address
Received Trash	“received_trash”	Controller has received trash
Communication Error	“communication_error”	Device is in communication error
No Space	“no_space”	No space for device
Device Not Recognised	“device_not_recognised”	Device can’t be recognised on controller
Device Not Supported	“device_not_supported”	Device isn’t supported on controller
Battery Duration Failure	“battery_duration_failure”	Device has failed duration test
Battery Failure	“battery_failure”	Battery Failure Status
Emergency Lamp Failure	“emergency_lamp_failure”	Device has reported emergency lamp failure
Function Test Delay Exceeded	“function_test_delay_exceeded”	DALI test delay has been exceeded (device couldn’t start function test)
Duration Test Delay Exceeded	“duration_test_delay_exceeded”	DALI test delay has been exceeded (device couldn’t start duration test)
Function Test Failure	“function_test_failure”	Device failed function test
Duration Test Failure	“duration_test_failure”	Device failed duration test
Input Device Error	“input_device_error”	Generic control device input error
Internal Error	“internal_error”	Generic Internal error
ECG Failure	“ecg_failure”	Control Gear has reported failure
Switch Stuck	“switch_stuck”	Device is continuously reporting switch status
Lamp Failure	“lamp_failure”	Device is reporting lamp failure

examples:

```
{base_topic}/ecg/{device}/error/communication_error/
```

```
{base_topic}/ecd/{device}/error/communication_error/
```

```
{base_topic}/emergency/{device}/error/battery_duration_failure/
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "error_type": {error_type: int},
    "error_message": {error_message: int},
    "error_value": {error_value: int}
}
```

Payload Values

Name	Type	Description
error_type	int	Raw error type value
error_message	string	Error Message
error_value	int	Error value reported (Usually 0 for cleared or 1 for set)

Payload Definitions

Variable	Value	Meaning
error_type	0	No Description
error_type	1	Short Address Is Mask
error_type	2	Received Trash
error_type	3	Communication Error
error_type	4	No Space
error_type	5	Device Not Recognised
error_type	6	Device Not Supported
error_type	7	Battery Duration Failure
error_type	8	Battery Failure
error_type	9	Emergency Lamp Failure
error_type	10	Function Test Delay Exceeded
error_type	11	Duration Test Delay Exceeded
error_type	12	Function Test Failure
error_type	13	Duration Test Failure
error_type	14	Input Device Error
error_type	15	Application Controller Error
error_type	16	Control Gear Failure
error_type	17	Switch Stuck
error_type	18	Lamp Failure

Lux Configuration

Provides information to describe the current lux configuration of the device.

Payload Topic

```
{base_topic}/ecd/{device}/lux
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "measure_period": {measurement_period: int},
    "update_interval": {update_interval: int},
    "units": {units: string},

}
```

Payload Values

Name	Type	Description
measurement_period	int	Measurement interval of device
update_interval	int	Reported update interval of device (0 for on change)
units	string	Units reported

Lux

Provides information to describe the current lux level of the device.

Payload Topic

```
{base_topic}/ecd/{device}/lux/value
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "value": {lux_level: int},
    "instance": {instance_number: int}
}
```

Payload Values

Name	Type	Description
value	int	Current lux level
instance	int	Instance number reported (if available)

Motion Configuration

Provides information to describe the current motion/occupancy configuration of the device.

Payload Topic

```
{base_topic}/ecd/{device}/motion
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "measure_period": {measurement_period: int},
    "update_interval": {update_interval: int},
    "units": {units: string},

}
```

Payload Values

Name	Type	Description
measurement_period	int	Measurement interval of device
update_interval	int	Reported update interval of device (0 for on change)
units	string	Units reported

Motion

Provides information to describe the current motion status (occupancy) the device.

Payload Topic

```
{base_topic}/ecd/{device}/motion/value
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "value": {motion_value: int},
    "instance": {instance_number: int}

}
```

Payload Values

Name	Type	Description
value	int	Current motion/occupancy status
instance	int	Instance number reported (if available)

Button

Provides information to describe the current button press state of the device.

