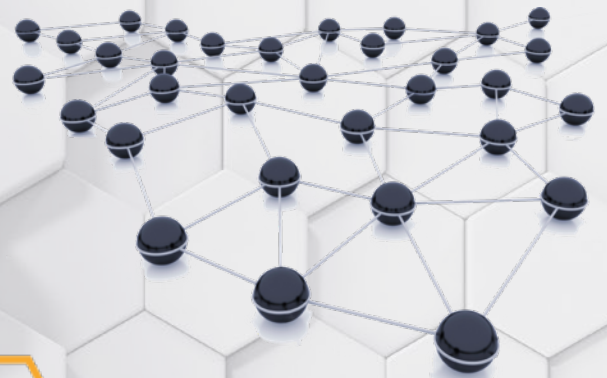


Libelium Cloud Bridge

Technical Guide



LIBELIUM CLOUD BRIDGE



Document version: v7.2 - 7/2019
© Libelium Comunicaciones Distribuidas S.L.

INDEX

1. General and safety information	4
1.1. Introduction.....	4
1.1.1. Overview	4
1.2. Data Flow	5
1.2.1. Direct HTTPS Connections	6
1.2.2. Meshlium IoT Gateway	7
1.2.3. LPWAN Callbacks.....	8
2. What Do You Need To Use The Libelium Cloud Bridge Service?	16
2.1. Create User Account	16
2.2. Register Devices	17
2.2.1. Adding Plug & Sense! Devices	17
2.2.2. Adding Meshlium Gateways	19
2.3. Activate Bridge License.....	21
2.4. Setup Cloud Connector	24
3. Cloud Service Connectors	25
3.1. New Connector Instance.....	26
3.1.1. Configuration.....	26
3.1.2. Associated Devices	28
3.1.3. Logs.....	29
3.2. Connector Status.....	31
3.3. Removing a Connector Instance	32
3.4. Configuration Parameters.....	33
3.4.1. Alibaba Cloud	33
3.4.2. AWS IoT	35
3.4.3. Cumulocity	38
3.4.4. IBM Bluemix.....	39
3.4.5. Microsoft Azure Event Hubs	41
3.4.6. Microsoft Azure IoT Hubs	44
3.4.7. MQTT	46
3.4.8. SAP Hana	47
3.4.9. Telefónica IoT Cloud	48
4. Manage Your Devices	49
4.1. API Keys	50
4.1.1. Creating The Keys.....	51
4.1.2. Show And Delete Existing API Keys.....	53
4.2. Encryption Keys	54
4.2.1. Change Key.....	54

5. Sending Data Frames	56
5.1. Data Streaming	56
5.1.1. Receiving Data	56
5.1.2. Buffering Data	75
5.1.3. Syncing With The Cloud Services	76
5.2. Encryption In Depth.....	77
6. Gateway Configuration	78
6.1. Remote Gateway Management.....	79
6.1.1. Enable Remote Management	80
6.1.2. Connect	81
6.2. Connection Timeout.....	83
7. Programming Sensor Nodes To Use The Libelium Cloud Bridge Service	84
7.1. Using The Programming Cloud Service	84
7.2. Using Wasp mote API (HTTPS Example)	86

1. General and safety information

1.1. Introduction

1.1.1. Overview

The Libelium Cloud Bridge service enables to any sensor node to push data directly and in a secured way to the main cloud services.

The Libelium Cloud Bridge is a service for connecting sensor nodes with many cloud services available for data analysis and graphical representation. Data goes through the Libelium Cloud Bridge service to the final cloud service. The Libelium Cloud Bridge service has a buffer of limited size that is cleared as far as data is sent to the final cloud service. The Libelium Cloud Bridge service does not store any data.

The cloud service connectors currently available in the Libelium Cloud Bridge service are:

- Alibaba Cloud
- AWS IoT
- Cumulocity
- IBM Bluemix
- Microsoft Azure Event Hubs
- Microsoft Azure IoT Hubs
- MQTT
- SAP Hana
- Telefónica IoT Cloud

The Libelium Cloud Bridge service has three functionalities:

- Configure your cloud connectors
- Manage your devices
- Configure your gateways

A license is required to access any of the Libelium Cloud Bridge service functionalities . These functionalities are oriented to send data from your sensor node or Meshlium gateway to any cloud service available.

1.2. Data Flow

Sensor nodes are the source: data that will be sent from these sensor devices to the cloud services.

Cloud services may implement dashboards and data analytics applications that need to be fed with data from the sensor nodes. The Libelium Cloud Bridge service will easily connect the sensor nodes with the cloud services (AWS IoT, Microsoft Azure, IBM Bluemix, Alibaba Cloud, ...) as shown in the next diagram.

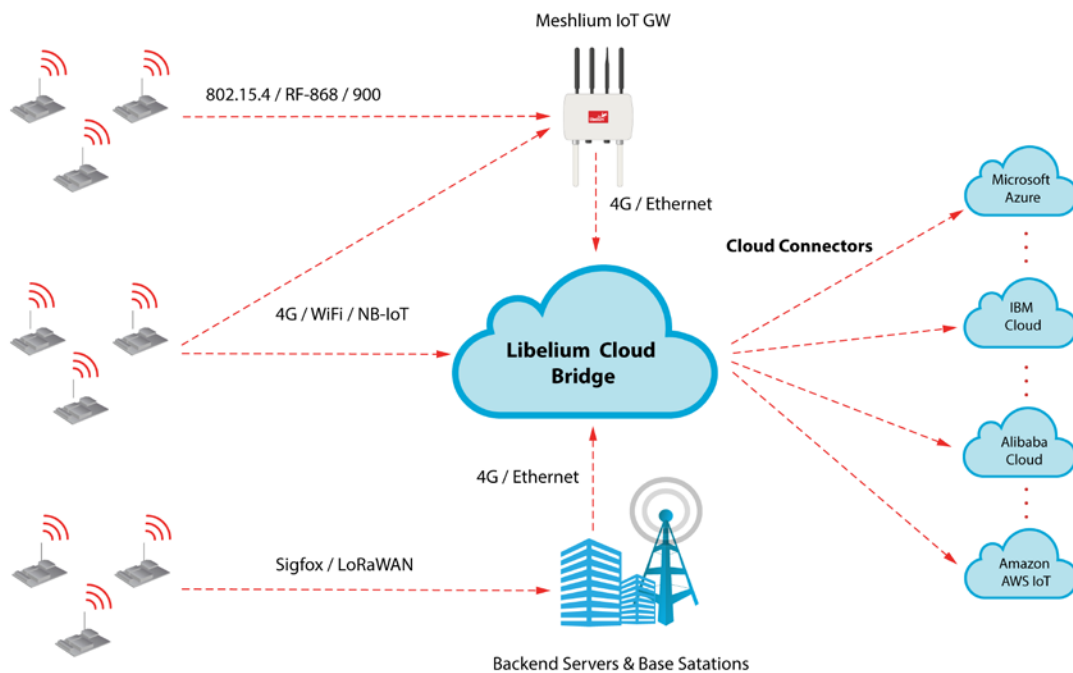


Figure: Libelium Cloud Bridge service data flow

1.2.1. Direct HTTPS Connections

Sensor nodes using NB-IoT / Cat-M /Cat-M, 4G and WiFi communication protocols can send data directly to the Libelium Cloud Bridge service.

Libelium Cloud Bridge service is listening to HTTPS requests to receive data from sensor nodes. Valid requests must comply with the following requirements:

- The sensor node has been registered in the Libelium Cloud (Services Cloud Manager) user account
- A valid authentication API Key has been associated with the sensor node
- Integrity of the Libelium Cloud Bridge service encryption layer is respected

Using the Programming Cloud Service for programming the sensor nodes will transparently include a valid API Key and the encryption layer functions. Programming the sensor nodes using the Waspnote IDE will require the user to include the valid API Key and use encryption functions in the source code. A detailed description of both processes is given in the section “Programming sensor nodes to use Libelium Cloud Bridge service” of this guide.

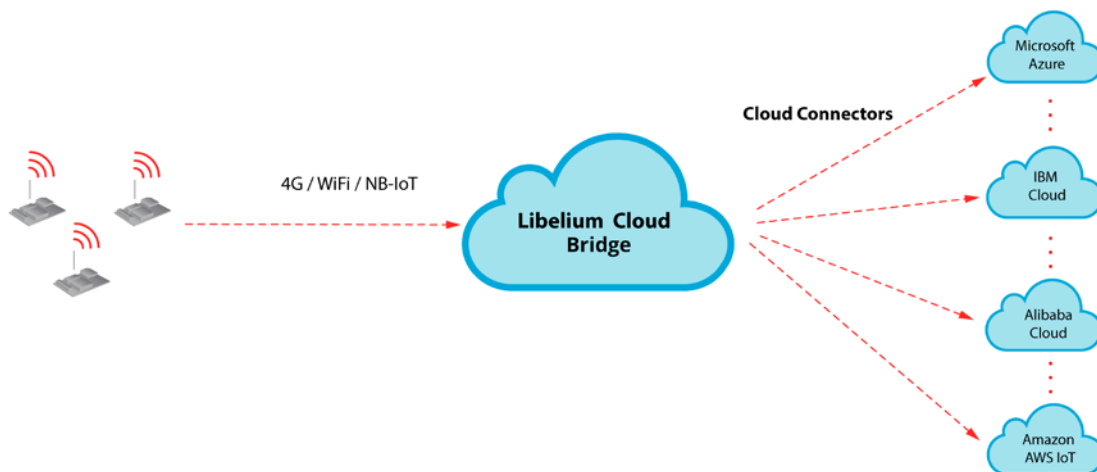


Figure: Libelium Cloud Bridge service, direct HTTPS data flow

1.2.2. Meshlium IoT Gateway

Sensor nodes using link level RF communication protocols like (802.15.4, RF-868, RF-900) must be properly configured for sending data to the Meshlium gateway.

Note: Meshlium (and Bridge) do not support the ZigBee protocol.

Sensor nodes using NB-IoT / Cat-M / Cat-M, 4G and WiFi communication protocols may be configured to send data to a Meshlium gateway too (as an alternative to the direct HTTPS connection to the Bridge, as explained in the section above).

The Meshlium gateway must be previously registered in the Libelium Cloud user account using the activation code provided by Libelium.

Meshlium must be also configured in the section “Cloud Connectors” of the Manager System (Libelium Cloud Bridge service connector) with the same user account used to register the Meshlium gateway in the Services Cloud Manager.

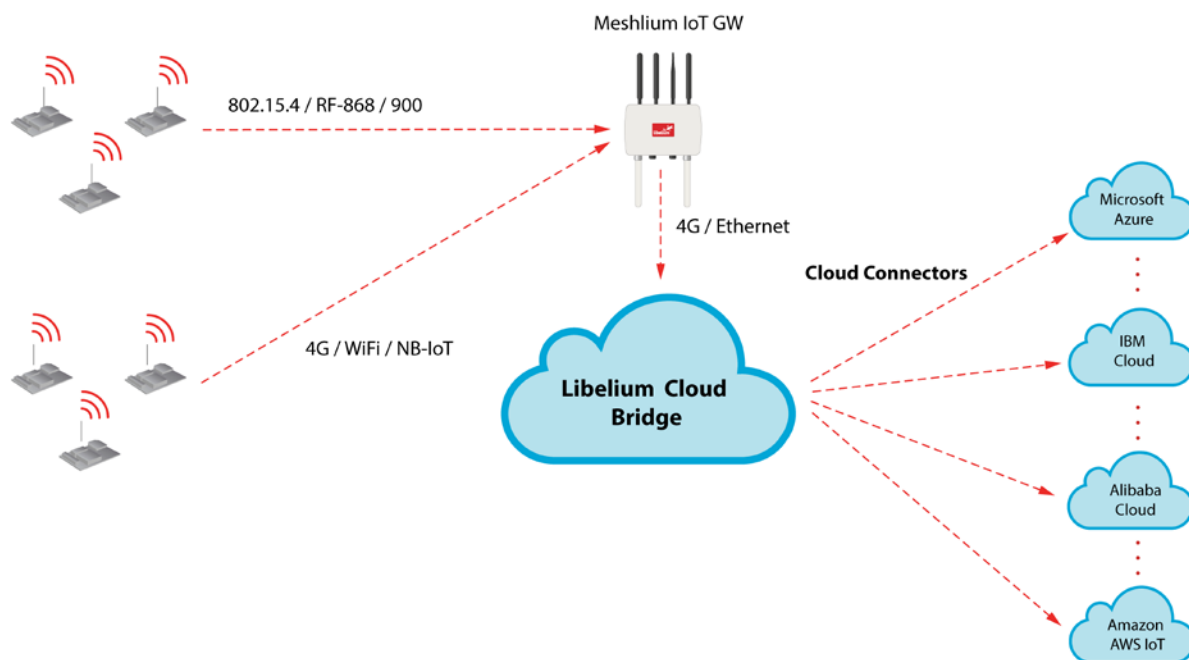


Figure: Libelium Cloud Bridge service, Meshlium data flow

1.2.3. LPWAN Callbacks

Sensor nodes using Sigfox and LoRaWAN communication protocols need to send data directly to the base stations. Callback functions of the LPWAN backend systems are used to redirect data from the base stations to the Libelium Cloud Bridge service. The Libelium Cloud Bridge service provides data security and user privacy implementing a security layer for the incoming data from the LPWAN backend systems to the cloud services.

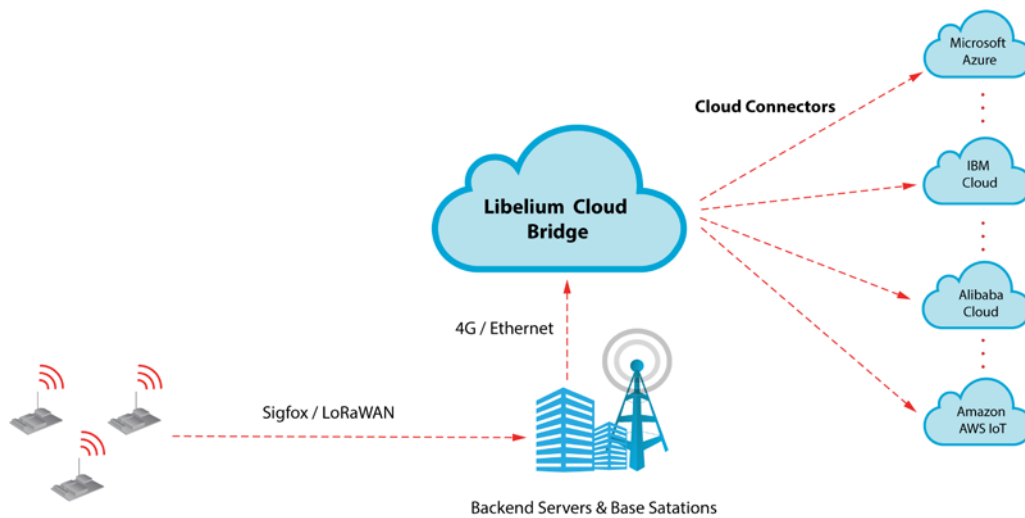


Figure: Libelium Cloud Bridge service, LPWAN data flow

The Libelium Cloud Bridge service is listening to HTTPS requests to receive data from LPWAN base stations. Valid requests must comply with the following requirements:

- The sensor node has been registered in the Libelium Cloud user account
- A valid authentication API Key has been associated with the sensor node
- Integrity of the “tiny frame” is respected

All LPWAN backend servers perform these steps in a different way:

- Create a device (optional)
- Create a frame type
- Create a frame structure (like JSON)
- Create a callback URL to redirect data (this URL is provided by Libelium in this case)

The Libelium Cloud Bridge service is compatible out-of-the-box with Sigfox, Lorient and Actility; this is accomplished by the user, configuring the callback function of each platform so that it points to the Libelium Cloud Bridge service.

The Libelium Cloud Bridge service provides the specific configuration parameters required for each LPWAN backend server, publishing to all the chosen final cloud services the data received from any LPWAN backend server. So Bridge behaves like a data hub and provides high scalability and makes data available as received.

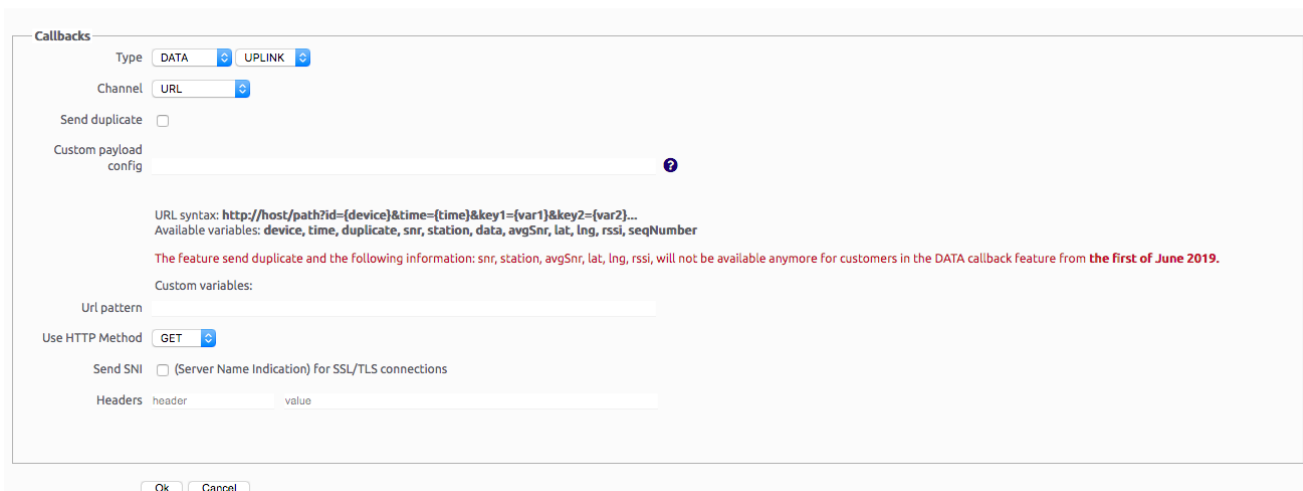
The user’s data integrity is kept with API Key tokens used for authentication. These user credentials may be revoked anytime and regenerated again; as a result of renovating credentials, all new frames will be rejected by the Bridge until the user updates the authentication API Key in each sensor node.

1.2.3.1. Sigfox

Sigfox backend server may be configured to redirect the data received from the base stations to the Libelium Cloud Bridge service. A new callback server in the Sigfox panel must be created using the URL: <https://hw.libelium.com/hw/sigfox>

Click on “Callbacks” in the left menu and then create a new callback: click the “New” button in the window loaded.

