Macrotech Lock Controllers

Use cases and Message Payloads

Last updated on: 11 Feb 2022

Table of Contents

[**Introduction**](#_39p6cl3ktedd) **3**

[**Messages - Packet Structures and JSON Payload**](#_fxgs24v28ydn) **3**

[**Message Data Types**](#_dt4z0jwzctcb) **4**

[**Device Id - Mapping**](#_pf0dq7xwksic) **10**

[**Use cases**](#_3phfkf2mvbwx) **10**

[Get Status - All Locks](#_j06ch9v8pba4) 10

[JSON Payload - Cloud IoT Server to IoT Gateway](#_cntzz7fwgx73) 10

[PDU Structure - IoT Gateway to Lock controller nodes](#_68a804q81vey) 11

[Response for Get Status - All Locks](#_y8o93fp8dl7j) 11

[Get Status - Individual Lock](#_sujl27bi7b86) 11

[JSON Payload - Cloud IoT Server to IoT Gateway](#_l379482sb34g) 11

[PDU Structure - IoT Gateway to a specific Lock controller node](#_wb3evhharqgh) 11

[Response for Get Status - Individual Lock](#_u0n591h1njzo) 12

[PDU Structure - Lock controller node to IoT Gateway](#_ck4xwk76eej) 12

[JSON Payload - IoT Gateway to Cloud IoT Server](#_4a8d5dudq0g) 13

[Unlock Door - Normal Operation](#_bbveaii1nn1t) 13

[JSON Payload - Cloud IoT Server to IoT Gateway](#_7ghm4j11fp8b) 13

[PDU Structure - IoT Gateway to a specific Lock controller node](#_zi4ev4dfgu8n) 13

[Response for Unlock Door - Normal Operation](#_6vuicsvk4hjn) 14

[PDU Structure - Lock controller node to IoT Gateway](#_w4q3pdlfx4kg) 14

[JSON Payload - IoT Gateway to Cloud IoT Server](#_r7e7tsloheh5) 14

[Unlock Door - Emergency Operation](#_80gu1mt8ta3j) 14

[JSON Payload - Cloud IoT Server to IoT Gateway](#_hn4cwadtd4t4) 15

[PDU Structure - IoT Gateway to a specific Lock controller node](#_3jroj0g28r7v) 15

[Response for Unlock Door - Emergency Operation](#_azac3sx39pca) 15

[PDU Structure - Lock controller node to IoT Gateway](#_jwyzc5fm4o8g) 15

[JSON Payload - IoT Gateway to Cloud IoT Server](#_dt0hou47x9gc) 15

[Heartbeat - Periodic Telemetry Message](#_je5zlljx66dx) 16

[PDU Structure - Lock controller node to IoT Gateway](#_546e1y1vcbfy) 16

[JSON Payload - IoT Gateway to Cloud IoT Server](#_o2ttv01k3imv) 16

[Status Update Message](#_x4z1spnzm3kz) 17

[PDU Structure - Lock controller node to IoT Gateway](#_y9uqn4m7gsvq) 17

[JSON Payload - IoT Gateway to Cloud IoT Server](#_uheawern1cbw) 17

[Alarms - Low-Battery and Deadbolt Alerts](#_3j9s0caoohp9) 17

[PDU Structure - Lock controller node to IoT Gateway](#_sj3spqaw72of) 17

[JSON Payload - IoT Gateway to Cloud IoT Server](#_5pvgu6k86gob) 18

[**Appendix A - Wirepas Node Address and Device Id mapping CSV file**](#_4zqfbezibtzq) **18**

# Introduction

This document lists various use cases pertaining to managing the RF network of Macrotech’s Lock controllers along with the corresponding message exchanging payload structures between:

* Cloud IoT Platform (Macrotech’s CloudExt) <-> IoT Gateway
* IoT Gateway <-> Wirepas Sink node <-> Lock controller nodes

# Messages - Packet Structures and JSON Payload

Between Cloud IoT Platform and IoT Gateway, the messages are exchanged in JSON payload using MQTT protocol.

