Macrotech Lock Controllers

Requirements and MQTT References

4 Nov 2022

# Introduction

This document captures the requirements for the interworking of IoT gateway and Macrotech’s CloudExt IoT Platform and the proposal from LiteNet Technology for MQTT references required for the interworking.

The document is updated to incorporate the changes required by CloudExt IoT Platform for handling MQTT messages.

# Gateway Requirements

Gateway serves as the middleware between Lock controller nodes in RF mesh (Wirepas mesh) network and Cloud hosted IoT Server (Macrotech’s CloudExt IoT Platform). The gateway performs the data/payload translation for exchanging the information between Lock controller nodes and the IoT server.

Here is the list of broad requirements for the gateway as a middleware:

1. Enable Cloud IoT Server to query Lock controller nodes to get the status of the lock, which includes the following details: The query can be either for an individual lock controller or for all lock controllers in the mesh network associated with the gateway.
   1. Status (online/offline),
   2. Battery Power,
   3. Deadbolt Status,
   4. Latch Status,
   5. Autolock Status
   6. Firmware Version
   7. Current Date and Time Setting
   8. RFID Details:
      1. Macrotech ID,
      2. Hotel ID,
      3. Admin Key,
      4. Staff Key,
      5. Guest Key,
      6. Lock Type,
      7. Floor,
      8. Room Number
2. Enable Cloud IoT Server to unlock door (opening the latch) in case of normal operation (when the deadbolt is not engaged)
3. Enable Cloud IoT Server to unlock door during emergency (when the deadbolt is engaged)
4. Enable Cloud IoT Server to enable or disable autolock functionality, so that the door can be opened without using NFC card or through the web interface if the autolock is disabled.
   1. Send telemetry data from the concerned lock controller soon after performing enable or disable autolock functionality.
5. Send periodic heartbeat messages (frequency is configurable) of all lock controllers to Cloud IoT Server (telemetry/heartbeat message)
6. Send an alert/alarm to Cloud IoT Server when the lock controller battery level drops below the minimum operating value (the exact value is determined by the firmware running on the lock controller nodes).
7. When a deadbolt is engaged or disengaged:
   1. Send an alert/alarm from that lock controller to Cloud IoT Server.
   2. Send telemetry notification from that lock controller to Cloud IoT Server.
8. When a latch is opened or closed:
   1. Send an alert/alarm from that lock controller to Cloud IoT Server.
   2. Send telemetry notification from that lock controller to Cloud IoT Server.
9. The gateway should automatically re-establish the session with MQTT broker (IoT server) when MQTT broker restarts.
10. Enable Cloud IoT Server to set Master Data on lock controller nodes. The Master Data consists of (1) Macrotech ID, which is a common ID for all Macrotech’s lock controllers deployed in various locations. (2) Hotel ID - A unique ID representing a particular hotel (3) Lock Type indicating if the lock controller is located in a guest room’s door or common door (4) Floor number (5) Room Number/Common Door Number (6) Autolock Setting
11. Enable Cloud IoT Server to set Admin Key on a specific or all lock controller nodes (broadcasting).
12. Enable Cloud IoT Server to set Staff Key on a specific lock controller node.
13. Enable Cloud IoT Server to set Guest Key on a specific lock controller node. To reset the guest data, the Guest Key should be set to 0.
14. Enable Cloud IoT Server to disable autolock functionality on all lock controllers in that mesh network, so that the door can be opened without using NFC card or through the web interface in case of fire or any emergency
15. Whenever an unlock is attempted by a RFID card (admin/staff/guest), send an alert/alarm containing that RFID card’s UID, the card type and the outcome of unlock operation (success/failure) from that lock controller to Cloud IoT Server.
16. Enable Cloud IoT Server to delete a specific card id or a particular type of card or or all card types from a particular lock controller or from all lock controllers.
17. Enable Cloud IoT server to set the date and time to a particular lock as well as all locks connected to the gateway at the same time.