Device type - Callback new



Callbacks

Type: DATA UPLINK

Channel: URL

Send duplicate: ☐

Custom payload config:

URL syntax: `http://host/path?id={device}&time={time}&key1={var1}&key2={var2}...`
 Available variables: `device, time, duplicate, snr, station, data, avgSnr, lat, lng, rssi, seqNumber`

The feature send duplicate and the following information: `snr, station, avgSnr, lat, lng, rssi`, will not be available anymore for customers in the DATA callback feature from the first of June 2019.

Custom variables:

Url pattern:

Use HTTP Method: GET

Send SNI: ☐ (Server Name Indication) for SSL/TLS connections

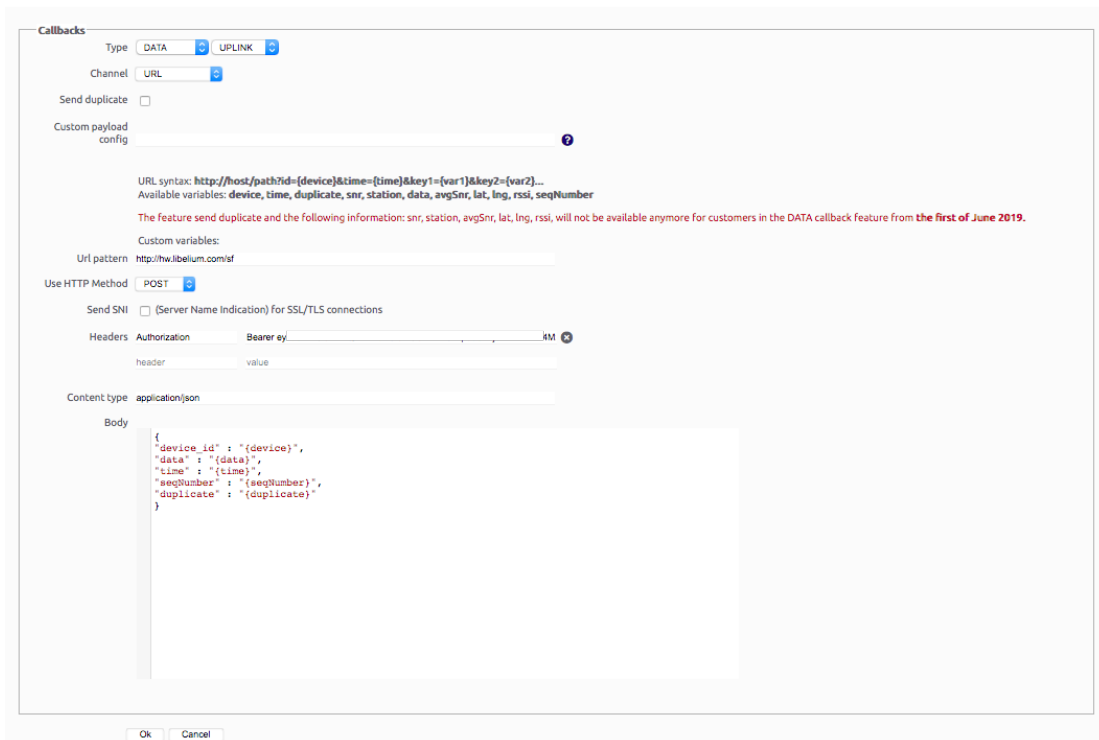
Headers:

header	value
--------	-------

Figure: Callback service creation

Fill in the displayed form following the indications shown in the next image.

Device type - Callback edition



Callbacks

Type: DATA UPLINK

Channel: URL

Send duplicate: ☐

Custom payload config:

URL syntax: `http://host/path?id={device}&time={time}&key1={var1}&key2={var2}...`
 Available variables: `device, time, duplicate, snr, station, data, avgSnr, lat, lng, rssi, seqNumber`

The feature send duplicate and the following information: `snr, station, avgSnr, lat, lng, rssi`, will not be available anymore for customers in the DATA callback feature from the first of June 2019.

Custom variables:

Url pattern: `http://hw.libelium.com/sf`

Use HTTP Method: POST

Send SNI: ☐ (Server Name Indication) for SSL/TLS connections

Headers:

header	value
Authorization	Bearer ey...

Content type: application/json

Body:

```
{
  "device_id": "{device}",
  "data": "{data}",
  "time": "{time}",
  "seqNumber": "{seqNumber}",
  "duplicate": "{duplicate}"
}
```

Figure: Callback service creation

The text in the field “URL pattern” must be the Libelium Cloud Bridge URL for Sigfox with extra parameters displayed in the form to send as much information as possible to the service.

Example of the text to write in the URL “pattern” field with the extra parameters:

```
{
https://hw.libelium.com/hw/sigfox/?id={device}&time={time}&duplicate={duplicate}&snr=
{snr}&station={station}&data={data}&avgSnr={avgSnr}&lat={lat}&lng={lng}&rssi={rssi}&seqNumb
er={seqNumber}&ack=true
}
```

The list of all available callbacks will be shown saving the form.

Enable this entry with a downlink and the Libelium Cloud Bridge service will receive the information.

1. Activate the “Downlink” column. Click on the bullet in this column, leaving the bullet coloured.

- Downlink active
- Downlink inactive

2. Enable the callback. Click on the check icon in the “Enable” column of the callback created and leave the check with a green colour.

- Callback enabled
- Callback disabled



Figure: Callback list

In the section “Sending Data Frames” → “Receiving Data” of this guide we describe in detail the steps of the configuration of the callback in the Sigfox backend server.

1.2.3.2. Lorient

Configuring the Libelium Cloud Bridge service as a callback server in the Lorient panel may be done using the URL: <https://hw.libelium.com/hw/loriot> in the “Data output” option of the “Network application” menu available for the devices.

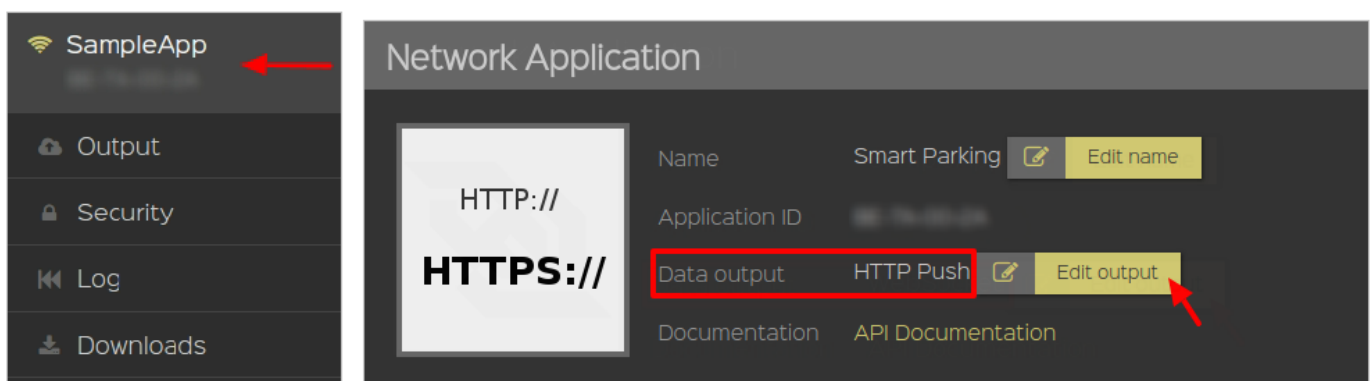


Figure: Output application selection

Click on “Edit output” to display all the information available about the selected output option. Click on the “Change” button in the detailed window to change the data output and select “HTTP Push” among the multiple choices from the list.

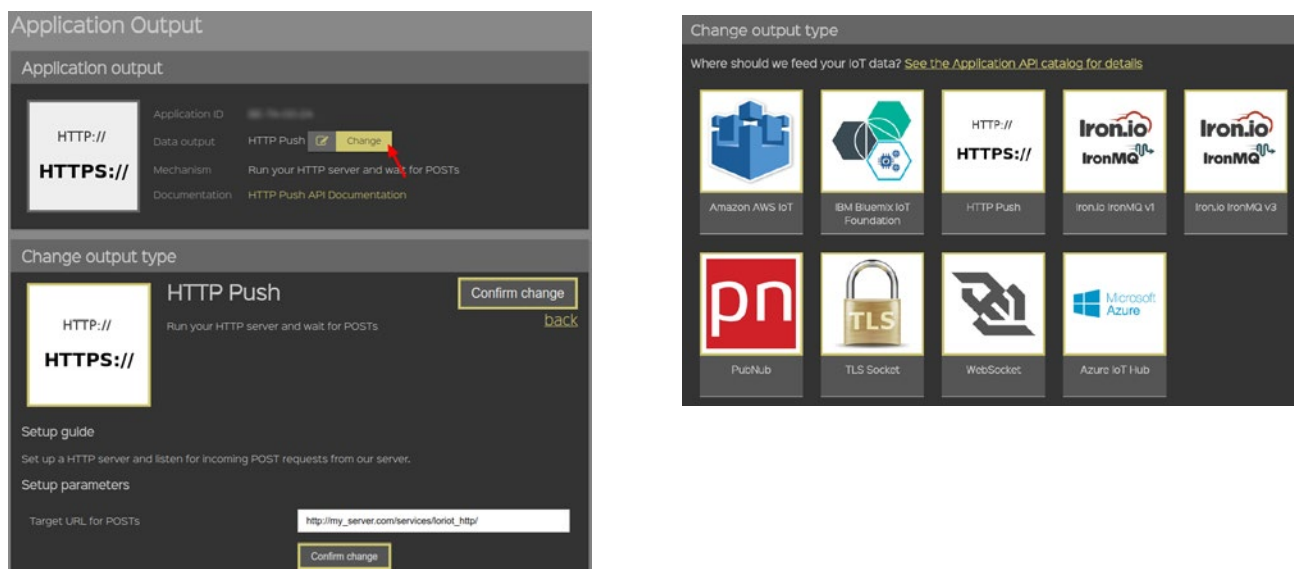


Figure: Selecting the output application

Once selected the new output, a new section “Change output type” will be shown below, with a text field labelled as “Target URL for POSTs”. Write in the text field the URL of the Libelium Cloud Bridge service callback for this service: <https://hw.libelium.com/hw/loriot>.

In the section “Sending Data Frames” → “Receiving Data” of this guide we describe in detail the steps of the configuration of the callback in the Lorient backend server.

1.2.3.3. Activity

Configuring the Libelium Cloud Bridge service as a callback server in the Activity panel is done using the Application Server feature. Create a new HTTP Application Server adding a new Destination in the “Route” field with this URL: <https://hw.libelium.com/hw/activity>.

Application server

Application server

Name: * SmartParking server

ID: TWA

Content Type: * JSON

Type: HTTP Application Server

Uplink/downlink security

Status: Inactive

Max timestamp deviation: -

Activate

Add a route

Add an additional route to this application server.

+ Add

Status

Last modification:

Updated by:

Figure: Application server form

A “Route” section will be created above with a form, just click the “Add” button to proceed.

Route

Source ports: *

Routing strategy: * Sequential

Destinations

Destination

Edit + Add Delete Up Down

Figure: Route form

Write the URL <https://hw.libelium.com/hw/activity> of the Libelium Cloud Bridge service callback for the Activity service in the new pop-up. Click the “Add” button to save the information.

Add destination

Destination: http://myserver.com/services/activity/

+ Add Close

Figure: Destination form

Navigate back to the application server form and click on “Save” to create the application server. Then, configure a new AS routing profile in the “Device Manager” clicking on “AS routing profiles” on the left sidebar menu.



Figure: AS Routing profiles menu

An “Add button” is shown in the first section. Below, in the second section, a list with all existing AS routing profiles will be displayed. Click on the “Add” button of “New AS routing profile” to create a new AS routing profile.

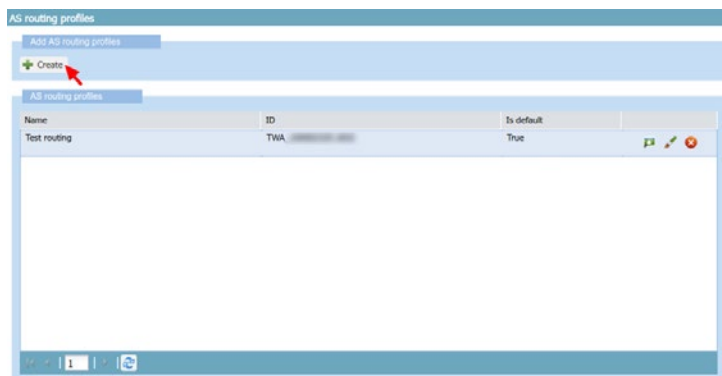


Figure: AS Routing profiles

A name must be typed in the new window displayed. Click the “Create” button to continue the process.

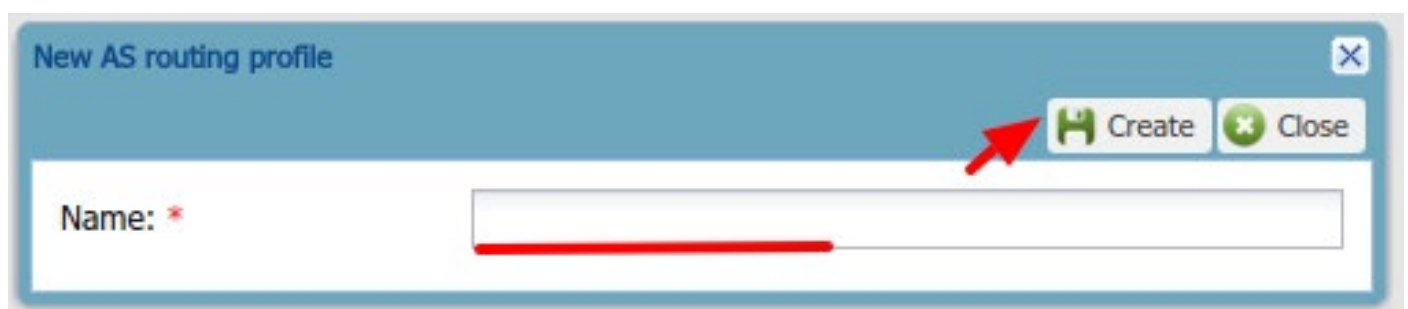
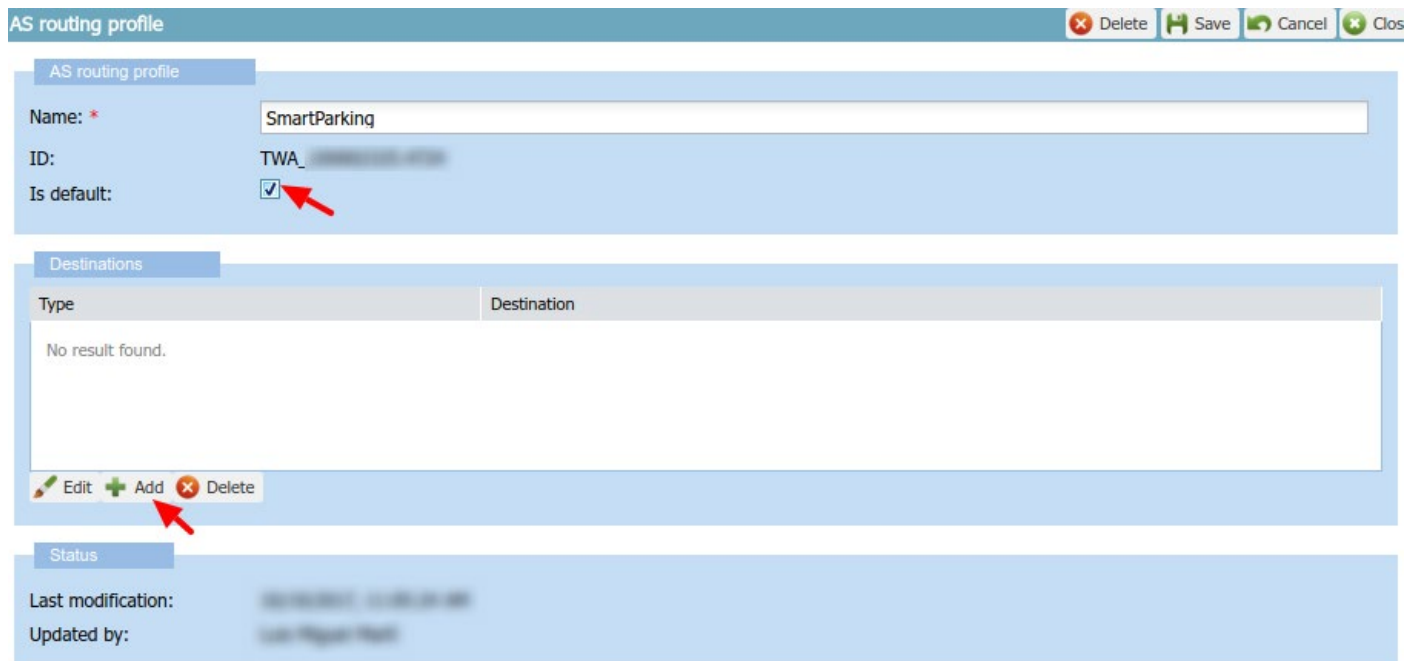


Figure: AS Routing profile name

In the new window, mark the check “Is default” and click the “Add” button in the “Add a route” section.




AS routing profile

AS routing profile


Name: * SmartParking

ID: TWA_...

Is default: ☒ 

Destinations

Type	Destination
No result found.	

Edit Add  Delete

Status

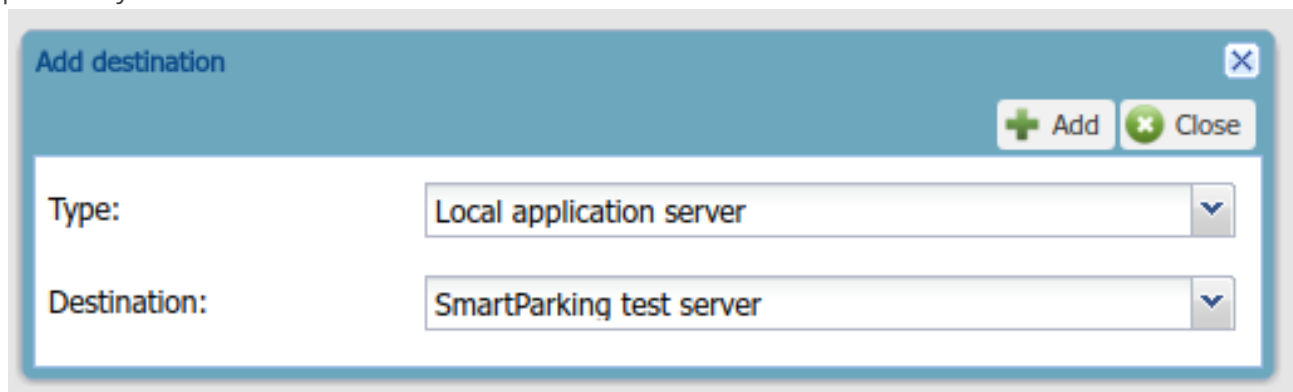
Last modification: ...