Between IoT Gateway and Lock controller nodes, the messages are exchanged as encoded packets of bytes via Wirepas Sink node. The structure of this encoded packets is given below:

| **Byte:Length** | **Field Name** | **Description** |
| --- | --- | --- |
| 0:1 | Beginning Marker | Beginning of Packet Marker. Should be 0xAA. |
| 1:1 | Message Type | First 2 Most Significant Bits (MSBs) specify destination address type:  01 - Unicast  10 - Multicast  11 - Broadcast  Next 3 bits specify the operation type:  000 - Get  001 - Set  010 - Notification  011 - Acknowledgement  Last 3 bits specify the type of data:  000 - Attribute  001 - State  010 - Telemetry  011 - Alarm |
| 2:4 or 1 or 0 | Source/Destination Address | * For messages sent from the individual node, the source address should be 4 bytes containing Wirepas node address of that individual node. * For messages sent from the sink node to an individual node, the destination address should be 4 bytes Wirepas node address of the target node. * For multicast messages sent from the sink node, this field is 1 byte containing the group number. * For broadcast messages sent from the sink node, this field should be absent (0 bytes). |
| 6 or 3 or 2:4 | Request ID | 4 bytes signed int used as the request identifier. The nodes should process the request id field in the following way:   * When a Get or Set type of packet is received, this request id should be sent back when sending the response for Get operation (Notification) or Set operation (Acknowledgement). This is very important as the gateway uses the request id to map the corresponding response it sends to Cloud IoT Server. * While sending Telemetry or Alarm messages, the request id should be -1. |
| 10 or 7 or 6:1 (optional) | Key and Length | First 4 Most Significant Bits (MSBs) specify the key of the data element.  With 4 bits, there can be 15 keys for each data type (Attribute, State, Telemetry and Alarm). Note that only 0x1 to 0xF are used (0x0 is not used).  Last 4 bits specify the length of the data element. With 4 bits, the data length can vary upto a maximum of 15 bytes. If no value is specified, then 0x0 is used. For example, in Get messages, only the key field is passed without any value. |
| 11 or 8 or 7:x | Value | If the data element’s length in the previous byte is non-zero, then the corresponding value is captured in N number of bytes, where N = Length as specified in the previous byte. |
| x:1 | End Marker | End of Packet Marker. Should be 0x00 (for this reason, 0x0 is not used for the key).  The packet may contain more than 1 data element and the end marker is used to mark the end of Key-Length-Value tuples. |

# Message Data Types

The following tables provide the list of supported lock controller data types (Attribute, State, Alarm and Telemetry) in Lock controller nodes (Wirepas RF mesh) and Cloud IoT Server (CloudExt IoT Platform).

| **Attribute** | | | |
| --- | --- | --- | --- |
| Device ID | Wirepas specific data type | Key | **0x10** |
| Data type | **Byte** |
| Size | **6** |
| Type | **Read-only** |
| IoT Server specific data type | ***macaddr*** - String in the form of xx-xx-xx-xx-xx-xx | |
| This field provides mapping of Device to it's hardware ID. Useful to map server side name to device side ID. Last 3 bytes of this ID are used by Wirepas as a Node address. | | |
| Firmware Version | Wirepas specific data type | Key | **0x20** |
| Data type | **Byte** |
| Size | **2** |
| Type | **Read-only** |
| IoT Server specific data type | ***verstr*** - String in the form {major}.{minor}.{maint}.  For example: 1.4.21, 1.0.00 | |
| Wirepas side:  1st byte contains 4 bits major version no. and 4 bits minor version number, and 2nd byte contains maintenance version number. Example 0x10 0x01 means version 1.0.1 | | |
| Location | Wirepas specific data type | Key | **0x30** |
| Data type | **String** |
| Size | **15** |
| Type | **Read Write** |
| IoT Server specific data type | String.  For example: “F1-101-Lock” | |
| This field contains free form text upto 15 characters long. It is used for uniquely naming the device to identify it’s geographical location. | | |
| Telemetry Sampling Interval | Wirepas specific data type | Key | **0x40** |
| Data type | **Short Integer** |
| Size | **2** |
| Type | **Read Write** |
| IoT Server specific data type | Integer | |
| Represents the telemetry sampling interval (periodicity) in seconds. For disabling telemetry, the value should be set 0. | | |