# Gateway MQTT Reference for CloudExt IoT Server

This section contains the proposed MQTT reference for Gateway and CloudExt IoT Server for exchanging the data to fulfill the [gateway requirements](#_g60gpvff2fuz).

**Note 1:**

MQTT topics and payload formats listed in this section are based on the Register Command document received from Macrotech on 4th Feb 2022.

**Note 2:**

In the JSON payload exchanged between CloudExt IoT Server and Gateway, the field “Device Id” in MQTT topics corresponds to the unique identifier given for the lock controller. In the RF mesh (Wirepas mesh) network, MAC address is used as the node identifier, which poses a challenge in identifying the physical location of the corresponding lock. Assigning a unique human readable name like F1-R101-Lock (lock in Room 101 of Floor 1) or something similar is a better naming convention with regards to CloudExt’s user interface. So, the gateway shall maintain a mapping of RF mesh MAC address of human-readable unique name and use this mapping for translating the device id.

The mapping file (a CSV file) shall be provided to the gateway at the time of deployment/commissioning of the gateway.

## Registering lock controller nodes information with CloudExt IoT Server

The gateway supports the following MQTT topics for exchanging the data between lock controllers and CloudExt IoT server:

| **MQTT Topic** | **Type** | **Description** |
| --- | --- | --- |
| CMD | Subscription | Subscribes to this topic to receive commands from CloudExt IoT server for performing operations like unlocking, get status, … on the lock controllers. |
| NOTIFICATION | Publishing | The gateway publishes the telemetry data and one-time attribute data received from lock controllers. |
| ALARM | Publishing | The gateway publishes the alarms such as low-battery, deadbolt-alerts received from lock controllers. |
| CMD\_RESP | Publishing | The gateway publishes the command responses received from lock controllers. |

Accordingly, the gateway sends the following JSON registration message to CloudExt IoT server during the gateway startup:

| MQTT Topic | **system/{Gateway ID}/register/group** |
| --- | --- |
| JSON Payload | {  "group\_details": [  {  "device\_id" : "{Device ID} like F1-R101-Lock>",  "device\_name" : "<Name or can be device\_id>",  "command\_topics" : [  {  "topic" : "{Device ID}/CMD",  "qos" : "1"  }  ],  "event\_topics" : [  {  "topic" : "{Device ID}/ALARM",  "qos" : "1"  },  {  "topic" : "{Device ID}/CMD\_RESP",  "qos" : "1"  },  {  "topic" : "{Device ID}/NOTIFICATION",  "qos" : "1"  }  ]  },  {  "device\_id" : "<Another {Device ID} like F3-R304-Lock>",  "device\_name" : "<Name or can be device\_id>",  "command\_topics" : [  {  "topic" : "{Device ID}/CMD",  "qos" : "1"  }  ],  "event\_topics" : [  {  "topic" : "{Device ID}/ALARM",  "qos" : "1"  },  {  "topic" : "{Device ID}/CMD\_RESP",  "qos" : "1"  },  {  "topic" : "{Device ID}/NOTIFICATION",  "qos" : "1"  }  ]  },  ...  ]  } |
| Description | In the JSON payload, the **group\_details** array object contains the data (same data) for each lock controller in the given RF mesh network. If there are 100 lock controllers, then group\_details contains 100 elements in the array, each representing the individual lock controller. |

## Commands from Cloud IoT Server to Gateway

### Get Status - Individual Lock

The following MQTT message is sent from Cloud IoT Server for querying the status of a specific lock in the RF mesh network associated with the gateway:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | { “Command” : “**getstatus**”, “Seq” : <integer> } |
| Description | In the 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 controller.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests. |