Updated by: ...

Figure: AS Routing profile route

A new pop-up will appear with two parameters to be completed:

- “Type” field: It is the kind of the application for our destination. “Local application server” must be selected.
- “Destination” field: All the application servers created are loaded in the drop-down, so the application server previously created must be selected.



Add destination

+ Add Close

Type: Local application server

Destination: SmartParking test server

Figure: Add a route

1.2.3.4. MultiTech LoRaWAN Base Station

Data frames are the new way to communicate between data hubs and sensors. The Libelium Cloud Bridge service provides a new method to receive data directly from the MultiTech base stations and synchronize all the values received with the cloud services configured. Many base stations may be configured to send data to the same user account of the Libelium Cloud Bridge service.

MultiTech base stations may be configured to send data frames to the Libelium Cloud Bridge service installing a lightweight application.

A link is created between the MultiTech Base Station and the Libelium Cloud Bridge service with periodic calls to the endpoint https://hw.libelium.com/hw/lw_multitech. It is recommended scheduling the application properly to avoid sending a large number of frames.

2. What Do You Need To Use The Libelium Cloud Bridge Service?

2.1. Create User Account

The Libelium Cloud Bridge service is available in the Libelium Services Cloud Manager site: <https://cloud.libelium.com/>

In order to access the Libelium Services Cloud Manager site, a valid user account is required. To register and set up a password go to <https://cloud.libelium.com/register> and click on "Create account".



Figure: Registering new user

Complete the information in the form (all the fields are mandatory):

- Name: Your name
- E-Mail address: A valid e-mail address, it will be used for verification purposes
- Password and Confirm password: Set your password

Accepting Terms & Conditions is also mandatory to create a new account in the Services Cloud Manager.

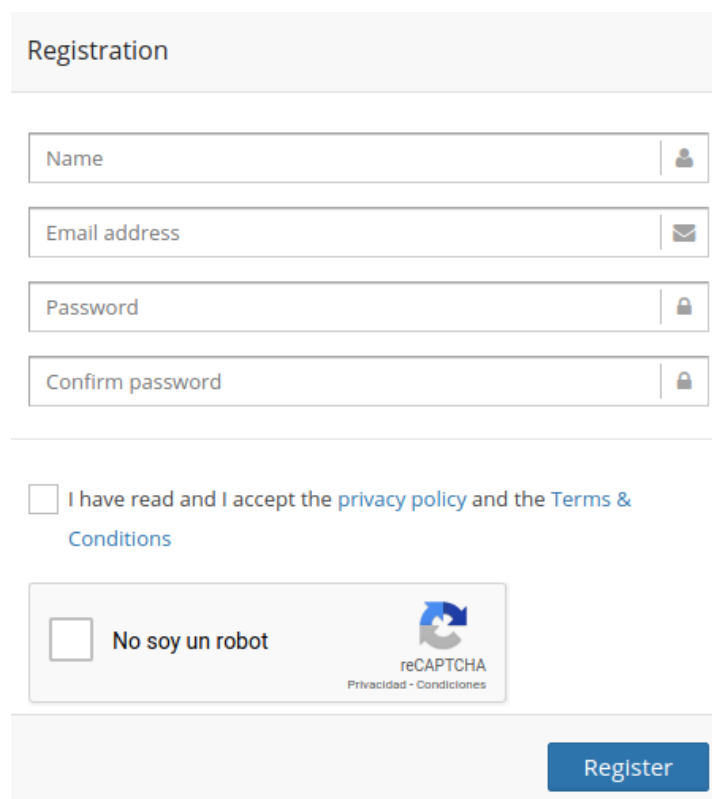
A registration form titled "Registration" in a light gray header. It contains four input fields: "Name" with a person icon, "Email address" with an envelope icon, "Password" with a lock icon, and "Confirm password" with a lock icon. Below the fields is a checkbox labeled "I have read and I accept the [privacy policy](#) and the [Terms & Conditions](#)". At the bottom is a reCAPTCHA widget with the text "No soy un robot" and a "reCAPTCHA Privacidad - Condiciones" link. A blue "Register" button is located at the bottom right of the form.

Figure: Registering new user form

Only one valid user in the Libelium Services Cloud Manager site is needed for all the services, existing users of other Services Cloud Manager services (MySignals, Programming Cloud Service, ...) may use their current account. All the previously registered devices in the Libelium Cloud will be available as well to the Libelium Cloud Bridge service.

2.2. Register Devices

Sensor nodes, MySignals devices and Meshlium gateways should be registered with the user account to properly use the Libelium Services Cloud Manager services. MySignals devices registration must be done using the app available in the Google Play store for Android and the App Store for iOS. Plug & Sense! sensor nodes and Meshlium gateways are registered using activations codes in the Libelium Service Cloud Manager.

2.2.1. Adding Plug & Sense! Devices

Click the “Add Device” button to add new devices to the Services Cloud Manager. Registering new devices is a 2-step procedure:

- Enter a valid Activation Code
- Confirm the action

Entering a Single Activation Code will register one device, while entering a Group Activation Code will register all devices belonging to the same sales order.

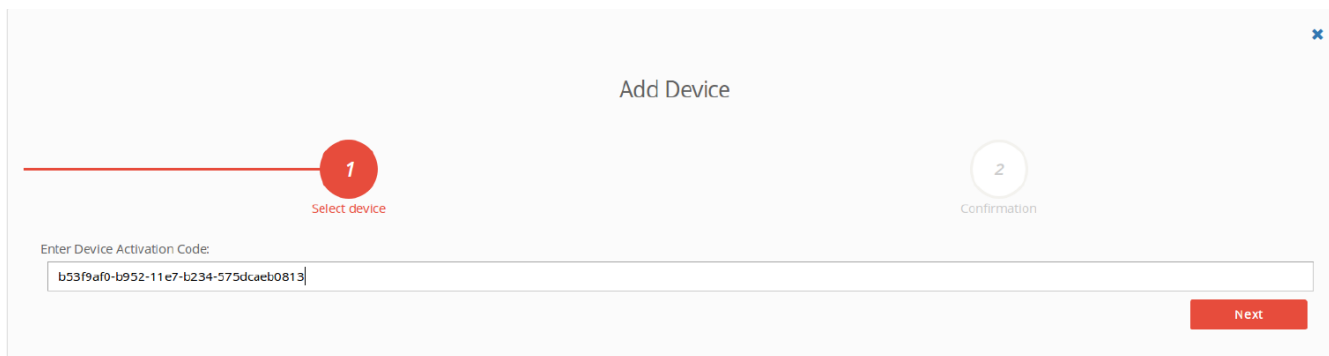


Figure: Registering a new device

When the process is successfully finished, a message will show that the device has been correctly added to the “My Devices” panel, confirming the ownership of the device.

Success!



The device has been added to your account

OK

Figure: Successful registration

If the process cannot be correctly finished, a message will notify the error:

Not valid code error message: “Invalid activation code: please check that the activation code typed is valid”.

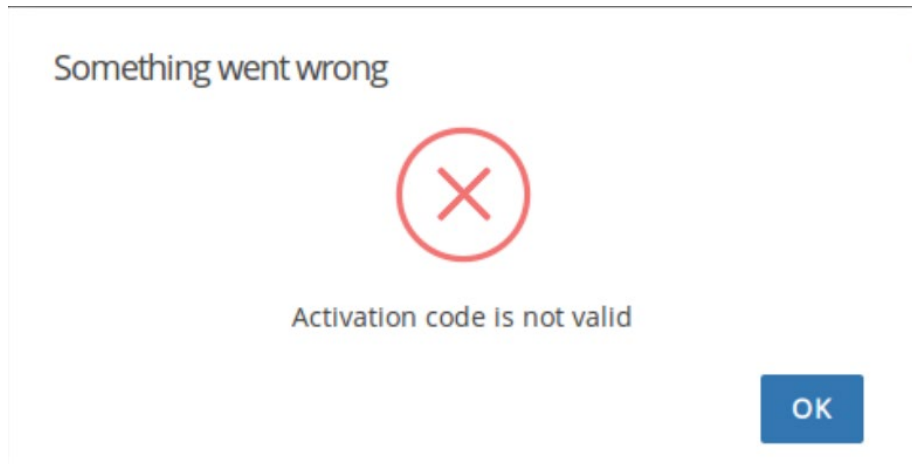


Figure: Registering a new device

Not available code error message: "Activation code previously used". The device will appear in the "My Devices" panel with a status of "Transfer pending" in the action column. The current owner of the device will be notified about another user claiming for the ownership of that device. The pending transfer device will also appear in the current owner "Claimed by others" panel, so the current owner of the device may accept or reject the ownership request of the device.

<input type="checkbox"/>	5600868251668210	Plug & Sense!	Device #5600868251668210	Smart City :: Dandreview	Node #3 installed at GTM	 
--------------------------	------------------	---------------	--------------------------	--------------------------	--------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure: Registering a new device

2.2.2. Adding Meshlium Gateways

Click the “Add Device” button to add new gateways to the Libelium Cloud Bridge service. Registering new gateways is a 2-step procedure:

- Enter a valid activation code
- Confirm the action

Entering a “Single Activation Code” will register one gateway, while entering a “Group Activation Code” will register all gateways belonging to the same sales order.

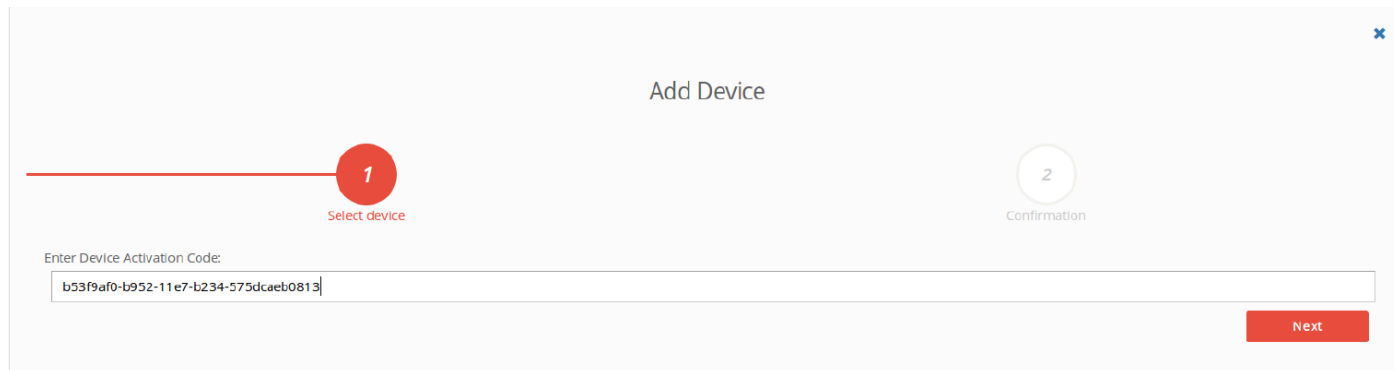


Figure: Registering a new device

When the process is finished successfully, a message will show that the device has been correctly added to the “My Devices” panel, confirming the ownership of the device.

Success!



The device has been added to your account

OK

Figure: Registering a new device

If the process cannot be correctly finished, a message will notify the error:

Not valid code error message: “Invalid activation code: please check that the activation code typed is valid”.

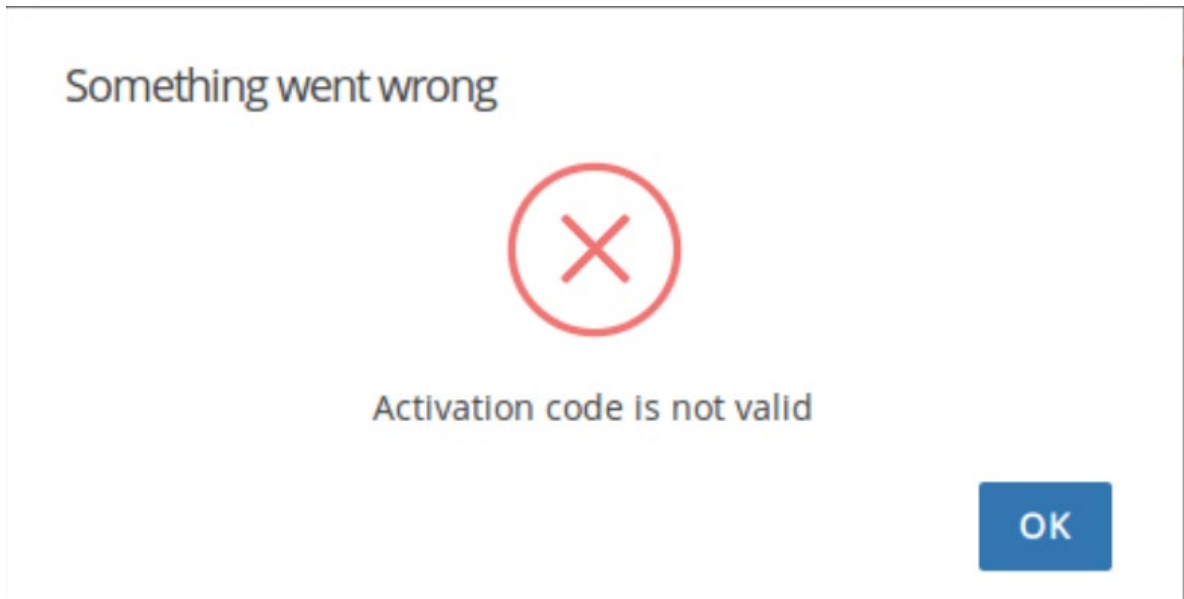


Figure: Registering a new device

Not available code error message: "Activation code previously used". The device will appear in the "My Devices" panel with a status of "Transfer pending" in the action column. The current owner of the device will be notified about another user claiming for the ownership of that device. The pending transfer device will also appear in the current owner "Claimed by others" panel, so the current owner of the device may accept or reject the ownership request of the device.

<input type="checkbox"/>	5600868251668210	Plug & Sense!	Device #5600868251668210	Smart City :: Dandreview	Node #3 installed at GTM	 
--------------------------	------------------	---------------	--------------------------	--------------------------	--------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure: Registering a new device

2.3. Activate Bridge License

A valid user account and some registered devices are two things needed to properly use any Libelium Cloud service. The Libelium Cloud Bridge service can be activated using a valid license activation code.

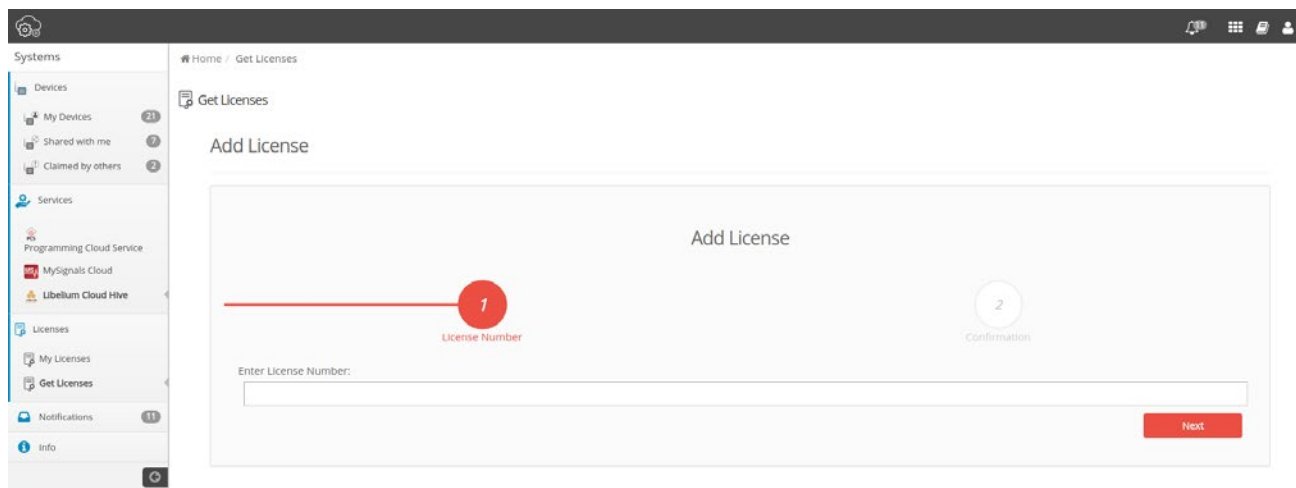


Figure: Screen licenses menu

Registering a new license is a two-step procedure:

- Enter a valid license activation code
- Confirm the action

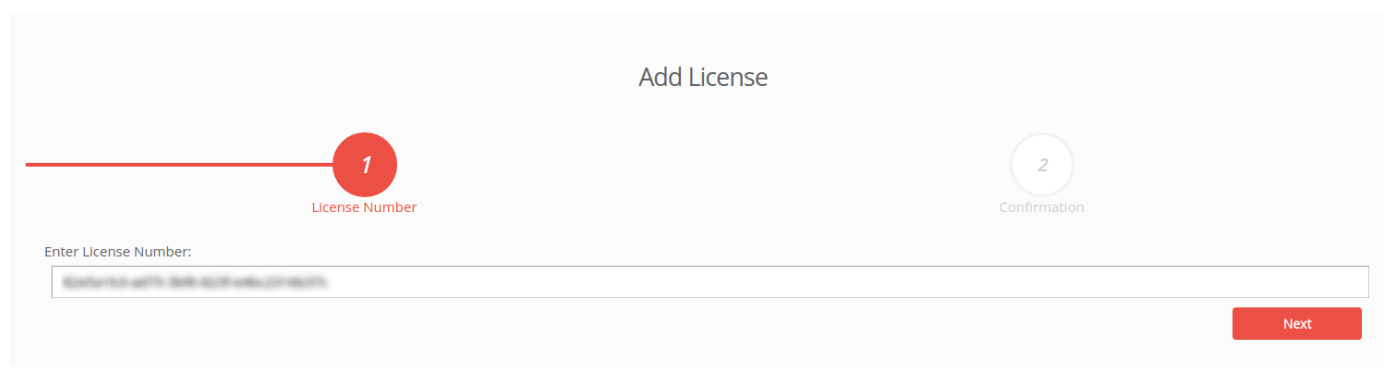


Figure: Registering a new license

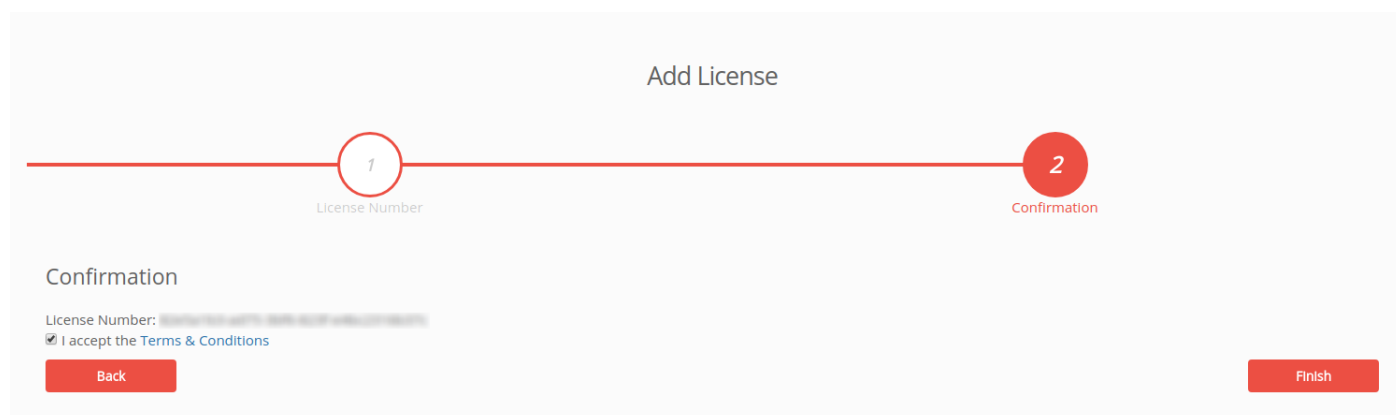


Figure: Registering a new license

The Services Cloud Manager will validate the activation code, displaying a visual confirmation. When the process is finished successfully, a message will show that the license has been correctly added to the “My Licenses” panel.