| **State** | | | |
| --- | --- | --- | --- |
| lock | Wirepas specific data type | Key | **0x10** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read Write** |
| IoT Server specific data type | Boolean (true/false) | |
| Used for activating or deactivating the lock when the deadbolt is disengaged. If 'true', activates the lock. For deactivating the lock, the value should be 'false'.   * For activating the lock, the byte value should be 0x01, which is equivalent to Boolean true. * For deactivating the lock, the byte value should be 0x00, which is equivalent to Boolean false. * As of now, only deactivation of lock is supported for “Set” operation. | | |
| emergency-lock | Wirepas specific data type | Key | **0x20** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read Write** |
| IoT Server specific data type | Boolean (true/false) | |
| Used for activating or deactivating the lock in emergency situations (when the deadbolt is engaged). If 'true', activates the lock. For deactivating the lock, the value should be 'false'.   * For activating the lock, the byte value should be 0x01, which is equivalent to Boolean true. * For deactivating the lock, the byte value should be 0x00, which is equivalent to Boolean false. * As of now, only deactivation of lock is supported for “Set” operation. | | |
| status | Wirepas specific data type | Key | **0x30** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | ***status*** - String (“online” or “offline”) | |
| Represents the current latch status.   * If closed, the value of the byte is going to be 0x01, which is equivalent to “online”. * If opened, the value of the byte is going to be 0x00, which is equivalent to “offline”. | | |
| battery | Wirepas specific data type | Key | **0x40** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | ***char*** - Integer with a value in the range 0-100 | |
| Represents the current charge level of the battery in percentage. The value ranges from 0 to 100. | | |
| deadbolt | Wirepas specific data type | Key | **0x50** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | Boolean (true/false) | |
| Used for providing the current status of the deadbolt.   * If the deadbolt is engaged, the value of the byte is going to be 0x01, which is equivalent to Boolean true * If the deadbolt is disengaged, the value of the byte is going to be 0x00, which is equivalent to Boolean false | | |
| latch | Wirepas specific data type | Key | **0x60** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | Boolean (true/false) | |
| Used for providing the current status of the latch..   * If the latch is closed, the value of the byte is going to be 0x01, which is equivalent to Boolean true * If the latch is opened, the value of the byte is going to be 0x00, which is equivalent to Boolean false | | |

| **Alarm** | | | |
| --- | --- | --- | --- |
| low-battery | Wirepas specific data type | Key | **0x10** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | Boolean (true) | |
| This alarm is triggered when the battery level drops below 10%. When the alarm is generated, it’s value is always set to 0x01, which is equivalent to Boolean true | | |
| deadbolt-alert | Wirepas specific data type | Key | **0x20** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | Boolean (true/false) | |
| This alarm is triggered when the deadbolt is engaged or disengaged. When the deadbolt is engaged, the corresponding value is set to 0x01 (Boolean true) and when the deadbolt is disengaged, the value is set to 0x00 (Boolean false) | | |
| latch-alert | **Wirepas specific data type** | Key | **0x30** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | Boolean (true/false) | |
| This alarm is triggered when the latch is opened and closed. When the latch is closed, the corresponding value is set to 0x01 (Boolean true) and when the latch is opened, the value is set to 0x00 (Boolean false). | | |