### Get Status - From All Lock Controllers

The following MQTT message is sent from Cloud IoT Server to the gateway to get the status from all lock controllers:

| MQTT Topic | **{Gateway ID}/CMD** |
| --- | --- |
| JSON Payload | { “Command” : “**getstatus**”, “Seq” : <integer> } |
| Description | In the MQTT topic, **{Gateway ID}** is set with the user-readable string as listed in CloudExt’s UI (for example: Macrotech\_GW\_1) so that the command is broadcasted to all lock controllers in the mesh network associated with the gateway.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  **NOTE:**   * When this command is sent to the gateway, the gateway shall take care of broadcasting this command to all lock controllers in the given mesh network. * CloudExt IoT broker/server would receive as many responses as the number of active lock controllers in the given mesh network. All such responses carry the same Seq number sent in the command request. |

### Unlock Door - Normal Operation

The following MQTT message is sent from Cloud IoT Server for unlocking a door, when the deadbolt is not engaged from inside:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | { “Command” : “**unlock**”, “Seq” : <integer> } |
| Description | In the 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.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests. |

### Unlock Door - Emergency Operation

The following MQTT message is sent from Cloud IoT Server for unlocking a door, when the deadbolt is engaged from inside :

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | { “Command” : “**emunlock**”, “Seq” : <integer> } |
| Description | In the 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 as part of emergency operation.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests. |

### Enable Autolock

The following MQTT message is sent from Cloud IoT Server for enabling the autolock:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | { “Command” : “**enable\_autolock**”, “Seq” : <integer> } |
| Description | In the 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 autolock for the corresponding door should be enabled. When autolock is enabled, the locked door can only be opened either using NFC card or via web interface.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests. |

### Disable Autolock

The following MQTT message is sent from Cloud IoT Server for disabling the autolock:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | { “Command” : “**disable\_autolock**”, “Seq” : <integer> } |
| Description | In the 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 autolock for the corresponding door should be disabled. When autolock is disabled, the door remains unlocked.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests. |

### Disable Autolock On All Lock Controllers - Emergency Operation

The following MQTT message is sent from Cloud IoT Server to the gateway to disable the autolock on all lock controllers (**IMPORTANT NOTE:** this command shall be used only during an emergency such as fire detection):

| MQTT Topic | **{Gateway ID}/CMD** |
| --- | --- |
| JSON Payload | { “Command” : “**disable\_autolock**”, “Seq” : <integer> } |
| Description | In the MQTT topic, **{Gateway ID}** is set with the user-readable string as listed in CloudExt’s UI (for example: Macrotech\_GW\_1) so that the command is broadcasted to all lock controllers in the mesh network associated with the gateway.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  **NOTE:**   * When this command is sent to the gateway, the gateway shall take care of broadcasting this command to all lock controllers in the given mesh network. * CloudExt IoT broker/server would receive as many responses as the number of active lock controllers in the given mesh network. All such responses carry the same Seq number sent in the command request. |

### Set Master Data

The following MQTT message is sent from Cloud IoT Server for setting the master data on a lock controller:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | {  "Command" : "**set\_master\_data**",  "Seq" : <integer>,  "Params" : {  "macrotech\_id" : <unsigned int>,  "hotel\_id" : <unsigned short int>,  "lock\_type" : "guest\_room" or "common\_door",  "floor" : 0 to 255,  "room\_number" : <unsigned short>,  "autolock" : true/false  }  } |
| Description | In the 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 master data for the corresponding lock controller should be set.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following parameters:  (1) Macrotech ID, which is a common ID for all Macrotech’s lock controllers deployed in various locations.  (2) Hotel ID - A unique ID representing a particular hotel  (3) Lock Type indicating if the lock controller is located in a guest room’s door or common door  (4) Floor number (from 0 to 255)  (5) Room Number/Common Door Number  (6) Autolock Setting - true/false |

### Set Admin Key

The following MQTT message is sent from Cloud IoT Server for setting the admin key on a lock controller:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | {  "Command" : "**set\_admin\_key**",  "Seq" : <integer>,  "Params" : {  "key" : <unsigned int>  }  } |
| Description | In the 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 staff data for the corresponding lock controller should be set.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following parameters:  (1) key assigned to admin cards that can be used to unlock all rooms/doors. |