Success!



License added

OK

Figure: Success message

If the process cannot be correctly finished, a message will notify the error:

Not valid code error message: “Invalid activation code: please check that the activation code typed is valid”.

Something went wrong



License not valid

OK

Figure: Error message

The “Get Licenses” panel shows information about the license type for each service, click on any of them to purchase new licenses.

Basic License	Pro License POPULAR	Elite License
150 € /year	300 € /year	1499 € /year
<ul style="list-style-type: none"> ✓ 5 Nodes to program ✓ All P&SI models supported ✓ All Sensors supported ✓ Actuators Control (Relay ON/OFF in P&SI Security) ✓ Sigfox / LoRaWAN Settings Setup ✓ 802.15.4 / RF 868/900MHz Settings Setup ✓ 4G / WIFI Settings Setup ✓ GPS Settings Setup ✓ HTTP (4G / WIFI) ✗ HTTPS (4G / WIFI) ✗ RF Link Encryption (802.15.4 / RF 868 / 900) ✗ AES 256 Payload Encryption ✗ Industrial Protocols Support (ModBus/CanBus over RS232/RS485) ✓ Low Battery Warning ✓ Templates Manager ✗ Batch Programming (Generate up to 100 binaries in one click) 	<ul style="list-style-type: none"> ✓ 20 Nodes to program ✓ All P&SI models supported ✓ All Sensors supported ✓ Actuators Control (Relay ON/OFF in P&SI Security) ✓ Sigfox / LoRaWAN Settings Setup ✓ 802.15.4 / RF 868/900MHz Settings Setup ✓ 4G / WIFI Settings Setup ✓ GPS Settings Setup ✓ HTTP (4G / WIFI) ✓ HTTPS (4G / WIFI) ✓ RF Link Encryption (802.15.4 / RF 868 / 900) ✓ AES 256 Payload Encryption ✓ Industrial Protocols Support (ModBus/CanBus over RS232/RS485) ✓ Low Battery Warning ✓ Templates Manager ✗ Batch Programming (Generate up to 100 binaries in one click) 	<ul style="list-style-type: none"> ✓ 100 Nodes to program ✓ All P&SI models supported ✓ All Sensors supported ✓ Actuators Control (Relay ON/OFF in P&SI Security) ✓ Sigfox / LoRaWAN Settings Setup ✓ 802.15.4 / RF 868/900MHz Settings Setup ✓ 4G / WIFI Settings Setup ✓ GPS Settings Setup ✓ HTTP (4G / WIFI) ✓ HTTPS (4G / WIFI) ✓ RF Link Encryption (802.15.4 / RF 868 / 900) ✓ AES 256 Payload Encryption ✓ Industrial Protocols Support (ModBus/CanBus over RS232/RS485) ✓ Low Battery Warning ✓ Templates Manager ✓ Batch Programming (Generate up to 100 binaries in one click)
Buy your license!	Buy your license!	Buy your license!

Figure: Purchase license

2.4. Setup Cloud Connector

The Libelium Cloud Bridge service is focused in sending data received from the sensor nodes to any of the available cloud services (Alibaba Cloud, AWS IoT, Cumulocity, IBM Bluemix, Microsoft Azure Event Hubs, Microsoft Azure IoT Hubs, MQTT, SAP Hana and Telefónica IoT Cloud). The Libelium Cloud Bridge service requires at least one valid account in the destination cloud service, the connection credentials provided by the destination cloud services must be configured to properly send data to them. In case of having more than one valid account in the destination cloud service, several instances of one connector may be configured.

◀ CONFIGURED CONNECTORS > ADD CONNECTOR

≡ More actions

Add connector



✕ Cancel

Figure: Configuration of a new cloud connector

The specific parameters for each cloud service configuration are described in the section “Cloud Service Connectors → Configuration Parameters” of this guide.

3. Cloud Service Connectors

In this view, all the cloud service connector instances that have been already configured by the user are listed. The first time accessing to the Libelium Cloud Bridge service the list is empty, the message “You don’t have any connector configured, click on the button below to add a new one” is displayed.

At least one cloud service connector instance must be configured to use the Libelium Cloud Bridge service. So programming sensor nodes to send data to the Libelium Cloud Bridge service before a valid cloud service instance is configured will fill the buffer and the sensor nodes will receive errors when trying to send data.

Adding the first cloud connector instance to the Libelium Cloud Bridge service:

More actions

Configured connectors

You don't have any connector configured, click on the button below to add a new one.

+ Add New

Figure: Configuration of a new cloud connector

Configured connector instances are shown in a grid with basic information:

- Project name: Descriptive text for the instances.
- Status: Active, Not-Active and Error.
- Date: Cloud service connector instance creation date.

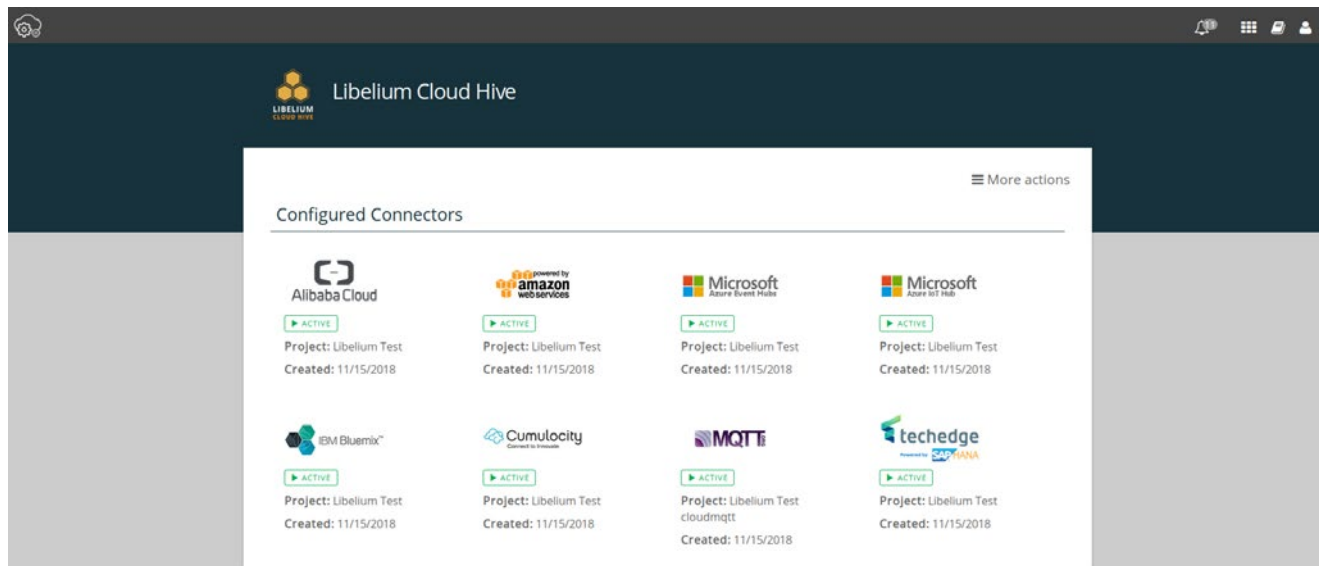


Figure: Configured cloud connectors list

3.1. New Connector Instance

Adding a new cloud service connector instance is a standard process for all different cloud connectors available. There is a first tab for the configuration parameters, a second tab to associate the sensor nodes that will feed the connector with data, and a 3rd tab to check the logs messages received in the Libelium Cloud Bridge service from the destination cloud service.

3.1.1. Configuration

The first step for adding a new cloud service connector instance is to configure the Libelium Cloud Bridge service with the valid user credentials needed to connect and synchronize data to the cloud service. Remember that this information must be given by the final cloud server (eg, AWS); Libelium just does not own or operate those final clouds. Most of those cloud services require to finalize a license purchase process to obtain the required parameters needed to properly authenticate and accept data from the Libelium Cloud Bridge service.

Configuration parameters are not exactly the same for all the cloud services, most of them include user and password authentication credentials combined with some API key for the service. They will be properly explained in the specific section of this guide dedicated for each one.

[← CONFIGURED CONNECTORS](#) > CONNECTOR CONFIGURATION

☰ More actions

[✎ Configuration](#) [🔗 Associated devices](#) [📄 Logs](#)



Project Name:

✖ NON ACTIVE

Root-Ca Key

Private Key

Certificate

Host

🗑 REMOVE CONNECTOR

You can always add it again later.

Figure: Example of configuration parameters form

3.1.1.1. Checking Configuration

Press the “Check” button to test the connection with the cloud service using the values filled in the form for the required parameters. Error messages in the Log tab should be monitored.



Figure: Check configuration button

When the connection with the cloud service is successful, the “Start Sync” button will be enabled. Click the “Start Sync” button to activate the instance and start sending values to the cloud service.



Figure: Start sync button

3.1.2. Associated Devices

Associating sensor nodes to the cloud service connector instance is the most relevant step in the configuration of the Libelium Cloud Bridge service. As explained in the introduction of this guide, the Libelium Cloud Bridge service receives data from the sensor nodes in three different ways (direct HTTPS connections, Meshlium connector, and LWPAN backend callbacks). All this data received is buffered waiting to be synchronized to one or several cloud services.

Associating sensor nodes to a cloud service connector instance is the way for the Libelium Cloud Bridge service to know what buffered data must be sent to the cloud service connector instances.

The simplest configuration would be to send all data from all sensor nodes to a single cloud service connector instance, but we may want to group sensor nodes for different projects and send data to different cloud services.

In the “Associated devices” tab a table is shown, listing all the information related to the registered sensor nodes.

[CONFIGURED CONNECTORS](#) > ASSOCIATED DEVICES

More actions

Configuration Associated devices Logs

☐ ADD ALL NODES TO CONNECTOR 2 devices selected

▶ ACTIVE

Icon	Device Name	Serial	Last value synced:	Status
	Waspmote :1, 1	Serial: 1	unknown	✓
	Waspmote :1, 2	Serial: 2	unknown	✓
	Waspmote :1, 3	Serial: 3	unknown	
	Waspmote :1, 4	Serial: 4	unknown	
	Waspmote :1, 5	Serial: 5	unknown	
	Waspmote :2, 1	Serial: 1	unknown	
	Waspmote :2, 2	Serial: 2	unknown	
	Waspmote :2, 3	Serial: 3	unknown	
	Waspmote :2, 4	Serial: 4	unknown	
	Waspmote :3, 1	Serial: 1	unknown	
	Waspmote :3, 2	Serial: 2	unknown	
	Waspmote :3, 3	Serial: 3	unknown	
	Waspmote :3, 4	Serial: 4	unknown	

🗑️ REMOVE CONNECTOR
You can always add it again later.

Save

Figure: Associated devices list

Data from the selected sensor nodes will be sent to the cloud service using the credentials configured in the instance.

Click “Add all nodes to connectors” to select all sensor nodes at once.



Figure: Select all devices

Click “Remove all nodes to connector” to clear all sensor nodes selected.



Figure: Deselect all devices

3.1.3. Logs

The “Logs” tab shows the information and error messages registering the result of the communication requests of Bridge while trying to send data to the cloud service. Different communication protocols are used to send data to the cloud services, messages related to the transport layer of the protocol can be identified and will be marked as INFO or ERROR. User intervention may be required to analyse the meaning of the messages generated in the application layer and decide if some actions are required.

An example of different types of messages is shown in the following image.

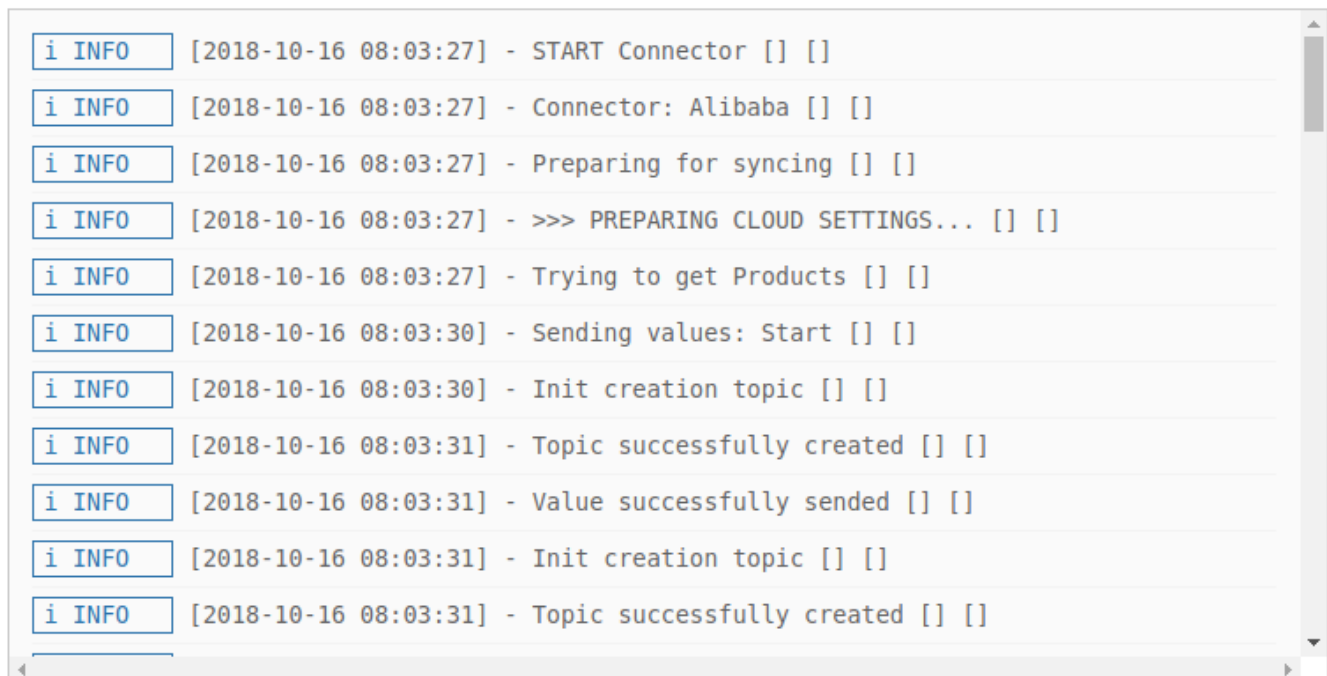


Figure: Log view

Click the “Refresh” button to refresh the logs view and load any new message available, latter messages will appear at the bottom.



Figure: Log refresh

Different types of log messages are identified depending on the response of the cloud service. A response is considered an error when the transport layer of the protocol returns an identified error code. A message with the "ERROR" tag will be displayed.

HTTPS transport layer error codes are:

- Client error (4xx), meaning that the request contains bad syntax or cannot be fulfilled
- Server error (5xx), meaning that the server failed to fulfill an apparently valid request

! ERROR [2018-11-15 11:50:47] - Exception in sync: Error connecting

Figure: Log error message

Any other message will be tagged as "INFO", showing the text received in the message from the application layer.

i INFO [2018-10-16 08:03:35] - Value successfully sended [] []

Figure: Log info message

3.2. Connector Status

The properly configured cloud service connector instances send periodically to the cloud services the data buffered that has been received from the sensor nodes. The results of the communication requests can be monitored with the cloud service connector instance status.

Cloud service connector instance states:

- **Active:** The configuration is active when all fields are correct and there are not errors syncing the values.
- **Non Active:** The connector is non active when one or more parameters are missing or misconfigured.
- **Error:** This status is displayed when a connector has been configured correctly and an error occurred in the data synchronization.



▶ ACTIVE

Project: Libelium Test

Created: 11/13/2018



✕ NON ACTIVE

Project:

Created: 11/14/2018



✕ ERROR

Project:

Created: 11/14/2018

Figure: Status of configured connectors

3.3. Removing a Connector Instance

Cloud service connector instances can be removed anytime: click the “Remove Connector” button to erase the current connector instance.



Figure: Remove configured connector

Only the configuration of this instance will be deleted, it will not affect the data previously sent to the cloud service. The same configuration parameters may be used to configure the cloud service in the future and data will be sent again.

Removing the last cloud service connector instance (or the only one) may cause in filling the Libelium Cloud Bridge service buffer because no data will be synchronized (the Bridge deletes overbuffer data). When the buffer is full, the Libelium Cloud Bridge service will not accept any more data, so sensor nodes trying to send data to the Libelium Cloud Bridge service when its buffer is full will receive communication errors.

3.4. Configuration Parameters

The user must configure at least one cloud service connector instance to use the Libelium Cloud Bridge service. Each one of the cloud services available in the Libelium Cloud Bridge service (Alibaba Cloud, AWS IoT, Cumulocity, IBM Bluemix, Microsoft Azure Event Hubs, Microsoft Azure IoT Hubs, MQTT, SAP Hana and Telefónica IoT Cloud) requires different parameters.

3.4.1. Alibaba Cloud

The screenshot shows the 'CONNECTOR CONFIGURATION' page for the 'Alibaba Cloud' connector. The page has a breadcrumb 'CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION' and three tabs: 'Configuration' (active), 'Associated devices', and 'Logs'. On the right, there's a 'More actions' menu with the Alibaba Cloud logo and a 'NON ACTIVE' status. Below the logo is a 'REMOVE CONNECTOR' button with the text 'You can always add it again later.' The configuration form on the left includes: 'Project Name' (text input), 'Access Key Id' (text input), 'Access Key Secret' (text input), and 'Region' (a dropdown menu currently showing '<Nothing selected>').