| **Telemetry** | | | |
| --- | --- | --- | --- |
| status | Wirepas specific data type | Key | **0x10** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | ***status*** - String (“online” or “offline”) | |
| Represents the current lock status, which represents the latch status.   * If the latch is closed, the value of the byte is going to be 0x01, which is equivalent to “online”. * If the latch is opened, the value of the byte is going to be 0x00, which is equivalent to “offline”. | | |
| battery | Wirepas specific data type | Key | **0x20** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | ***char*** - Integer with a value in the range 0-100 | |
| * Represents the current charge level of the battery in percentage. The value ranges from 0 to 100. | | |
| deadbolt | Wirepas specific data type | Key | **0x30** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | Boolean (true/false) | |
| Represents the current deadbolt status.   * If the deadbolt is engaged, the value of the byte is going to be 0x01, which is equivalent to Boolean true. * If the deadbolt is disengaged, the value of the byte is going to be 0x00, which is equivalent to Boolean false. | | |
| latch | Wirepas specific data type | Key | **0x40** |
| Data type | **Byte** |
| Size | **1** |
| Type | **Read-only** |
| IoT Server specific data type | Boolean (true/false) | |
| Represents the current latch status.   * If the latch is closed, the value of the byte is going to be 0x01, which is equivalent to Boolean true. * If the latch is opened, the value of the byte is going to be 0x00, which is equivalent to Boolean false. | | |

# Device Id - Mapping

Each lock controller node is mapped with a unique identifier. In Wirepas RF mesh network, the lock controller node uses *Wirepas node address*, which is a 6 byte integer value.

In CloudExt IoT Platform, each controller node is identified with a human-readable string instead of *Wirepas node address*.

The IoT Gateway performs the device id translation (Wirepas node address to CloudExt human-readable string and vice-versa) while exchanging the messages between Wirepas RF mesh network and Cloud IoT Platform.

So, it is important to provide the mapping file (CSV) containing Wirepas node address mapped to CloudExt human-readable string before deploying/starting IoT Gateway.

See [Appendix A](#_4zqfbezibtzq) for an example mapping CSV file.

# Use cases

**Note:**

* Device Id mentioned in MQTT topics represents the human-readable string used in CloudExt IoT Platform to represent a lock controller node.
* Node Address mentioned in PDU Structure represents the wirepas node address of the corresponding lock controller node in Wirepas RF mesh network.







## Get Status - Individual Lock

Cloud IoT Server initiates this operation by sending a command to get the status of a specific lock in the Wirepas RF mesh network to which the gateway is connected.

### JSON Payload - Cloud IoT Server to IoT Gateway

| MQTT Topic | {Device Id}/CMD |
| --- | --- |
| JSON Payload | { “Command” : “getstatus” } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the status should be fetched from that lock |



### PDU Structure - IoT Gateway to a specific Lock controller node

The following PDU is sent to Lock controller nodes for Get-State Endpoint (EP), which is **0x01**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. Signed int (4 bytes) |
| 5:1 | 0x30 | Key = status, Length = 0 |
| 6:1 | 0x40 | Key = battery, Length = 0 |
| 7:1 | 0x50 | Key = deadbolt, Length = 0 |
| 8:1 | 0x60 | Key = latch, Length = 0 |
| 9:1 | 0x00 | End of Packet Marker |

## Response for Get Status - Individual Lock

### PDU Structure - Lock controller node to IoT Gateway

The following PDU is sent from the Lock controller node on Notification-State Endpoint (EP) for the destination, which is **0x11**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. Signed int (4 bytes). This request id must be the request id that was sent in the *Get Status* request message. See sections [Get Status - All Locks](#_j06ch9v8pba4) and [Get Status - Individual Lock](#_rjwp9reavl9e) for additional information. |
| 5:1 | 0x31 | Key = status, Length = 1 |
| 6:1 | 0x0 or 0x1 | status. 0x0 means offline and 0x1 means online. |
| 7:1 | 0x41 | Key = battery, Length = 1 |
| 8:1 | 0x0 to 0x64 | Battery level in percentage. 0 (0x) to 100 (0x64). |
| 9:1 | 0x51 | Key = deadbolt, Length = 1 |
| 10:1 | 0x0 or 0x1 | Deadbolt status. 0x0 means disengaged and 0x1 means engaged. |
| 11:1 | 0x61 | Key = latch, Length = 1 |
| 12:1 | 0x0 or 0x1 | Latch status. 0x0 means opened and 0x1 means closed. |
| 13:1 | 0x00 | End of Packet Marker |