### Set Admin Key On All Lock Controllers

The following MQTT message is sent from Cloud IoT Server for setting the admin key on all lock controllers:

| MQTT Topic | **{Gateway ID}/CMD** |
| --- | --- |
| JSON Payload | {  "Command" : "**set\_admin\_key**",  "Seq" : <integer>,  "Params" : {  "key" : <unsigned int>  }  } |
| Description | In the 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 staff data for the corresponding lock controller should be set.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following parameters:  (1) key assigned to admin cards that can be used to unlock all rooms/doors.  **NOTE:**   * When this command is sent to the gateway, the gateway shall take care of broadcasting this command to all lock controllers in the given mesh network. * CloudExt IoT broker/server would receive as many responses as the number of active lock controllers in the given mesh network. All such responses carry the same Seq number sent in the command request. |

### Set Staff Key

The following MQTT message is sent from Cloud IoT Server for setting the staff data on a lock controller:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | {  "Command" : "**set\_staff\_key**",  "Seq" : <integer>,  "Params" : {  "key" : <unsigned int>  }  } |
| Description | In the 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 staff data for the corresponding lock controller should be set.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following parameters:  (1) key assigned to a staff card that can be used to unlock certain guest room doors. |

### Set Guest Key

The following MQTT message is sent from Cloud IoT Server for setting the guest data on a lock controller:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | {  "Command" : "**set\_guest\_key**",  "Seq" : <integer>,  "Params" : {  "key" : <unsigned int>  }  } |
| Description | In the 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 guest data for the corresponding lock controller should be set.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following parameters:  (1) key assigned to a guest card to open that particular guest room. To reset the guest key, the key should be set to 0. |

### Delete RFID Card Data

The following MQTT message is sent from Cloud IoT Server to delete the stored RFID card data (master or staff or guest or all of them or a specific card id) from a specific lock controller:

| MQTT Topic | **{Device ID}/CMD** |
| --- | --- |
| JSON Payload | {  "Command" : "**deletecard**",  "Seq" : <integer>,  "Params" : {  "card\_id" : <unsigned int>,  "card\_type" : "master" | "staff" | "guest" | "all"  }  } |
| Description | In the 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 RFID data should be deleted from that lock controller.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following mutually exclusive parameters:   1. Card id, the id of the card to be deleted. If the card id field is present, the card type field is ignored. 2. Card type, the type of card to be deleted. Card type can be either one of these: master, staff, guest, all. If all is set, all card data stored on the specified lock controller shall be deleted. Note that if the card id field is present, then this field is ignored. |

### Delete RFID Card Data from All Lock Controllers

The following MQTT message is sent from Cloud IoT Server to the gateway to delete the stored RFID card data (master or staff or guest or all of them or a specific card id) from all lock controllers in the mesh network:

| MQTT Topic | **{Gateway ID}/CMD** |
| --- | --- |
| JSON Payload | {  "Command" : "**deletecard**",  "Seq" : <integer>,  "Params" : {  "card\_id" : <unsigned int>,  "card\_type" : "master" | "staff" | "guest" | "all"  }  } |
| Description | In the MQTT topic, **{Gateway ID}** is set with the user-readable string as listed in CloudExt’s UI (for example: Macrotech\_GW\_1) so that the command is broadcasted to all lock controllers in the mesh network associated with the gateway.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following mutually exclusive parameters:   1. Card id, the id of the card to be deleted. If the card id field is present, the card type field is ignored. 2. Card type, the type of card to be deleted. Card type can be either one of these: master, staff, guest, all. If all is set, all card data stored on the specified lock controller shall be deleted. Note that if the card id field is present, then this field is ignored.   **NOTE:**   * When this command is sent to the gateway, the gateway shall take care of broadcasting this command to all lock controllers in the given mesh network. * CloudExt IoT broker/server would receive as many responses as the number of active lock controllers in the given mesh network. All such responses carry the same Seq number sent in the command request. |