Figure: Alibaba Cloud connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- Access Key ID: Key ID provided in the registration process
- Access Key Secret: Key Secret provided in the registration process
- Region: Region provided in the registration process

Click on the "Access Key" section of the profile account to get your access keys.

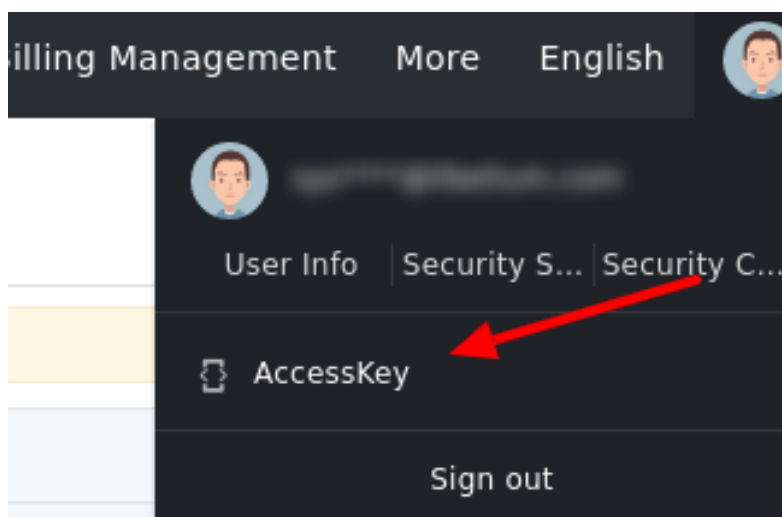


Figure: Access key

Once you are inside, press the "Create Access Key" button on the "Security Management" section.

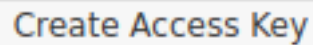


Figure: Create access key

When it has been created correctly, a pop-up window will appear with the data of the key that has just been generated.

Create User Access Key

✕

CAUTION: The user Access Key must be downloaded immediately. It will not be available after this session.

✓

Access Key created.

AccessKey Details

^

AccessKeyID :
LTAISg4afRXFPCIE

AccessKeySecret :
n9107V1XMiNxsl0FlmhVg07ATEpNR7

Save AccessKey Information

Figure: Access key details

If you want to store the information on your computer, you only have to press the "Save Access Key Information" button and a file with csv extension will automatically be saved on your PC with the key data generated.



Figure: Save access key

After this you can consult all your generated keys in the panel. To see the secret key, press simply in the button "Show".

User Access Key					Create Access Key
AccessKey ID	Access Key Secret	Status	Time Created	Action	
LTAGp4m5m55	Show	Enable	2020-11-10 10:00:00	Disable	Delete
LTAGp4m5m55	Show	Enable	2020-11-10 10:00:00	Disable	Delete

Figure: Show access key secret

3.4.2. AWS IoT

CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION

More actions

powered by
amazon
web services

NON ACTIVE

REMOVE CONNECTOR
You can always add it again later.

Configuration Associated devices Logs

Project Name:

Root-Ca Key

Private Key

Certificate

Host

Figure: AWS IoT connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- Root-Ca Key: Authority of the certificate to be used
- Private Key: Private key of the certificate to be used
- Certificate: Certificate to be used
- Host: Host that the certificate has been issued for

First of all you had to create a "Thing" in the "Manage" → "Things" section on the AWS IoT Service. When you have created your own thing, click on the window with the name of the thing you just created.

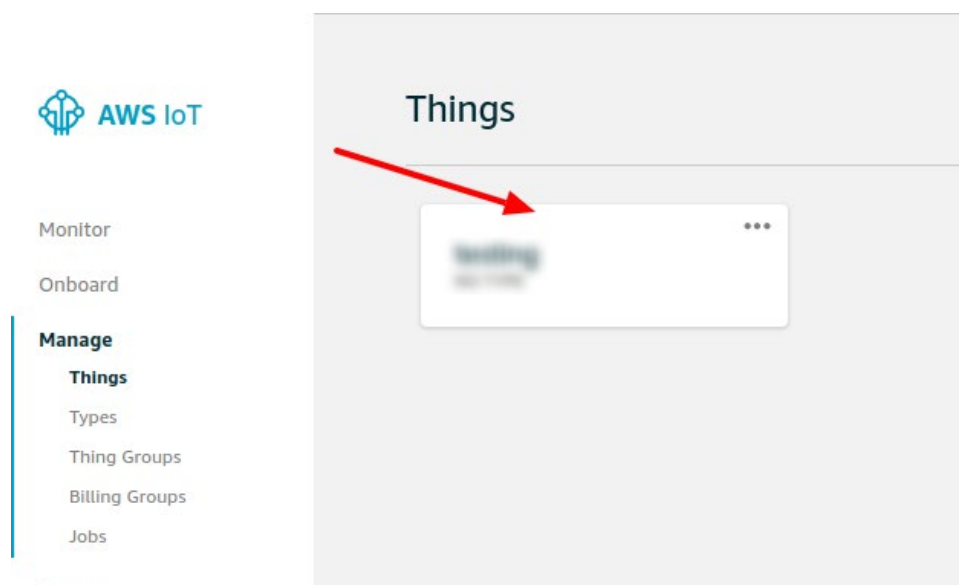


Figure: Create a Thing

To get the HOST parameter you will have to go to the “Interact” section and in the “HTTPS” section you will see the HOST that you will have to copy in the connector configuration.

HTTPS

Update your Thing Shadow using this Rest API Endpoint. [Learn more](#)

a [redacted] -ats.iot.eu-west-2.amazonaws.com

Figure: Host parameter

To get the Private Key and Certificate if you do not have them previously created, you will have to go to the “Security” section and then press the “Create Certificate” button.

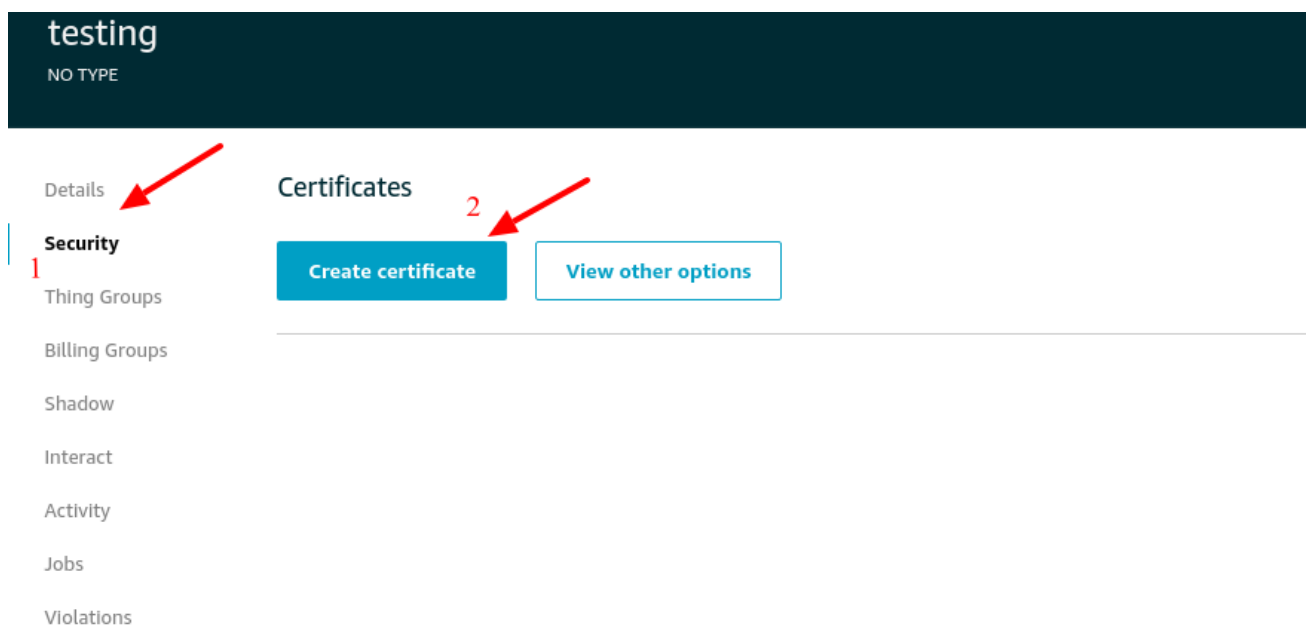


Figure: Create certificate

Download the three keys displayed before closing the window because they will never be shown again.

In order to connect a device, you need to download the following:

A certificate for this thing	[redacted].cert.pem	Download
A public key	[redacted].public.key	Download
A private key	[redacted].private.key	Download

Figure: Download certificates

Click “Download” to save the root CA certificate.

You also need to download a root CA for AWS IoT:

A root CA for AWS IoT **Download** 

Figure: CA certificate download

Click the “Activate” button for assigning the keys to the Thing that you have just created. Click the “Done” button to finish the process and close all windows.



Activate

Figure: Activate certificates

Copy the content of each downloaded file in their respective fields.

3.4.3. Cumulocity

[CONFIGURED CONNECTORS](#) > CONNECTOR CONFIGURATION

More actions

[Configuration](#) [Associated devices](#) [Logs](#)



NON ACTIVE

REMOVE CONNECTOR

You can always add it again later.

Available tags:

```
#VARIABLE#
#TS#
#WASP_NAME#
#WASP_SERIAL#
#MESHLIUM_NAME#
#MESHLIUM_SERIAL#
#VALUE_POSITION_LAT#
#VALUE_POSITION_LONG#
#VALUE_POSITION_H#
#ID#
#ID_WASP#
#ID_SECRET#
#SENSOR#
#VALUE#
#MESHLIUM#
```

Project Name:

Tenant

Port Number

User

Password

Figure: Cumulocity connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- Tenant: The ID of your application
- User: The user that you use to log in to the Cumulocity Cloud
- Port Number: The port number of your tenant application
- Password: The password that you use to log in to the Cumulocity Cloud

3.4.4. IBM Bluemix

The screenshot shows the 'CONNECTOR CONFIGURATION' page for the IBM Bluemix connector. The page has a breadcrumb trail: 'CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION'. Below this, there are three tabs: 'Configuration' (selected), 'Associated devices', and 'Logs'. On the right side, there is a 'More actions' menu icon and the IBM Bluemix logo. Below the logo, there is a 'NON ACTIVE' status indicator and a 'REMOVE CONNECTOR' button with a warning icon. The main form area contains five input fields, all of which are currently empty: 'Project Name:', 'Organization', 'API user', 'API password:', and 'Event ID:'. The 'API password:' field has a small eye icon to toggle visibility. The 'REMOVE CONNECTOR' button has a tooltip that says 'You can always add it again later.'

Figure: IBM Bluemix connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- Organization: The organization ID of your dashboard
- API Password: The Authorization Token generated
- API User: The API key generated
- Event ID: Field used to configure the event where you want to send the information, if you do not know what to type in this field, you can use EID as value

Go to Access API Keys menu, and click on the "Generate API Key" button, on the bottom left side. Select "Backend Trusted Application" as API role. You will get the API Key and Authentication Token. It is very important to write them down (save them on your PC), as there is no way to request them and they will be used later.




Figure: Generate API Key

Generate API Key

Authentication tokens are non-recoverable. If you misplace this token, you will need to re-register the API key to generate a new authentication token.

Select API Role(s)

[What are they?](#)

 Add another role

Comment

Set API key expiry



Cancel

Generate

Figure: New API Key form

3.4.5. Microsoft Azure Event Hubs

◀ CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION

More actions

Configuration Associated devices Logs

Project Name:

Namespace

Directive name:

Directive key

name

Template file:

Microsoft Azure Event Hubs

✖ NON ACTIVE

REMOVE CONNECTOR

You can always add it again later.

Figure: Microsoft Event Hubs connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Namespace: Name of the Event Hub where you want to send data to.
- Directive Name: Name of the shared access policies.
- Directive Key: Key of the policy.
- Name: Name assigned on the entity.
- Template File: File with the template.

Create in the Microsoft panel a new Event Hub. The name assigned to the Event Hub will be the name to be provided in the "Namespace" field of the connector configuration.

Create Event Hub □ ×

Event Hubs

* Name ⓘ

Partition Count ⓘ

Message Retention ⓘ

Capture ⓘ

On Off

Figure: Event hub namespace

Create a Policy on "Shared access policies" of the "Settings" section. The name of the shared access policies will be the name to be provided in the "Directive Name" field of the connector configuration.

myeventhub - Shared access policies
Event Hubs Instance

Search (Ctrl+/)

Overview

Access control (IAM)

Diagnose and solve problems

Settings

Shared access policies

Properties

Locks

Automation script

Entities

Consumer groups

Features

Capture

Support + troubleshooting

New support request

+ Add

Search to filter items...

POLICY

Figure: Event hub directive

Click in one of the policies that you have created to get the “Directive Key”. A new window will be opened on the right with the “Primary Key” parameter, that is the value needed to fill the “Directive Key” field of the connector configuration.

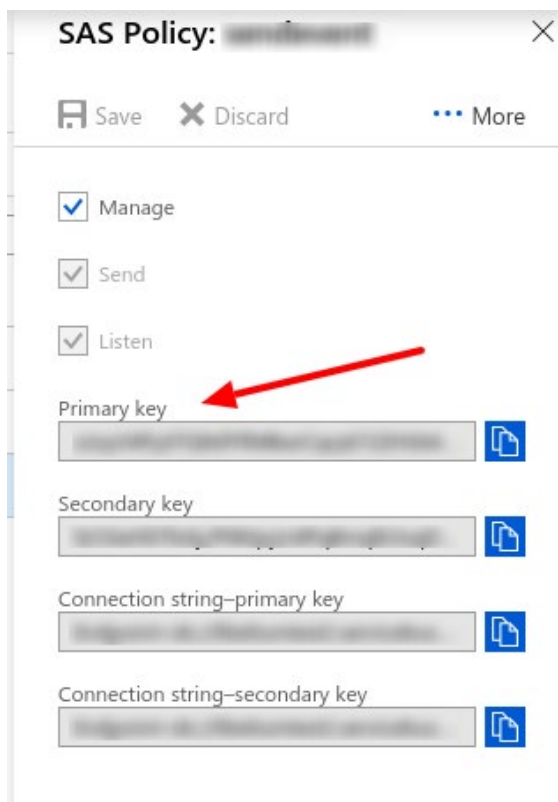


Figure: Event hub policy key

Add a new entity in the “Entities” section of the newly Event hub created. The name assigned to the new Entity will be the “Name” field of the connector configuration.

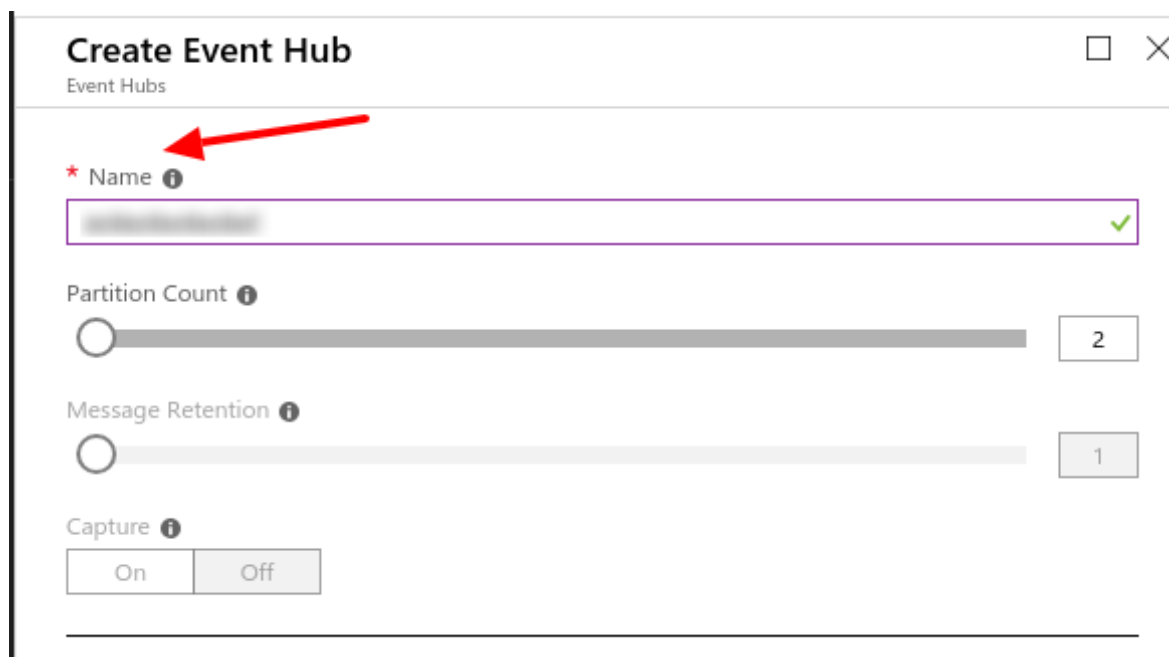


Figure: Event hub entity name

3.4.6. Microsoft Azure IoT Hubs

← CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION

≡ More actions

Configuration Associated devices Logs

Project Name:

Connection String

Number Requests:

Sync Interval

Log Level



✖ NON ACTIVE

REMOVE CONNECTOR

You can always add it again later.

Figure: Microsoft IoT Hub connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- Connection String: String to use in the connector

Create an item on the "IoT Hub" section and click on the link with the name of the item you have just created.

IoT Hub				
Default Directory				
+ Add Edit columns Refresh Assign tags				
Subscriptions: Pay As You Go?				
Filter by name...	All resource groups	All locations	All tags	No grouping
1 items				
NAME	TYPE	RESOURCE GROUP	LOCATION	SUBSCRIPTION
my-iot-hub	IoT Hub	my-resource-group	North Europe	Pay As You Go?

Figure: IoT hub

Click on the “Settings” section of “Shared access policies”.

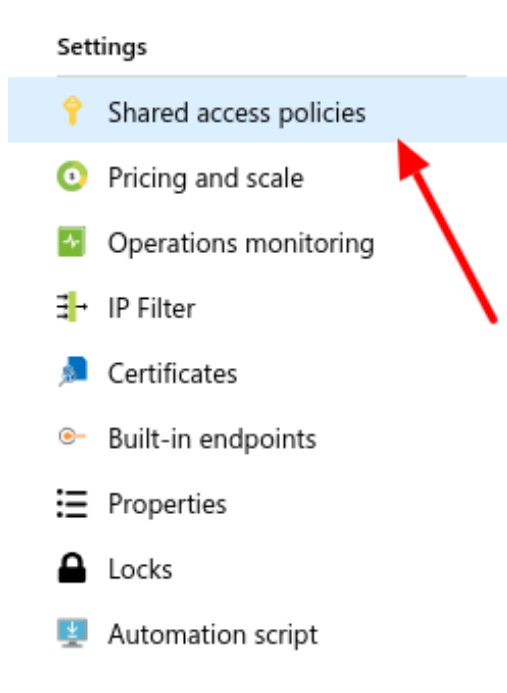


Figure: Shared policies

Click the “Add” button to assign the name and the permissions needed. Click on the name and a window will appear on the right with the “Connection String”, which you will have to copy and paste in the configuration of your connector.