### JSON Payload - IoT Gateway to Cloud IoT Server

The following MQTT response message is sent from IoT Gateway to Cloud IoT Server for Get Status request:

| MQTT Topic | {Device Id}/CMD\_RESP |
| --- | --- |
| JSON Payload | {  “Command” : “getstatus”,  “Response” : {  “status” : “online”,  “battery” : 85,  “deadbolt” : false,  “latch” : false  }  } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the response corresponds to that lock.  The JSON payload contains two fields: command and response.  The command field contains the name of the command for which the response is associated.  The response object contains multiple fields:   * status - can be either “online” or “offline” indicating the lock status * battery - provides the battery level in % ranging between 0 and 100. * deadbolt - boolean value of true (if engaged) or false (if disengaged) * latch - boolean value of true (if closed) or false (if opened) |



## Unlock Door - Normal Operation

Cloud IoT Server initiates this operation of unlocking a specific door, where the corresponding deadbolt is already disengaged (normal state).

### JSON Payload - Cloud IoT Server to IoT Gateway

| MQTT Topic | {Device Id}/CMD |
| --- | --- |
| JSON Payload | { “Command” : “unlock” } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the corresponding door should be unlocked. |



### PDU Structure - IoT Gateway to a specific Lock controller node

The following PDU is sent to Lock controller node for Set-State Endpoint (EP), which is **0x09**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. Signed int (4 bytes) |
| 5:1 | 0x11 | Key = lock, Length = 1 |
| 6:1 | 0x00 | 0x0 means unlock the door. |
| 7:1 | 0x00 | End of Packet Marker |

## Response for Unlock Door - Normal Operation

### PDU Structure - Lock controller node to IoT Gateway

The lock controller node sends the following response for [*Unlock Door - Normal Operation*](#_bbveaii1nn1t) request on Acknowledgement-State Endpoint (EP) for the destination, which is **0x19**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. Signed int (4 bytes). This request id must be the request id that was sent in the [Unlock Door - Normal Operation](#_bbveaii1nn1t) request message. |
| 5:1 | 0x11 | Key = lock, Length = 1 |
| 61 | 0x0 or 0x1 | Error code. 0x0 means a successful operation. 0x1 indicates a failed unlocking operation. |
| 7:1 | 0x00 | End of Packet Marker. |

### JSON Payload - IoT Gateway to Cloud IoT Server

The following MQTT response message is sent from IoT Gateway to Cloud IoT Server for [*Unlock Door - Normal Operation*](#_bbveaii1nn1t) request:

| MQTT Topic | {Device Id}/CMD\_RESP |
| --- | --- |
| JSON Payload | {  “Command” : “unlock”,  “Response” : {  “status” : “success”  }  } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the response corresponds to that lock.  The JSON payload contains two fields: command and response.  The response object contains a status field, which can be either “success” or “failed”. |



## Unlock Door - Emergency Operation

Cloud IoT Server initiates this operation of unlocking a specific door, where the corresponding deadbolt is engaged (emergency state).

### JSON Payload - Cloud IoT Server to IoT Gateway

| MQTT Topic | {Device Id}/CMD |
| --- | --- |
| JSON Payload | { “Command” : “emunlock” } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the corresponding door should be unlocked under emergency. |



### PDU Structure - IoT Gateway to a specific Lock controller node

The following PDU is sent to Lock controller node for Set-State Endpoint (EP), which is **0x09**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. Signed int (4 bytes) |
| 5:1 | 0x21 | Key = emergency-lock, Length = 1 |
| 6:1 | 0x00 | 0x0 means unlock the door. |
| 7:1 | 0x00 | End of Packet Marker |