Payload Topic

```
{base_topic}/ecd/{device}/button
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "press_type": {press_type: int},
    "instance": {instance_number: int}
}
```

Payload Values

Name	Type	Description
press_type	int	Press Type
instance	int	Instance number reported (if available)

Payload Definitions

Variable	Value	Meaning
press_type	1	Short Press
press_type	2	Long Press

Absolute Button

Provides information to describe the current button press state of the device.

Payload Topic

```
{base_topic}/ecd/{device}/absolute_button
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "abs_value": {press_type: int},
    "instance": {instance_number: int}
}
```

Payload Values

Name	Type	Description
abs_value	int	Absolute button value
instance	int	Instance number reported (if available)

Group Description

Provides information to describe a group.

Payload Topic

```
{base_topic}/group/{group_id}
```

Payload JSON

```
{
    "session_id": {session_id: int},

    "label": {group_label: int},
    "id": {group_id: int}
}
```

Payload Values

Name	Type	Description
label	string	Group label
id	int	Group ID

Beacon Transmitter Configuration

Provides information to describe beacon configuration for device. Payloads frames are separated into iBeacon and Eddystone frames depending on the beacon type.

Version Requirement

```
>= v1.7.7
```

Payload Topic

```
{base_topic}/beacon/{device_id}/transmitter/{beacon_id}
```

iBeacon Payload

```
{
    "session_id": {session_id: int},

    "type": {beacon_type: int},
    "interval": {transmit_interval: int},
    "tx_power": {tx_power: int},
    "ibeacon_frame": {
        "proximity_uuid": {ibeacon_uuid: int list [0:15]},
        "major": {ibeacon_major: int list [0:1]},
        "minor": {ibeacon_minor: int list [0:1]}
    }
}
```

Eddystone Payload

```
{
    "session_id": {session_id: int},

    "type": {beacon_type: int},
    "interval": {transmit_interval: int},
    "tx_power": {tx_power: int},
    "eddystone_frame": {
        "namespace": {namespace: int list [0:9]},
        "instance": {instance: int list [0:5]},
    }
}
```

Payload Values

Name	Type	Description
type	int	Type of beacon
interval	int	Beacon interval in ms
tx_power	int	Transmit power level of beacon (Gain)
eddystone_frame.namespace	list	Byte list for Eddystone namespace (10 bytes)
eddystone_frame.instance	list	Byte list for Eddystone instance (6 bytes)
ibeacon_frame.proximity_uuid	list	Byte list for iBeacon proximity UUID (16 bytes)
ibeacon_frame.major	list	Byte list for iBeacon major (2 bytes)
ibeacon_frame.minor	list	Byte list for iBeacon minor (2 bytes)

Payload Definitions

Variable	Value	Meaning
type	0	Beacon Disabled
type	1	Beacon zencontrol Discoverable
type	2	iBeacon
type	3	Eddystone
tx_power	0	0dBm Gain
tx_power	1	0dBm Gain
tx_power	2	3dBm Gain
tx_power	3	3dBm Gain
tx_power	4	4dBm Gain
tx_power	5	4dBm Gain
tx_power	6	4dBm Gain
tx_power	7	4dBm Gain
tx_power	8	4dBm Gain

Devicebound Messages

Telemetry can be sent to the control module publishing to the devicebound topic. On Successful ingestion of a command, a command response will be published with a response for the given commands. Some commands have optional values, that can be omitted, furthermore, sanitisation of min and max values will be applied to certain values.

Devicebound Topic:

```
{base_topic}/messages/devicebound
```

Devicebound Payloads

Each message is categorised by the target, type, ID and request ID and may contain multiple messages to the same target. The request ID is a random value sent to the device to identify messages sent, the same request ID is sent on a message response.

Example Payload:

```
{
    "id": {target_device:int},
    "rid": {request_id:int},
    "target": {target_type:int},

    "{command_1}": {command_payload:JSON Object},
    "{command_n}": {command_payload:JSON Object},
    ...
}
```

For ECD and ECG commands, the "id" refers to the DALI Address of the device. For Beacon commands, the "id" refers to the device_id of the target beacon as beacons can be shared across multiple logical indexes.