### Set Date and Time

The following MQTT message is sent from Cloud IoT Server for setting the system date and time on a lock controller:

| MQTT Topic | {Device ID}/CMD |
| --- | --- |
| JSON Payload | {  "Command" : "set\_date\_time",  "Seq" : <integer>,  "Params" : {  "date\_time" : "YYYY-MM-DD, hh:mm:ss"  }  } |
| Description | In the 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 staff data for the corresponding lock controller should be set.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  The params field contains the following parameters:  (1) date\_time, a string in “YYYY-MM-DD, hh:mm:ss” form indicating the date and time to be set on the lock controller. The date\_time fields are as listed below:  YYYY - Year (for example: 2022)  MM - Month in the range 01 to 12 (01 - Jan … 12 - Dec)  DD - Date in the range 01 to 31  hh - Hour in 24-hour format in the range 00 to 23  mm - Minutes in the range 00 to 59  ss - Seconds in the range 00 to 59  **IMPORTANT NOTE:**  Since lock controller nodes do not have the timezone setting feature, the gateway sends the epoch seconds (UTC) as is without adjusting the seconds to the local timezone. Likewise, when it receives the epoch seconds from the lock controller nodes, it simply converts the epoch seconds to YYYY-MM-DD… form without adjusting for the local timezone. |

### Set Date and Time On All Lock Controllers

The following MQTT message is sent from Cloud IoT Server for setting the system date and time on all lock controllers:

| MQTT Topic | {Gateway ID}/CMD |
| --- | --- |
| JSON Payload | {  "Command" : "set\_date\_time",  "Seq" : <integer>,  "Params" : {  "date\_time" : "YYYY-MM-DD, hh:mm:ss"  }  } |
| Description | In the 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 staff data for the corresponding lock controller should be set.  The Seq field contains an integer that should be returned back as is in the response payload. The Seq id is used by the IoT server for tracking the requests.  (1) date\_time, a string in “YYYY-MM-DD, hh:mm:ss” form indicating the date and time to be set on the lock controller. The date\_time fields are as listed below:  YYYY - Year (for example: 2022)  MM - Month in the range 01 to 12 (01 - Jan … 12 - Dec)  DD - Date in the range 01 to 31  hh - Hour in 24-hour format in the range 00 to 24  mm - Minutes in the range 00 to 59  ss - Seconds in the range 00 to 59  **IMPORTANT NOTE:**  Since lock controller nodes do not have the timezone setting feature, the gateway sends the epoch seconds (UTC) as is without adjusting the seconds to the local timezone. Likewise, when it receives the epoch seconds from the lock controller nodes, it simply converts the epoch seconds to YYYY-MM-DD… form without adjusting for the local timezone.  NOTE:   * When this command is sent to the gateway, the gateway shall take care of broadcasting this command to all lock controllers in the given mesh network. * CloudExt IoT broker/server would receive as many responses as the number of active lock controllers in the given mesh network. All such responses carry the same Seq number sent in the command request. |

## Command Responses from Gateway to Cloud IoT Server

### Response for Get Status

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

**Note:**

When CloudExt IoT Server sends a query to get the status of all locks, there is no single response containing the details of all locks. Instead, individual responses are sent for each lock.