## Response for Unlock Door - Emergency Operation

### PDU Structure - Lock controller node to IoT Gateway

The lock controller node sends the following response for [*Unlock Door - Emergency Operation*](#_80gu1mt8ta3j) request on Acknowledgement-State Endpoint (EP) for the destination, which is **0x19**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. Signed int (4 bytes). This request id must be the request id that was sent in the [Unlock Door - Emergency Operation](#_80gu1mt8ta3j) request message. |
| 5:1 | 0x21 | Key = emergency-lock, Length = 1 |
| 6:1 | 0x0 or 0x1 | Error code. 0x0 means a successful operation. 0x1 indicates a failed emergency unlocking operation. |
| 7:1 | 0x00 | End of Packet Marker. |

### JSON Payload - IoT Gateway to Cloud IoT Server

The following MQTT response message is sent from IoT Gateway to Cloud IoT Server for [*Unlock Door - Emergency Operation*](#_80gu1mt8ta3j) request:

| MQTT Topic | {Device Id}/CMD\_RESP |
| --- | --- |
| JSON Payload | {  “Command” : “emunlock”,  “Response” : {  “status” : “success”  }  } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the response corresponds to that lock.  The JSON payload contains two fields: command and response.  The response object contains a status field, which can be either “success” or “failed”. |



## Heartbeat - Periodic Telemetry Message

All lock controllers in the wirepas RF mesh network periodically (frequency is configurable) send out the heartbeat message, which is sent to Cloud IoT Server for telemetry purpose.

### PDU Structure - Lock controller node to IoT Gateway

The lock controller node sends the following heartbeat message to IoT Gateway on Notification-Telemetry Endpoint (EP) for the destination, which is **0x12**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. For heartbeat messages, the request must be set to -1. |
| 5:1 | 0x11 | Key = status, Length = 1 |
| 6:1 | 0x0 or 0x1 | Status. 0x0 means offline and 0x1 means online. |
| 7:1 | 0x21 | Key = battery, Length = 1 |
| 8:1 | 0x0 to 0x64 | Battery level in percentage. 0 (0x) to 100 (0x64). |
| 9:1 | 0x31 | Key = deadbolt, Length = 1 |
| 10:1 | 0x0 or 0x1 | Deadbolt status. 0x0 means disengaged and 0x1 means engaged. |
| 11:1 | 0x41 | Key = latch, Length = 1 |
| 12:1 | 0x0 or 0x1 | Latch status. 0x0 means opened and 0x1 means closed. |
| 13:1 | 0x00 | End of Packet Marker |

### JSON Payload - IoT Gateway to Cloud IoT Server

The following MQTT message is sent from IoT Gateway to Cloud IoT Server for heartbeat message:

| MQTT Topic | {Device Id}/NOTIFICATION |
| --- | --- |
| JSON Payload | {  “Notification” : {  “status” : “online”,  “battery” : 85,  “deadbolt” : false,  “latch” : false  }  } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the notification corresponds to that lock.  The heartbeat contents are part of Notification object comprising of the following fields:   * status - can be either “online” or “offline” indicating the lock status * battery - provides the battery level in % ranging between 0 and 100. * deadbolt - boolean value of true (if engaged) or false (if disengaged) * latch - boolean value of true (if closed) or false (if opened) |



## Status Update Message

A lock controller sends out a *Status Update* message, when it detects any changes in the state of the following parameters:

* Lock
* Deadbolt
* Latch

The PDU structure and the JSON payload are the same as that sent out in case of Hearbeat message.

### PDU Structure - Lock controller node to IoT Gateway

See the PDU structure described in [Heartbeat - Periodic Telemetry Message](#_546e1y1vcbfy) section.

### JSON Payload - IoT Gateway to Cloud IoT Server

See the JSON payload described in [Heartbeat - Periodic Telemetry Message](#_o2ttv01k3imv) section.