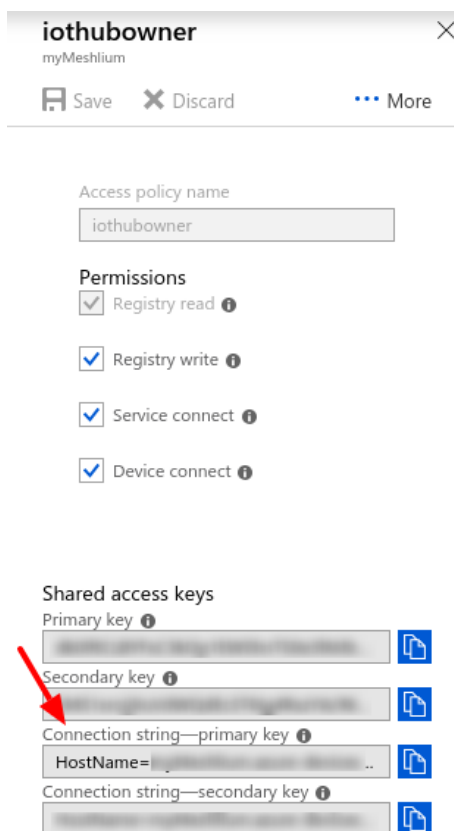
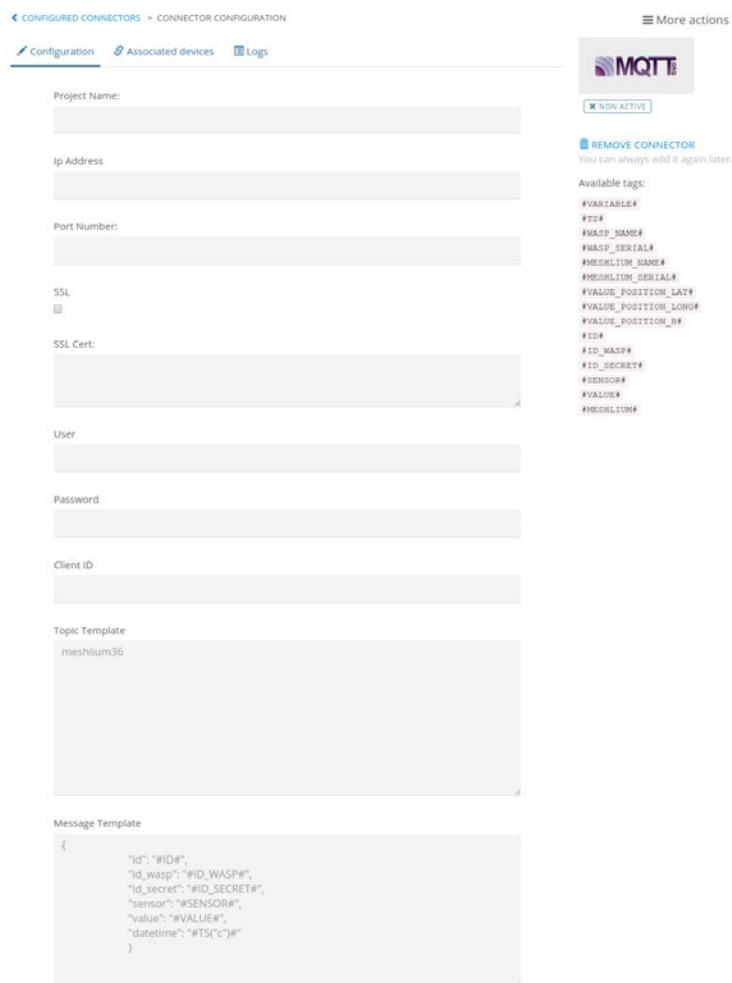


Figure: Connection string

3.4.7. MQTT



CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION

Configuration Associated devices Logs

Project Name:

IP Address:

Port Number:

SSL

SSL Cert:

User:

Password:

Client ID:

Topic Template

meshlium36

Message Template

```
{
  "id": "#ID#",
  "id_wasp": "#ID_WASP#",
  "id_secret": "#ID_SECRET#",
  "sensor": "#SENSOR#",
  "value": "#VALUE#",
  "datetime": "#TSLC"
}
```

More actions

MQTT

NON ACTIVE

REMOVE CONNECTOR
You can always add it again later.

Available tags:

- #VARIABLE#
- #TS#
- #WASP_NAME#
- #WASP_SERIAL#
- #MESHLIUM_NAME#
- #MESHLIUM_SERIAL#
- #VALUE_POSITION_LAT#
- #VALUE_POSITION_LONG#
- #VALUE_POSITION_H#
- #ID#
- #ID_WASP#
- #ID_SECRET#
- #SENSOR#
- #VALUE#
- #MESHLIUM#

Figure: MQTT connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- IP Address: The ip address of your application
- Port: The port number of your application
- SSL & SSL Cert: Certificate for secure communications
- User: The user that you use to log in to the MQTT Cloud
- Password: The password that you use to log in to the MQTT Cloud
- Client ID: The Client ID of your application
- Topic Template: Template for the MQTT topic
- Message Template: Template for the MQTT message

3.4.8. SAP Hana

← CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION

Configuration Associated devices Logs

HCP Configuration

Project Name:

Device Token

Device ID



MMS Endpoint

Message ID

Connector Configuration

Meshlium Name

More actions

 **techedge**
Powered by  **SAP HANA**

NON ACTIVE

REMOVE CONNECTOR
You can always add it again later.

Figure: Figure: SAP Hana connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- Device token: Your device token
- Device ID: Your device ID
- MMS Endpoint: Endpoint to be used.
- Message ID: ID of the message
- Meshlium Name: Meshlium name

3.4.9. Telefónica IoT Cloud

< CONFIGURED CONNECTORS > CONNECTOR CONFIGURATION

[Configuration](#) [Associated devices](#) [Logs](#)

More actions

Telefónica

✖ NON ACTIVE

[REMOVE CONNECTOR](#)

You can always add it again later.

Available tags:

#VARIABLE#
#TS#
#WASP_NAME#
#WASP_SERIAL#
#MESHLIUM_NAME#
#MESHLIUM_SERIAL#
#VALUE_POSITION_LAT#
#VALUE_POSITION_LONG#
#VALUE_POSITION_H#
#ID#
#ID_WASP#
#ID_SECRET#
#SENSOR#
#VALUE#
#MESHLIUM#

Project Name:

Server

Port

Api

Check

Start Sync

Figure: Telefónica IoT Cloud connector configuration parameters

Complete the information in the form (all the fields are mandatory):

- Project Name: Your project's name
- Server: Host name
- Port: The port number of your application
- API: The API key of your user panel account

4. Manage Your Devices

The Libelium Cloud Bridge service shows in this section all your registered sensor nodes. You can manage two different keys in this section.

More actions

- ✓ Configure your Connectors
- ☰ Gateway Configuration
- 🔍 Manage your Devices

Connector	Status	Project	Created
powered by amazon web services	ACTIVE	Project: Libelium Test	Created: 11/28/2018
Microsoft Azure Event Hubs	ACTIVE	Project: Libelium Test	Created: 11/28/2018
MQTT	ACTIVE	Project: Libelium Test	Created: 11/28/2018
Cumulocity	ACTIVE	Project: Libelium Test	Created: 11/28/2018
techedge	ACTIVE	Project: Libelium Test	Created: 11/28/2018

Figure: Manage your devices menu

4.1. API Keys

API Keys are needed to identify the sensor nodes with the user account and the type of license associated to it. A valid API Key is mandatory to establish a communication with the Libelium Cloud Bridge service.

In other words, API Keys are used to authenticate the calls to the Libelium Cloud Bridge service. Sensor nodes need to include a valid API Key in the headers of their HTTPS calls to successfully connect with the Libelium Cloud Bridge service.

The “Personal Access Keys” section shows all the API Keys which are currently active.

Personal Access Keys			Create New Key
Name			
Smart City :: South Carriefurt		Show	Delete
Smart City :: Danielshire		Show	Delete

Figure: Creating a new key

API Keys are transparently generated when using the Programming Cloud Service. As soon as an API Key is generated by the Programming Cloud Service, it is stored in the Libelium Cloud Bridge service, being listed in the “Personal Access Keys” section.

Select communication module *

WiFi

ESSID

libelium_AP

Security

Select

IP method

Select

Select protocol and destination *

Send to Libelium Cloud Hive

Use Payload Encryption (AES 256)

☐

Figure: Use the Programming Cloud Service to send data to Libelium Cloud Bridge service

The Programming Cloud Service includes the API Key of each node in the binary programs generated. In the case of programming the sensor using Libelium Waspnote IDE, the API Key needs to be copied from this section and then pasted in the source file.

4.1.1. Creating The Keys

If the Programming Cloud Service is not used for programming the sensor nodes, an API Key is still needed. In this case the user must manually create this API Key. The user has to insert it in the source code written on the Wasp mote IDE.

Click the "Create New Key" button to manually create a new API Key. Manually created API Keys are also listed in the "Personal Access Keys" section.

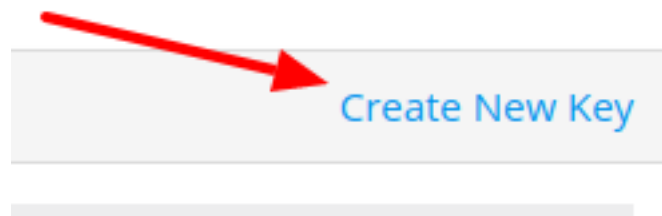


Figure: Link for creating a new key

When creating a new API Key, the user must provide a value in the "Name" field. This is a user-friendly name to tag the API Keys of your devices. Avoiding duplicated names is recommended, because this descriptive text may be used later for identifying the API Keys.

Create key

Name

Smart City :: South Carrieffurt

Close

Create

Figure: Name field for the new key

A pop-up window will rise showing the generated API Key. The user may copy it now to use in the source code when needed. The API Key may be retrieved anytime later.

Personal Access Token



Here is your new personal access token. You may now use this token to make API requests.



Close

Figure: Personal Access Key token

4.1.2. Show And Delete Existing API Keys

The API Key that you have just manually created is listed on the “Personal Access Keys” section. You can access to each API Key by pressing the “Show” button.

API keys

Personal Access Keys		Create New Key
Name		
Smart City :: South Carriefurt	Show	Delete
Smart City :: Danielshire	Show	Delete

Figure: Personal Access Key token

If you want to delete an API Key, find the API Key that you want to delete in the “Personal Access Key” list and press on the “Delete” button in the same row.

Personal Access Keys		Create New Key
Name		
Smart City :: South Carriefurt	Show	Delete
Smart City :: Danielshire	Show	Delete

Figure: Personal access token

A message to confirm that you want to delete the API Key will be shown. Remember that after deleting the API Key, it will not be possible to recover it again. Once you delete the API Key, the access to the Libelium Cloud Bridge service will be revoked for the devices using that API Key.

Delete API Key



Once you delete the API Key you revoke the access to the Libelium Cloud Hive to the devices which are using this API Key

Cancel

Confirm

Figure: Personal access token

4.2. Encryption Keys

If the user wants to encrypt the payload of frames, an encryption key for each node must be created previously. Encryption keys for the payload (AES-256) are needed when the sensor nodes are programmed to send data values to the final cloud services using end-to-end encryption.

These encryption keys are only useful when the user wants to implement an end-to-end security layer in the final cloud service.

Note: Remember that the Libelium Cloud Bridge service already implements an AES-128 encryption. This encryption is done by default by the Programming Cloud Service in a transparent way for the user. If writing code in the IDE, the user can call a simple function. In either of these two ways, frames travel secured from the sensor node to the Libelium Cloud Bridge service.

4.2.1. Change Key

Click the “Change key” button to edit the encryption key. In the pop-up window shown there is information about the the properties and values of the encryption key of each sensor node. In the top-left of the window the device name and the device serial id are shown, these two values identify the sensor node.

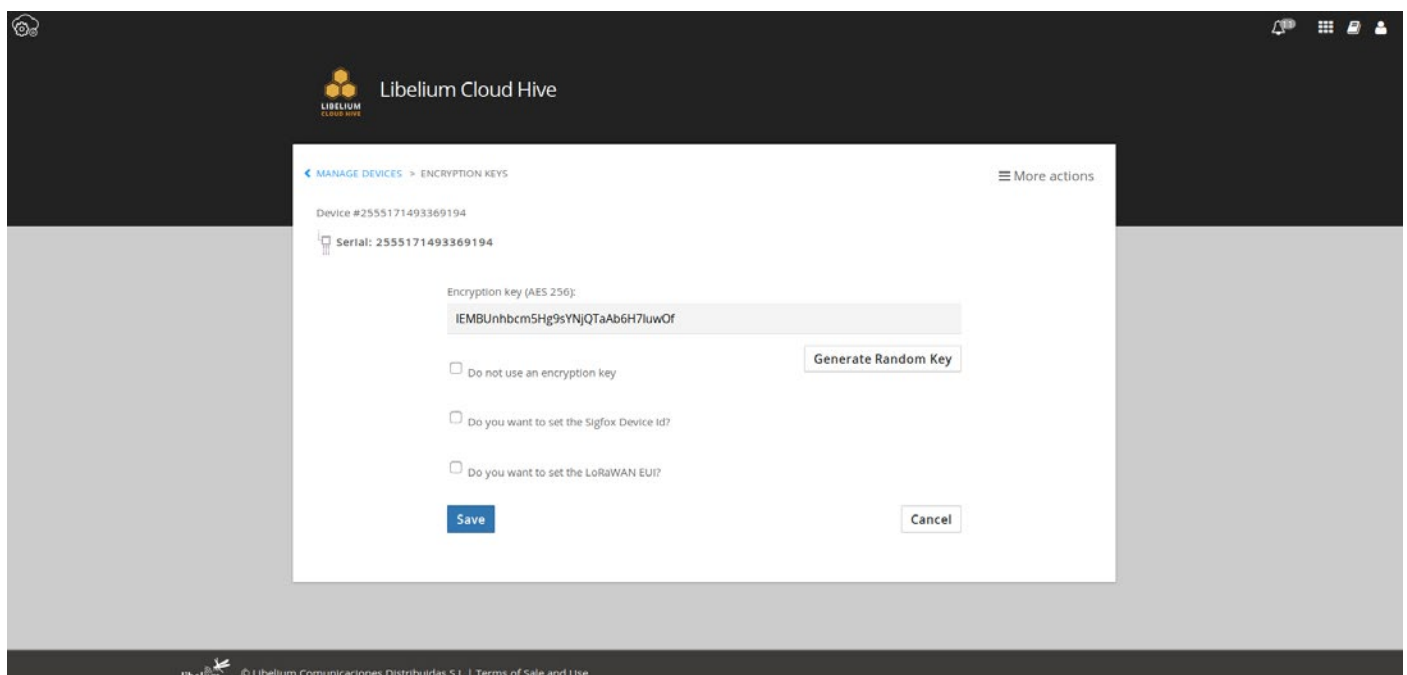


Figure: Edit encryption key

The current value of the AES-256 encryption key of the sensor node is shown in the field “Encryption key (AES 256)”, an initial random encryption key is assigned to each node in the activation process of the Services Cloud Manager. This value will be used by the Programming Cloud Service when choosing the option to encrypt the frames for this sensor node. If writing code in the IDE, the user should copy the value from this field to the source code.

The value of the AES-256 encryption key assigned to the node may be modified: the field “Encryption key (AES 256)” is editable and any text that complies with the AES-256 requirements may be typed, using a custom value or generating a new random AES-256 encryption key.

Click the button “Generate Random Key”, next to the editable field, and a new valid AES-256 encryption key value will be presented in the editable field. It is possible to click the button as many times as desired to get different values. Finally, click the button “Save” at the bottom to assign the new value to the sensor node.

AES-256 encryption keys may be edited as well from the “My Devices” section of the Services Cloud Manager, click the pencil icon on the “Action” column to open the edit window and check or modify the current encryption key assigned to each sensor node.

Edit Node
 ×

Name:

Project:

Description:

Encryption key (AES 256):

☐ Do not use an encryption key

☐ Do you want to set the Sigfox Device Id?

☐ Do you want to set the LoRaWAN EUI?

Figure: Personal access key

Figure: Generate a new random key to overwrite the existing key is also possible.

Edit Node
 ×

Name:

Project:

Description:

Encryption key (AES 256):

☐ Do not use an encryption key

☐ Do you want to set the Sigfox Device Id?

☐ Do you want to set the LoRaWAN EUI?

Figure: Change encryption key

5. Sending Data Frames

5.1. Data Streaming

The Libelium Cloud Bridge service receives data frames from the sensor nodes and sends all the data values to the different final cloud services configured of the list of available connectors: Alibaba Cloud, AWS IoT, Cumulocity, IBM Bluemix, Microsoft Azure Event Hubs, Microsoft Azure IoT Hubs, MQTT, SAP Hana and Telefónica IoT Cloud.

5.1.1. Receiving Data

Data may be received with direct HTTPS connections from the sensor nodes, using the Libelium Cloud Bridge service connector available in the Meshlium gateway and configuring the callback services of the LPWAN providers.

5.1.1.1. HTTPS Connections

Sensor nodes using NB-IoT / Cat-M, 4G and WiFi may be programmed to send data directly to the Libelium Cloud Bridge service.

Using Programming Cloud Service

The Programming Cloud Service includes the option “Send to Libelium Cloud Bridge” in the section “Select protocol and destination”. The valid API Key of the sensor node and all the encryption layers are implemented transparently by default in the binary program that the Programming Cloud Service generates.