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  “Command” : “**getstatus**”,  “Seq” : <integer that was received in the request message>,  “Response” : {  “status” : “online”,  “battery” : 85,  “deadbolt” : false,  “latch” : false,  "firmware\_version" : <string>,  "date\_time" : "YYYY-MM-DD, hh:mm:ss",  "master\_data" : {  "macrotech\_id" : <unsigned int>,  "hotel\_id" : <unsigned short int>,  "lock\_type" : "guest\_room" or "common\_door",  "floor" : 0 to 255,  "room\_number" : <unsigned short>,  "autolock" : true/false  },  "admin\_key" : {  "key" : <unsigned int>  },  "staff\_key" : {  "key" : <unsigned int>  },  "guest\_key" : {  "key" : <unsigned int>  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  In the response object:   * The status field can be either “online” or “offline” (if the lock controller is down; currently, this is not supported) * The battery level can be between 0 to 100 (percentage) * For deadbolt, a boolean value of true/false is sent indicating whether the deadbolt is engaged (true) or disengaged (false). * For latch, a boolean value of true/false is sent indicating whether the latch is closed (true) or opened (false). * For firmware version, the version string in “<major>.<minor>.<maintenance>” form (for example: 1.2.40) is sent. * For date\_time, which represents the current date and time set on that lock controller, the data is in “YYYY-MM-MM, hh:mm:ss” form. * For autolock, a boolean value of true/false is sent indicating whether the autolock is enabled (true) or disabled (false). * RFID details: Master Data { Macrotech ID, Hotel ID, Lock type, Floor, Room number and Autolock setting (true/false)}, Admin Key {key}, Staff Key {key}, Guest Key {key}   **IMPORTANT NOTE:**  In the case of “date\_time” field, the gateway converts the epoch seconds received from the lock controller to “YYYY-MM-DD, hh:mm:ss” string using the local timezone. In other words, the date\_time string sent in this response message represents the time with respect to the timezone set on the gateway. So, it is assumed that the gateway is configured with the appropriate timezone. |

### Response for Get Status from All Lock Controllers

The MQTT response given for **Get Status** is sent from every active registered lock controller in the given mesh network.

### Response for Unlock Door - Normal Operation

The following MQTT message response is sent from Gateway to Cloud IoT Server for unlock door (normal operation) command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  “Command” : “**unlock**”,  “Seq” : <integer that was received in the request message>,  “Response” : {  “status” : “success”,  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Unlock Door - Emergency Operation

The following MQTT message response is sent from Gateway to Cloud IoT Server for unlock door (emergency operation) command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  “Command” : **“emunlock**”,  “Seq” : <integer that was received in the request message>,  “Response” : {  “status” : “success”,  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Enable Autolock

The following MQTT message response is sent from Gateway to Cloud IoT Server for enable autolock command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  “Command” : “**enable\_autolock**”,  “Seq” : <integer that was received in the request message>,  “Response” : {  “status” : “success”,  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Disable Autolock

The following MQTT message response is sent from Gateway to Cloud IoT Server for disable autolock command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  “Command” : “**disable\_autolock**”,  “Seq” : <integer that was received in the request message>,  “Response” : {  “status” : “success”,  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Disable Autolock On All Lock Controllers - Emergency Operation

The MQTT response given for [Disable Autolock](#_9afdi36o2x2g) is sent from every active registered lock controller in the given mesh network.

### Response for Set Master Data

The following MQTT message response is sent from Gateway to Cloud IoT Server for set master data command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  **"**Command" : **"set\_master\_data**",  "Seq" : <integer that was received in the request message>,  **"**Response" : {  "status" : "success",  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Set Admin Key

The following MQTT message response is sent from Gateway to Cloud IoT Server for set admin key command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  **"**Command" : **"set\_admin\_key**",  "Seq" : <integer that was received in the request message>,  **"**Response" : {  "status" : "success",  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Set Admin Key On All Lock Controllers

The MQTT response given for Set Admin Key is sent from every active registered lock controller in the given mesh network.

### Response for Set Staff Key

The following MQTT message response is sent from Gateway to Cloud IoT Server for set staff key command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  **"**Command" : **"set\_staff\_key**",  "Seq" : <integer that was received in the request message>,  **"**Response" : {  "status" : "success",  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Set Guest Key

The following MQTT message response is sent from Gateway to Cloud IoT Server for set guest key command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  **"**Command" : **"set\_guest\_key**",  "Seq" : <integer that was received in the request message>,  **"**Response" : {  "status" : "success",  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Delete RFID Card Data

The following MQTT message response is sent from Gateway to Cloud IoT Server for delete RFID data command:

| MQTT Topic | **{Device ID}/CMD\_RESP** |
| --- | --- |
| JSON Payload | {  “Command” : “**deletecard**”,  “Seq” : <integer that was received in the request message>,  “Response” : {  “status” : “success”,  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Delete RFID Data On All Lock Controllers

The MQTT response given for Delete RFID Data is sent from every active registered lock controller in the given mesh network.