## Alarm - Low-Battery

## Lock controllers emit low-battery alarm/alert when the battery voltage level drops below the minimum operating value

### PDU Structure - Lock controller node to IoT Gateway

The lock controller node sends the following alarm message to IoT Gateway on Notification-Alarm Endpoint (EP) for the destination, which is **0x13**:

| **Byte:Length** | **Field Value** | **Description** |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. For alarm messages, the request must be set to -1. |
| 5:1 | 0x11 | Key = low-battery, Length = 1 |
| 6:1 | 0x1 | The value is always 0x1 |
| 7:1 | 0x00 | End of Packet Marker |

### JSON Payload - IoT Gateway to Cloud IoT Server

The following MQTT message is sent from IoT Gateway to Cloud IoT Server for alarms:

| MQTT Topic | {Device Id}/ALARM |
| --- | --- |
| JSON Payload | {  “Notification” : {  “low-battery” : true  }  } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the alarm corresponds to that lock.  In the JSON payload’s Notification field, low-battery is always true indicating the battery voltage level has dropped below the minimum operating value. |



## Alarm - Deadbolt Alert

Lock controller emits deadbolt alarm/alert when the corresponding deadbolt is either engaged or disengaged.

### PDU Structure - Lock controller node to IoT Gateway

The lock controller node sends the following alarm message to IoT Gateway on Notification-Alarm Endpoint (EP) for the destination, which is 0x13:

| Byte:Length | Field Value | Description |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. For alarm messages, the request must be set to -1. |
| 5:1 | 0x21 | Key = deadbolt-alert, Length = 1 |
| 6:1 | 0x0 or 0x1 | 0x1 means the deadbolt is engaged and 0x0 means the deadbolt is disengaged. |
| 7:1 | 0x00 | End of Packet Marker |

### JSON Payload - IoT Gateway to Cloud IoT Server

The following MQTT message is sent from IoT Gateway to Cloud IoT Server for alarms:

| MQTT Topic | {Device Id}/ALARM |
| --- | --- |
| JSON Payload | {  “Notification” : {  “deadbolt-alert” : true  }  } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the alarm corresponds to that lock.  In the JSON payload’s Notification field, deadbolt-alert can be either true (deadbolt engaged) or false (deadbolt disengaged). |

## Alarm - Latch Alert

Lock controller emits latch alarm/alert when the corresponding latch is either closed or opened.

### PDU Structure - Lock controller node to IoT Gateway

The lock controller node sends the following alarm message to IoT Gateway on Notification-Alarm Endpoint (EP) for the destination, which is 0x13:

| Byte:Length | Field Value | Description |
| --- | --- | --- |
| 1:4 | <Req Id> | Request ID. For alarm messages, the request must be set to -1. |
| 5:1 | 0x31 | Key = latch-alert, Length = 1 |
| 6:1 | 0x0 or 0x1 | 0x1 means the latch is closed and 0x0 means the latch is opened. |
| 7:1 | 0x00 | End of Packet Marker |

### JSON Payload - IoT Gateway to Cloud IoT Server

The following MQTT message is sent from IoT Gateway to Cloud IoT Server for alarms:

| MQTT Topic | {Device Id}/ALARM |
| --- | --- |
| JSON Payload | {  “Notification” : {  “latch-alert” : true  }  } |
| Description | In MQTT topic, {Device Id} is set with the user-readable string as listed in CloudExt’s UI (for example: F1-R101-Lock), indicating that the alarm corresponds to that lock.  In the JSON payload’s Notification field, latch-alert can be either true (latch closed) or false (latch opened). |

# Appendix A - Wirepas Node Address and Device Id mapping CSV file

Here is an example file showing the mapping of Wirepas Node Address and Device Id:

| name,address,type,remarks  F1-R101-Lock,0xDFD81E,Lock,"Lock in 1st Floor, Room 101"  F8-R812-Lock,0xB56FCD,Lock,"Lock in 8th Floor, Room 812"  F1-R102-Lock,0x38E823,Lock,"Lock in 1st Floor, Room 102" |
| --- |

The mapping CSV file contains the following columns/fields:

* name - The name of the lock controller as identified in CloudExt IoT Platform UI. It corresponds to **Device Id**. It is expected to be in human-readable format, for example, F8-R812-Lock, representing the lock in Room 812 on the 8th floor.
* address - The address assigned to the lock controller in Wirepas RF mesh network.
* type - Type of the node. For lock controllers, it should be **Lock**.
* remarks - Space for writing free form text, which can be used, if required, to include additional information. This field is optional.