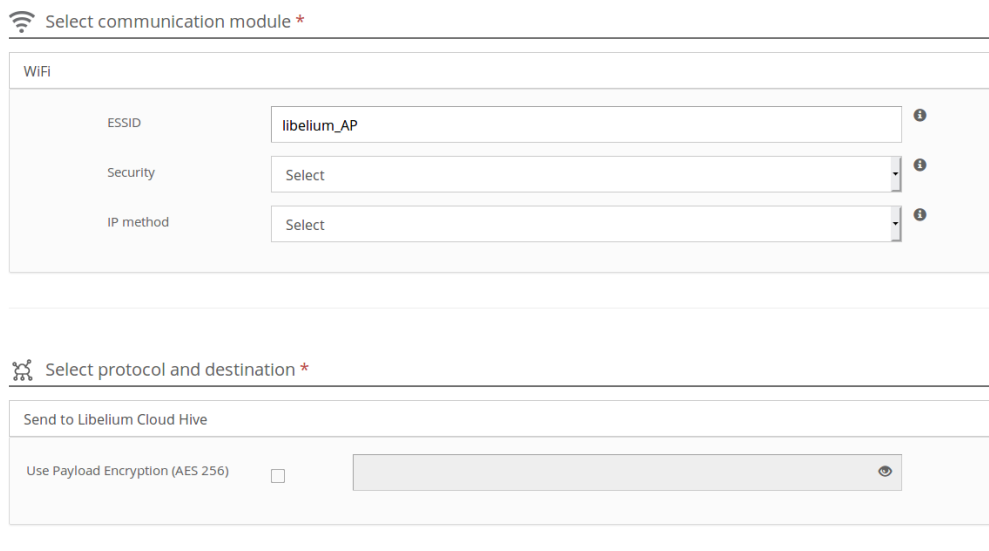


Figure: Programming Cloud Service , send to Libelium Cloud Bridge

Using Waspnote IDE

In the Waspnote API documentation there is a sketch available (the HTTPS example) that may be used for the user as the basis for coding on the Waspnote IDE, a valid call to the Libelium Cloud Bridge service.

Parameters needed for the HTTPS call to the Libelium Cloud Bridge service are:

- **Host:** hw.libelium.com
- **Action:** GET /hw/ps?frame=[FRAME TO SEND] HTTP/1.1
- **Authorization:** Bearer [API KEY FROM Bridge]

5.1.1.2. Meshlium Gateway Connector

Meshlium gateways can be configured to send data to the Libelium Cloud Bridge service. Meshlium gateways that are going to be used to send data to the Libelium Cloud Bridge service have to be registered in the Services Cloud Manager using their associated activation code.

IoT Premium > Libelium Cloud

Configuration
Log

User:
 Password:
 Enable Remote Management: ☒ ON



You don't have registered your meshlium , please registered on this [link](#)


Figure: Meshlium not registered message


The Libelium Cloud Bridge service connector must be configured with the same user account used to register the Meshlium gateway in the Services Cloud Manager. The Libelium Cloud Bridge service connector can be found in the section "Cloud Connectors" → "IoT Premium" of the Manager System.

IoT Premium > Libelium Cloud

Configuration
Log

User:
 Password:
 Enable Remote Management: ☒ ON




Cloud Synchronization Status

WASPMOTES

Figure: Libelium Cloud Bridge connector configuration

The Meshlium gateway will make the HTTPS request to the Libelium Cloud Bridge service sending the data gathered from the nodes that are linked to the Meshlium gateway using RF communication protocols, 4G or WiFi.

5.1.1.3. Sigfox

Sigfox deploys a Low Power Wide Area Network in many countries, and Sigfox owns the infrastructure. The Libelium Cloud Bridge service enhances the Sigfox network features giving access out-of-the-box to send the data to any of the final cloud services available (Alibaba Cloud, AWS IoT, Cumulocity, IBM Bluemix, Microsoft Azure Event Hubs, Microsoft Azure IoT Hubs, MQTT, SAP Hana and Telefónica IoT Cloud).

The user must have a valid Sigfox account to integrate the Libelium Cloud Bridge service in the Sigfox backend as a callback. Registering new Sigfox accounts can be done at <https://build.sigfox.com/sign-up>.

Sigfox Backend Configuration

As a mandatory process for all Sigfox accounts, the user must register a new device and a new device type.

Steps for registering a new device can be followed on this address:

<https://resources.sigfox.com/document/register-a-device>

As a summary, the user should do the following steps:

- Click on the “Device” menu
- Click on the “New” button in the upper right part of the screen
- Choose the desired group
- Complete the device information fields:
 - ID/PAC couple, location information is optional
 - Name
 - Select device type
- if your device is Sigfox Ready, Enter your product certificate
- The device default setting is “Activable”. If this option is disabled, all messages will be dropped and the device cannot consume any token. This option could be updated by editing the device.
- Click “OK” to validate.

Steps to create a new device type are also simple. On the following URL it is explained how to do that:

<https://resources.sigfox.com/document/create-a-new-device-type>

Summary of the actions to be done by the user:

- Go to the “Device type” tab
- Click on “New” button in the top right corner
- Fill in the device type information:
 - Name of your device type
 - Description of your device type
 - Set the keep-alive value in minutes (optional)
 - Contract: Link your device type to an active BSS order
 - Alert e-mail: If a callback fails, an email will be sent to the address below so that you can take action to fix the problem
 - Configure the downlink callback mode
 - Configure the display type, cf: Downlink callbacks

Using the “Edit” button on the top right corner you can edit a device type.

Callback Configuration To Libelium Cloud Bridge Service

Go to the “Device” menu and click on the device type of your device. A new menu will open and on the left side a “CALLBACKS” option is displayed, click on it.

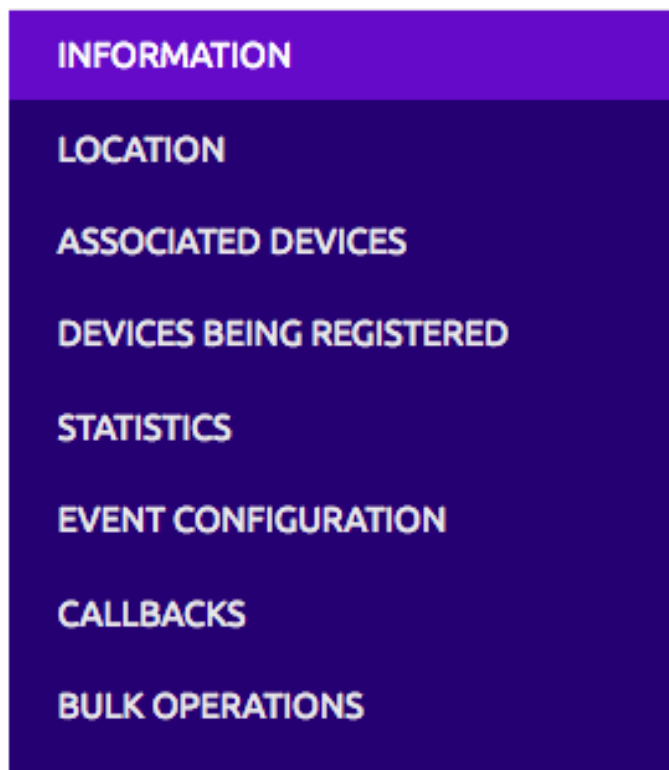


Figure: Sigfox callbacks menu

A new screen is displayed, there you can create a new callback request, For that, click on the “New” button on the top right corner and now you can type the data for a new callback request:

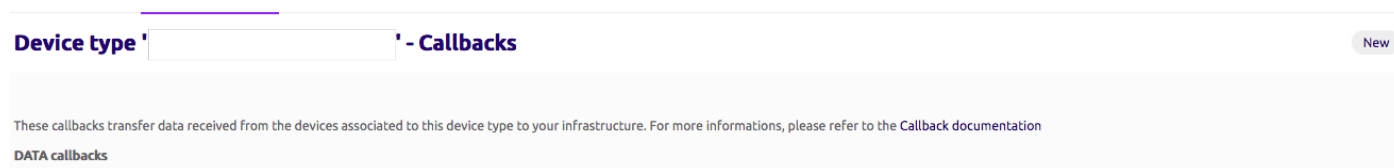


Figure: Sigfox new callback request

Select “Custom callback”, it is the first option on the options panel for a new kind of callback:

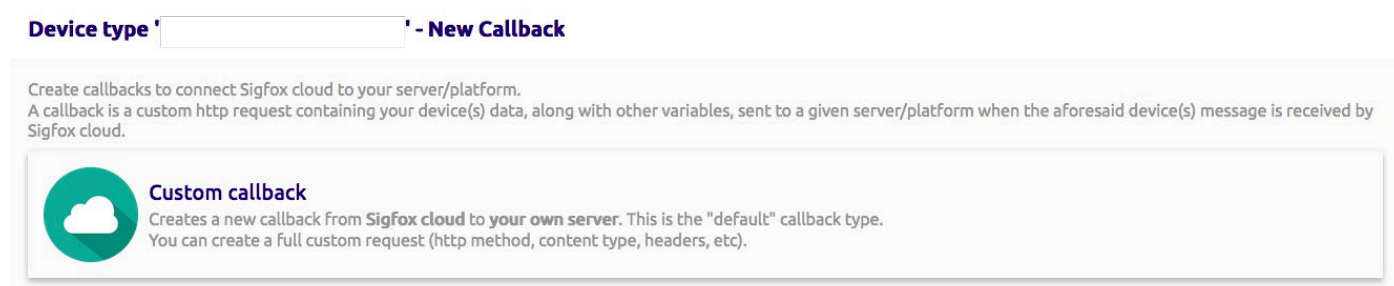


Figure: Sigfox custom callback option

The main configuration panel will display your custom callback:

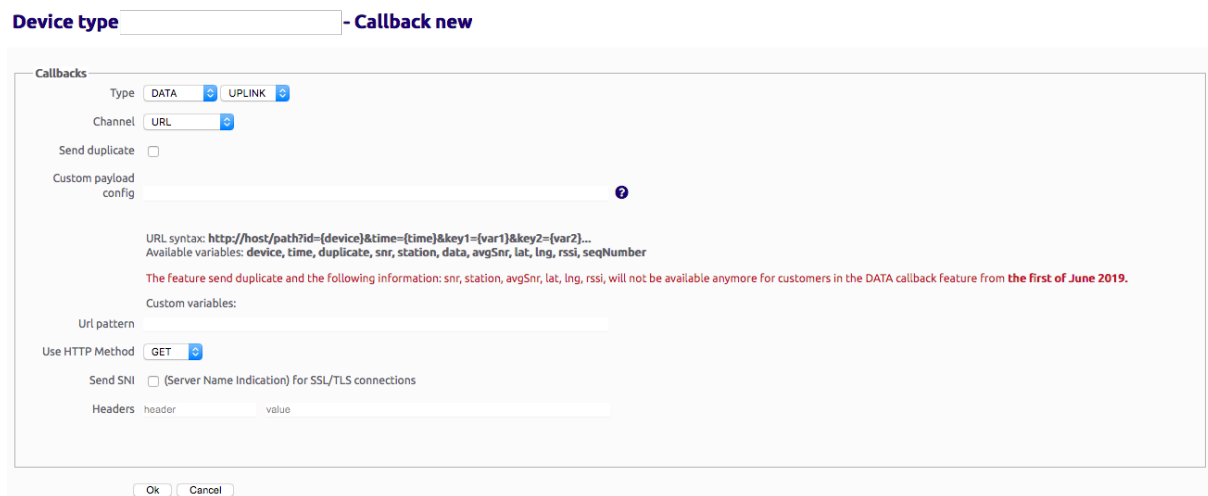


Figure: Sigfox custom callback panel

We need to understand this statements to clarify the values of the configuration creating the Sigfox custom callback for the Libelium Cloud Bridge service:

- There are three different callback types
- You have a set of available variables for each type of callback
- These variables are replaced by their value when a callback is called
- When receiving a callback, the client system must return an HTTP 2xx code within 10 seconds
- If the client system fails to process the callback during this time, an automatic retry will be scheduled 1 minute later

We will explain now the values chosen to properly create a Sigfox custom callback for sending data to the Libelium Cloud Bridge service:

- We will set the type to "DATA" and "UPLINK". As we want to upload the information to a given URL, the Channel option must be "URL".
- The URL pattern is the Libelium Cloud Bridge service endpoint for Sigfox callbacks, <https://hw.libelium.com/hw/sigfox>
- The request type is "POST". As we want to send data to a remote server, leave unchecked the "Send SNI" option.
- The Libelium Cloud Bridge service uses an API Key as the method to identify and authenticate to the user and the sensor node. Including the authentication API Key in the custom callback created is done adding to the headers option an authorization field with the value "Bearer LIBELIUM_CLOUD_TOKEN" (without quotes). The LIBELIUM_CLOUD_TOKEN can be copied from the Libelium Cloud Bridge service backend as explained in the section "API Keys" of this guide.
- The value for Content-type option must be application/json
- The body of the request will contain the data that will be sent to the Libelium Cloud Bridge service. It has to be short and properly formatted, JSON syntax will be used.
- Data fields that will be used to upload the information are:
 - device (string): device identifier (in hexadecimal – up to 8 characters <=> 4 bytes)
 - data (string): the user data (in hexadecimal)
 - time (int): the event timestamp (in seconds since the Unix Epoch)
 - seqNumber (int): the sequence number of the message, if available
 - duplicate (bool):
 - "true" if the message is a duplicate one, meaning that the backend has already processed this message from a different base station
 - "false" otherwise. To receive duplicate messages, you have to check the "send duplicate" box in the callback configuration page

- The Body field will be set with a JSON payload as the following one:

```
{
  "device_id" : "{device}",
  "data" : "{data}",
  "time" : "{time}",
  "seqNumber" : "{seqNumber}",
  "duplicate" : "{duplicate}"
}
```

- Click "Ok" to confirm changes and create the new callback.
- The callback configuration should be as follows:

Device type **- Callback edition**

Callbacks

Type:

Channel:

Send duplicate: ☐

Custom payload config:

URL syntax: [http://host/path?id={device}&time={time}&key1={var1}&key2={var2}...](#)
 Available variables: device, time, duplicate, snr, station, data, avgSnr, lat, lng, rssi, seqNumber
 The feature send duplicate and the following information: snr, station, avgSnr, lat, lng, rssi, will not be available anymore for customers in the DATA callback feature from **the first of June 2019**.

Custom variables:

Url pattern:

Use HTTP Method:

Send SNI: ☐ (Server Name Indication) for SSL/TLS connections

Headers:

header	value
Authorization	Bearer ey...AM

Content type:

Body:

```
{
  "device_id" : "{device}",
  "data" : "{data}",
  "time" : "{time}",
  "seqNumber" : "{seqNumber}",
  "duplicate" : "{duplicate}"
}
```

Figure: Sigfox custom callback configuration

After confirming changes, the new callback entry is listed on the "Callbacks" section:

ASSOCIATED DEVICES

DEVICES BEING REGISTERED

STATISTICS

EVENT CONFIGURATION

CALLBACKS

BULK OPERATIONS

These callbacks transfer data received from the devices associated to this device type to your infrastructure. For more informations, please refer to the [Callback documentation](#)

DATA callbacks


Downlink	Enable	Channel	Subtype	Duplicate	Batch	Information
	<input checked="" type="checkbox"/>		UPLINK	<input type="checkbox"/>	<input type="checkbox"/>	[POST] http://hw.libelium.com/sf

Figure: Sigfox callback list

The callback entry may be edited anytime. For further information about callback function please check out this Sigfox documentation link: <https://backend.sigfox.com/apidocs/callback>.

Log And Debug Callback Requests To Libelium Cloud Bridge service

On the device list, click on the "ID" field of your device and a new screen with a right menu will be shown. On the "Messages" option you can see a list of messages sent to the callback and you can watch if they arrived properly (green circle with up arrow) or they had some kind of problem:

Device - Messages

From date

To date

RESET FILTER CSV

page 1

Time	Data / Decoding	Location	Link quality	Callbacks
2018-10-03 08:33:51	810c <input type="text"/> 47 ASCII: _L_AM45,G			
2018-10-03 08:33:44	8109 <input type="text"/> 1 ASCII: _4dJjLA			
2018-10-03 08:23:30	80f <input type="text"/> 47 ASCII: _L_JAM&a,G			
2018-10-03 08:23:23	80L <input type="text"/> 341 ASCII: _4dJjLA			
2018-10-03 08:13:09	7f0 <input type="text"/> c347 ASCII: _L_BMj,G			
2018-10-03 08:13:03	7f0 <input type="text"/> 341			

Figure: Sigfox callback list

If you click on the callback icon, you get full information about the callback request:

Device - Messages

From date

To date

RESET FILTER CSV

Callback - OK

[OK] - Base station - 1 second

200 - - #1

POST http://hw.libelium.com/sf HTTP/1.0

authorization : Bearer

content-length : 138

accept-encoding : gzip,deflate

accept-language : fr

host : hw.libelium.com

user-agent : SIGFOX

accept-charset : UTF-8;q=0.9,*;q=0.7

content-type : application/json

```
{
  "device_id" : " ",
  "data" : "810c ",
  "time" : "2018-10-03 08:33:51",
  "seqNumber" : "1",
  "duplicate" : "false"
}
```

Figure: Sigfox callback list

5.1.1.4. Loriot

Loriot is a backend for LoRaWAN devices. It gathers information from many base stations and works as a data hub. We will configure the Loriot backend to send data to the Libelium Cloud Bridge service.

A valid Loriot account is needed to configure properly the backend. New Loriot accounts can be obtained at <https://www.loriot.io/register.html>.

Register New Device

You should register your device or gateway in Loriot before starting. This process is different and depends on the device you want to register, you can do this step on the "Gateways" section on the bottom left part of the web.

Create Application For Device

In the dashboard you can create a new application for your device, this is needed to send data to Libelium Cloud. Follow the on-screen steps to create an application.

Once you created your application, click on it to get the details and extended actions.