Payload Definitions:

Variable	Value	Meaning
target	0	Control Module
target	1	Emergency
target	2	ECG
target	3	ECG Group
target	4	ECD
target	5	ECD Group
target	6	Beacon

Level

Changes the current ARC and or Temperature level values of device(s). Values committed are ignored in the command, for example, if temperature isn't set, no temperature change is made. The level command conforms to DALI-2 Standards, therefore, if a temperature and arc change with a fade is set, only the arc level is faded.

Payload JSON

```
{
    "level": {
        "arc": {arc_level:int},
        "fade_time": {fade_time_ms:int},
        "temp": {temperature_kelvin:int}
```

(continues on next page)

(continued from previous page)

```
    }
}
```

Payload Values

Name	Type	Description	Required	Min/Max
arc	int	Target ARC level	False	0/255
fade_time	int	Target fade time in milliseconds	False	1000/65000
temp	int	Target temperature in kelvin	False	1-65000

Membership

Changes the current membership status of target device(s).

Payload JSON

```
{
    "membership": {
        "scenes": [{membership_0:int},{membership_1:int}, ... {membership_15:int}],
        "groups": [{membership_0:int},{membership_1:int}, ... {membership_15:int}]
    }
}
```

Payload Values

Name	Type	Description	Required	Min/Max
groups[n]	int	Group Status	True	0/1
scenes[n]	int	Membership Status	True	0/1

Beacon Config

Changes beacon configuration for a device. Type is automatically changed when the appropriate inner frame is sent.

Version Requirement

```
>= v1.7.7
```

iBeacon Payload JSON

```
{
    "beacon_config": {
        "position": {beacon_position: int},
        "interval": {interval: int},
        "tx_power": {tx_power: int},
        "ibeacon": {
            "proximity_uuid": {ibeacon_uuid: int list [0:15]},
            "major": {ibeacon_major: int list [0:1]},
            "minor": {ibeacon_minor: int list [0:1]}
        }
    }
}
```

Eddystone Payload JSON

```
{
    "beacon_config": {
        "position": {beacon_position: int},
        "interval": {interval: int},
        "tx_power": {tx_power: int},
        "eddystone": {
            "namespace": {namespace: int list [0:9]},
            "instance": {instance: int list [0:5]},
        }
    }
}
```

Payload Values

Name	Type	Description	Required	Min/Max
`position`	int	Beacon index (Beacon ID)	True	0/Number of beacons supported on device
`interval`	int	Transmit interval in ms	True	0/Max transmit time time for beacon type
`tx_power`	int	Transmit power (gain)	True	0/8
`eddystone.namespace`	list	Byte list for Eddystone namespace (10 bytes)	True (if Eddystone)	0/255 per index
`eddystone.instance`	list	Byte list for Eddystone instance (6 bytes)	True (if Eddystone)	0/255 per index
`ibeacon.proximity_uuid`	list	Byte list for iBeacon proxmity UUID (16 bytes)	True (if iBeacon)	0/255 per index
`ibeacon.major`	list	Byte list for iBeacon major (2 bytes)	True (if iBeacon)	0/255 per index
`ibeacon.minor`	list	Byte list for iBeacon minor (2 bytes)	True (if iBeacon)	0/255 per index

Payload Definitions

Variable	Value	Meaning
`type`	0	Beacon Disabled
`type`	1	Beacon zencontrol Discoverable
`type`	2	iBeacon
`type`	3	Eddystone
`tx_power`	0	0dBm Gain
`tx_power`	1	0dBm Gain
`tx_power`	2	3dBm Gain
`tx_power`	3	3dBm Gain
`tx_power`	4	4dBm Gain
`tx_power`	5	4dBm Gain
`tx_power`	6	4dBm Gain
`tx_power`	7	4dBm Gain
`tx_power`	8	4dBm Gain