### Response for Set Date and Time

The following MQTT message response is sent from Gateway to Cloud IoT Server for set date and time command:

| MQTT Topic | {Device ID}/CMD\_RESP |
| --- | --- |
| JSON Payload | {  "Command" : "set\_date\_time",  "Seq" : <integer that was received in the request message>,  "Response" : {  "status" : "success",  }  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The JSON payload contains three fields: command, seq, and response.  The command field contains the name of the command for which the response is associated.  The seq (sequence) field contains the integer that was received in the corresponding command request’s seq field.  The status field in the response can be either “success” or “failed” |

### Response for Set Date and Time On All Lock Controllers

The MQTT response given for Set Date and Time is sent from every active registered lock controller in the given mesh network.

## Telemetry, Notification from Gateway to Cloud IoT Server

### Heartbeat Message

The following MQTT message is sent from Gateway to Cloud IoT Server periodically as part of lock status (heartbeat) or when the state of autolock/latch/deadbolt changes:

| MQTT Topic | **{Device ID}/NOTIFICATION** |
| --- | --- |
| JSON Payload | {  “status” : “online”,  “battery” : 85,  “deadbolt” : false,  “latch” : false,  “autolock” : false  } |
| Description | In the 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 JSON payload is coming from that lock controller.  The status field can be either “online” or “offline” (if the lock controller is down; currently, this is not supported)  The battery level can be between 0 to 100 (percentage)  For deadbolt, a boolean value of true/false is sent indicating whether the deadbolt is engaged (true) or disengaged (false).  For latch, a boolean value of true/false is sent indicating whether the latch is closed (true) or opened (false).  For autolock, a boolean value of true/false is sent indicating whether the autolock is enabled (true) or disabled (false). |

## Alerts/Alarms from Gateway to Cloud IoT Server

### Low-Battery Alert

The following MQTT message is sent from Gateway to Cloud IoT Server when a lock controller’s battery level drops below the minimum operating value (the exact value is determined by the firmware running on the lock controller nodes)

| MQTT Topic | **{Device ID}/ALARM** |
| --- | --- |
| JSON Payload | { “low-battery” : true } |
| Description | Lock controllers emit a low-battery alarm/alert when the battery voltage level drops below the minimum operating value. |

### Deadbolt Alert

The following MQTT message is sent from Gateway to Cloud IoT Server the deadbolt is either engaged or disengaged.

| MQTT Topic | **{Device ID}/ALARM** |
| --- | --- |
| JSON Payload | { “deadbolt-alert” : true } |
| Description | deadbolt-alert can be either true (indicates deadbolt is engaged) or false (indicates deadbolt is disengaged). |

### Latch Alert

The following MQTT message is sent from Gateway to Cloud IoT Server the latch is either closed or opened.

| MQTT Topic | **{Device ID}/ALARM** |
| --- | --- |
| JSON Payload | { “latch-alert” : true } |
| Description | latch-alert can be either true (indicates latch is closed) or false (indicates latch is opened). |

### Unlock by RFID Card Alert

The following MQTT message is sent from Gateway to Cloud IoT Server when an attempt is made to unlock a door using an RFID card.

| MQTT Topic | **{Device ID}/ALARM** |
| --- | --- |
| JSON Payload | {  “unlock-by-rfid-alert” : {  “card\_id” : <unsigned int>,  “card\_type” : “master” | “staff” | “guest”,  “status” : “success” | “failure”  }  } |
| Description | The alert contains the following data:   * Card ID (master\_id or staff\_id or guest\_id) * Card Type (string: master, staff or guest) * Status - Whether or not the door was successfully unlocked. |

# 