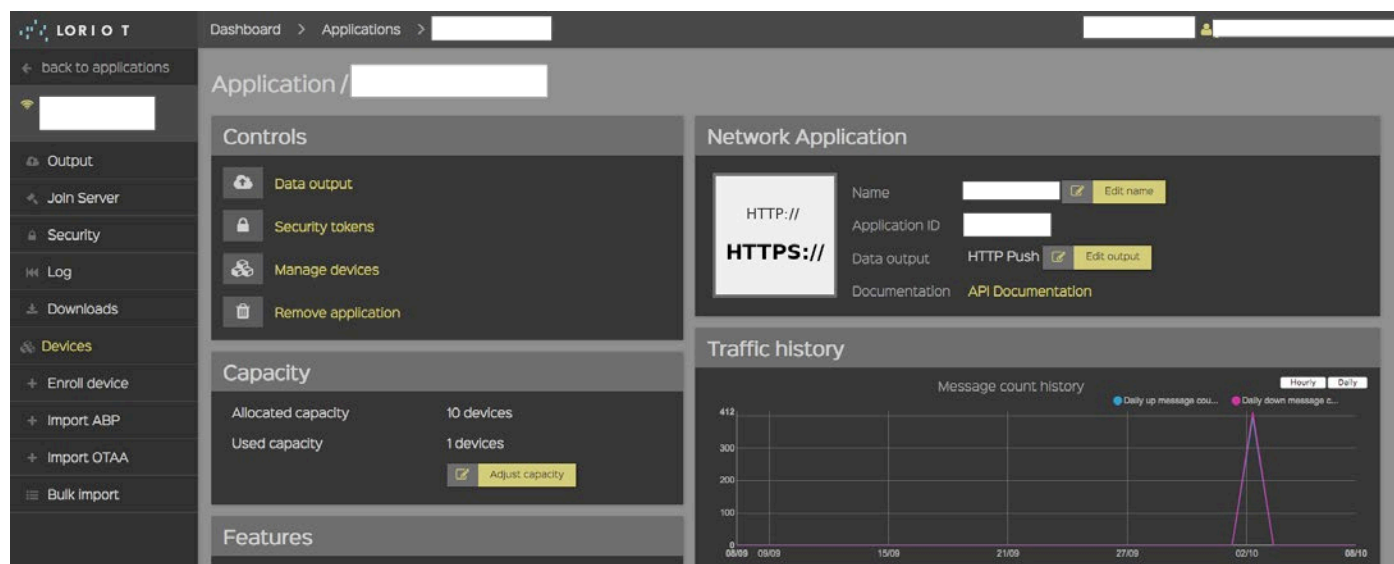


Figure: Loriot application

Redirect Data Output

There is a “Controls” section on the top left menu where you can configure the data output to redirect the incoming data to another service. Click on “Data output” to configure the data forwarding. Then you can choose among many services to redirect your data; for using the Libelium Cloud Bridge service, select “HTTP Push”:

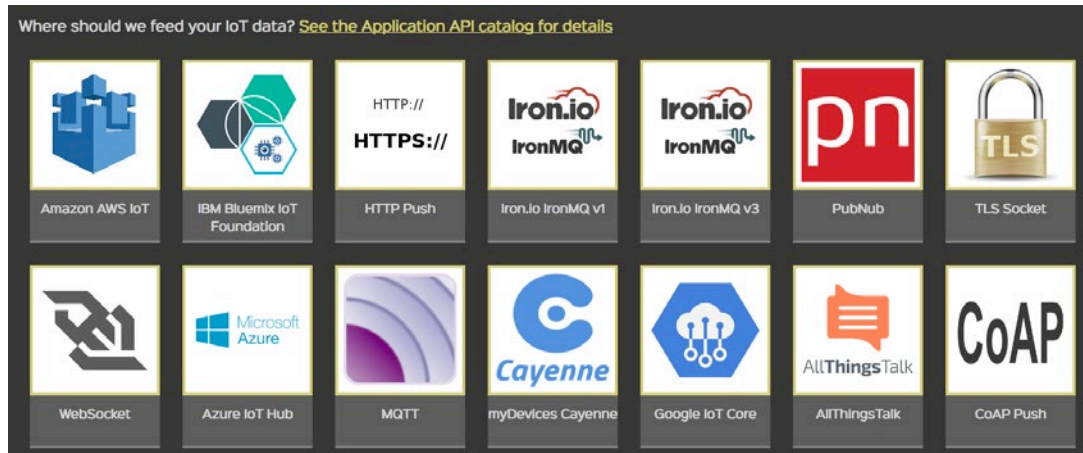


Figure: Loriot application

Select “HTTP Push” to have access to its documentation.

The configuration for this callback service is straightforward, you just type the Libelium URL and the value for the authentication field, in this case is “Bearer LIBELIUM_CLOUD_TOKEN”.

The Libelium Cloud Bridge service uses an API Key as the method to identify and authenticate to the user and the sensor node. Including the authentication API Key in the custom callback created is done adding to the headers option an authorization field with the value “Bearer LIBELIUM_CLOUD_TOKEN” (without quotes). The LIBELIUM_CLOUD_TOKEN can be copied from the Libelium Cloud Bridge service backend as explained in the section “API Keys” of this guide.

- **Target URL for POSTs:** <https://hw.libelium.com/hw/loriot>
- **Custom “Authorization” header value:** Bearer LIBELIUM_CLOUD_TOKEN

The HTTP Push configuration should be as follows:

Figure: Loriot application

Click on the button “Confirm change” to save your changes and the data will be forwarded to the Libelium Cloud Bridge service.

A JSON payload is not needed in this backend, the Loriot backend supplies a reliable JSON object that contains all information that Libelium Cloud Bridge service needs to manipulate the data.

5.1.1.5. Actility

The Actility backend is suitable for the user that has a good knowledge of IoT systems, because it is very flexible and full of useful features.

You will need an account to configure the Actility backend. If you do not have an account, get one on <https://www.actility.com/> or contact your Sales agent.

Device Configuration (Application Server)

The Actility service needs an Application Server and an Application Server Routing Profile (AS routing profile) to create a communication between Actility and the callback server.

First of all, create a new Application Server using the left side menu:



Figure: Actility application server

The first section allows you to create a new Application Server. The second section shows a list of all the Application Servers created. Click the “Add” button in the first section to create a new Application Server.

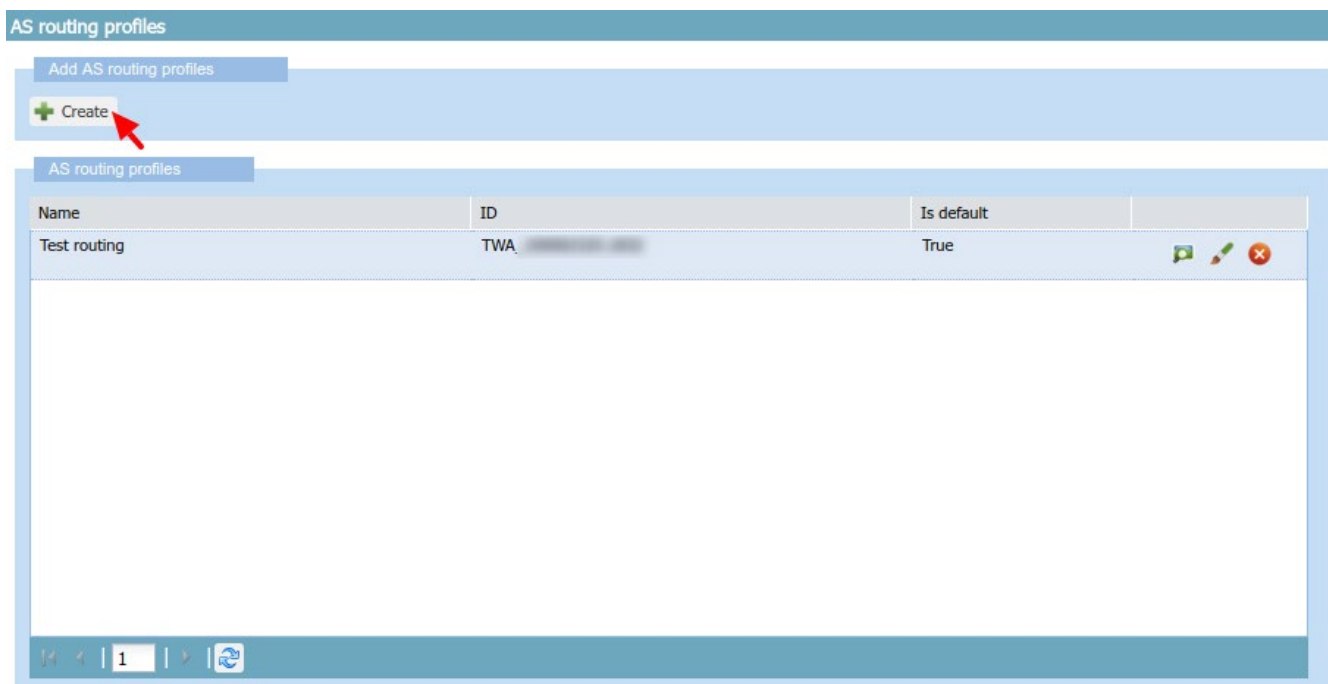


Figure: Add new application server

You must type a name and select the option “HTTP Application Server”. Then click the “Create” button to continue:

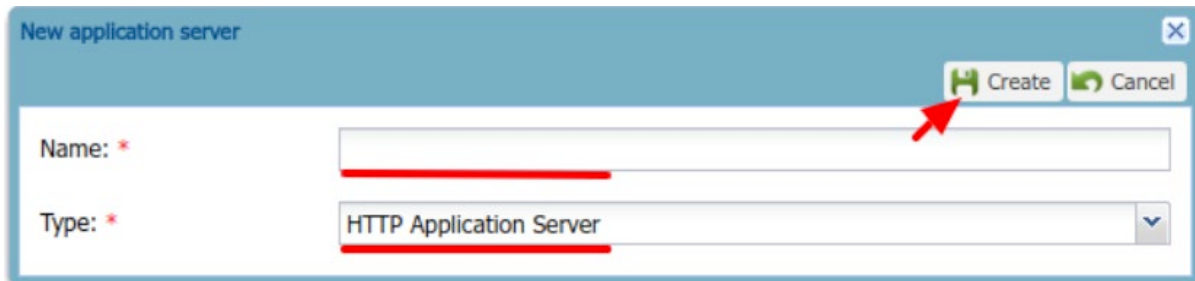


Figure: Creates new application server

The new window shows the name from the previous pop-up and many other new fields. Set the value “JSON” for the “Content Type” field.

The Libelium Cloud Bridge service adds another security layer to Actility cloud. It is mandatory to fulfil the “Uplink/downlink security” section for data integrity. Click on the “Activate” button and a new pop-up will appear:

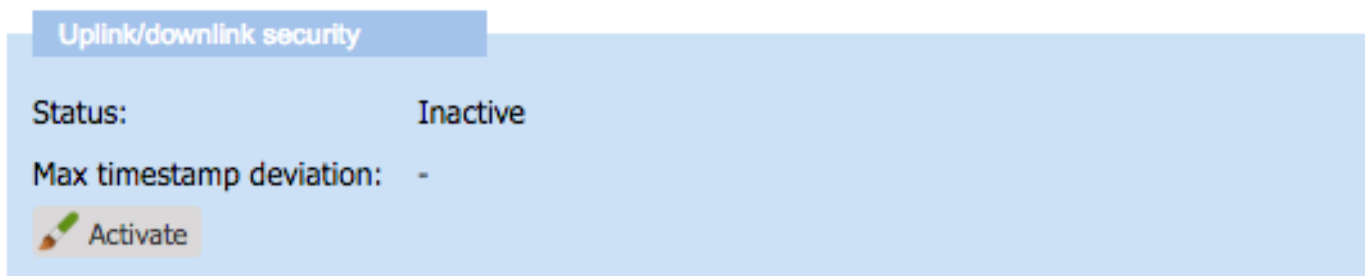


Figure: Activate new application server

In the new pop-up there are three fields: AS ID, LRC-AS Key and Max timestamp deviation, let’s explain each one:

- **AS ID:** It is a field to identify the user that is sending data, this information is supplied in the Actility Screen on the Libelium Cloud Bridge service
- **LRC-AS key:** A passphrase supplied by Actility Configuration on Libelium Services Cloud Manager
- **Max timestamp deviation:** Leave it empty

AS ID and **LRC-AS key** are mandatory fields, these values can be copied from the Libelium Cloud Bridge service backend. If the fields are not properly filled the data will not arrive to the Libelium Cloud Bridge service.

Now it is time to add a new route to send the data:

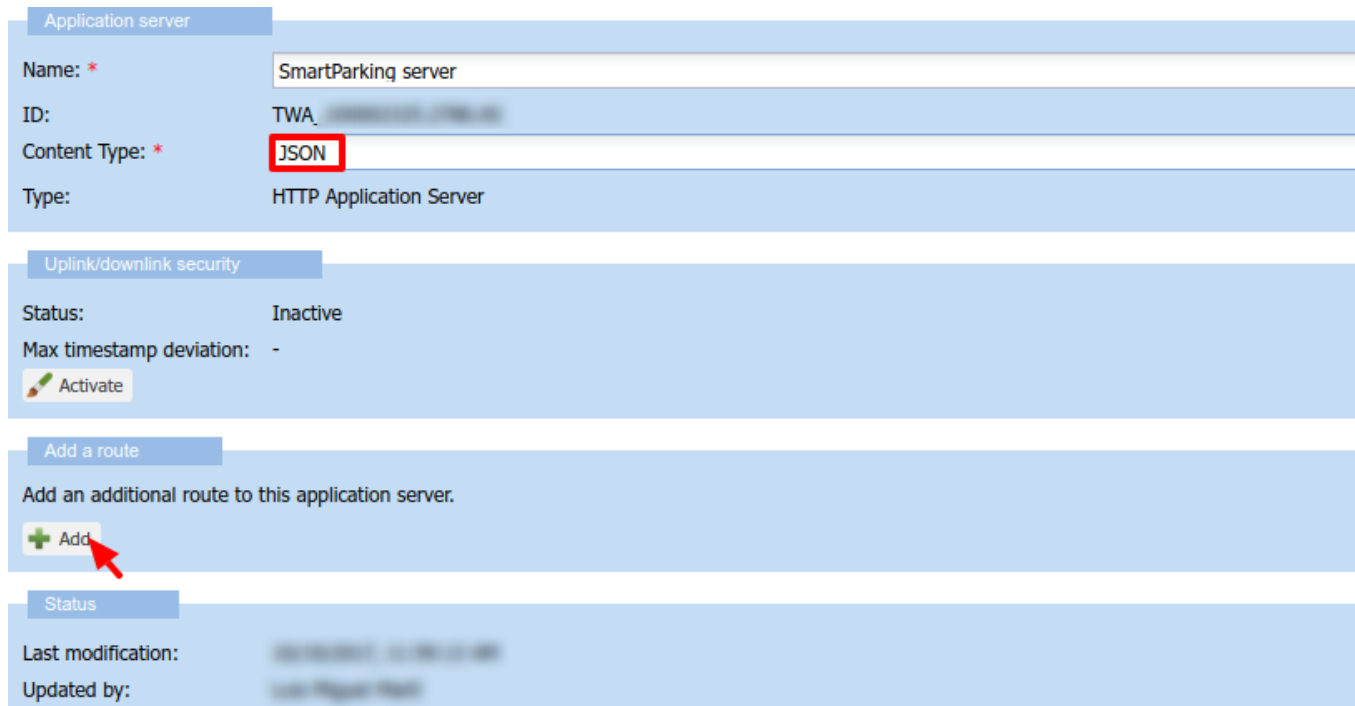


Figure: Add a new route

Click on the button “Add” from “Add a route” section, and a new window will be displayed. Then set the values for these fields:

- **Source ports:** *
- **Routing strategy:** Sequential

On the above destination box, click the “Add” button to type a new URL to callback the data:

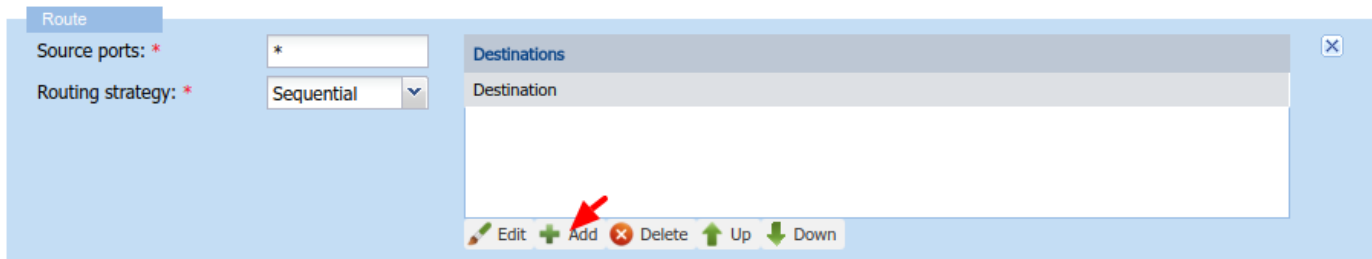


Figure: Create a new route

A pop-up appears, type the Libelium cloud URL on the text box: <https://hw.libelium.com/hw/activity>

Click the “Add” button to save the changes.

The “Route” section should be as follows:



Figure: New route

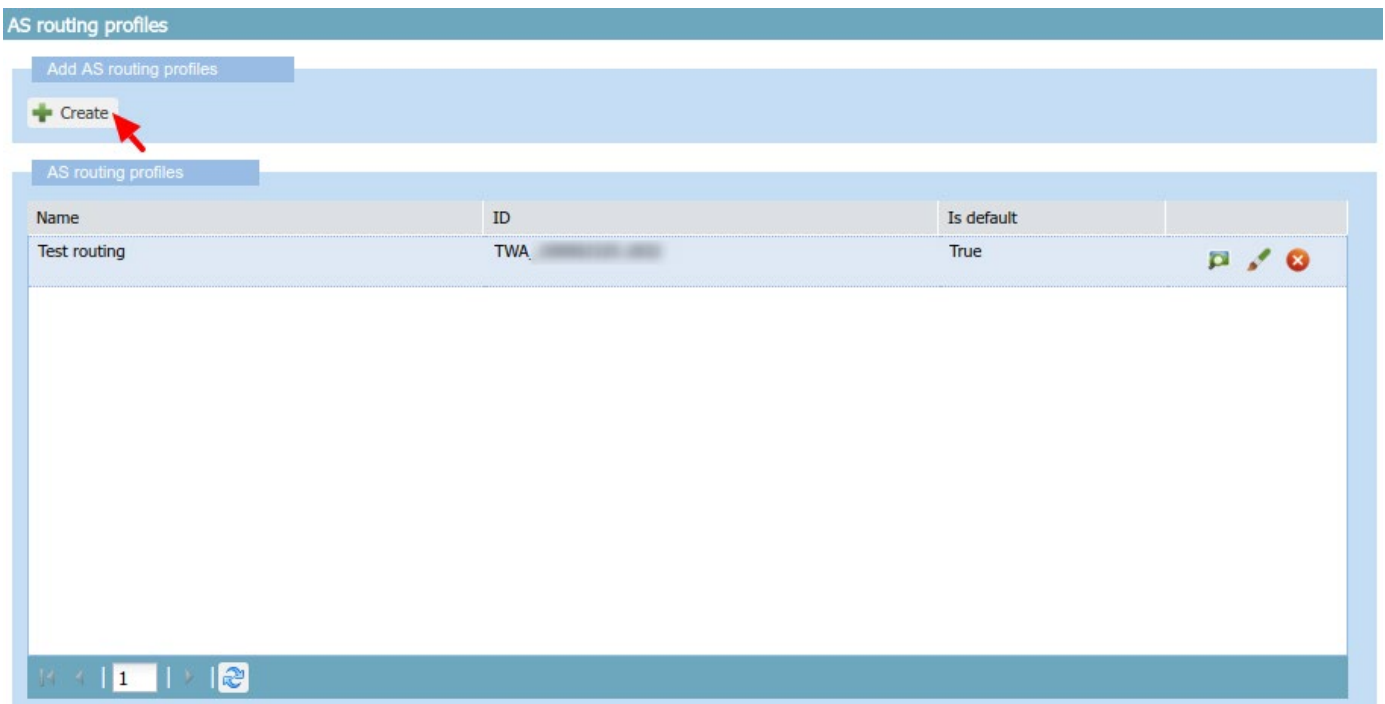
Device Configuration (AS Routing Profile)

In the main menu on the left side click on “AS routing profiles”.



Figure: Activity routing profiles

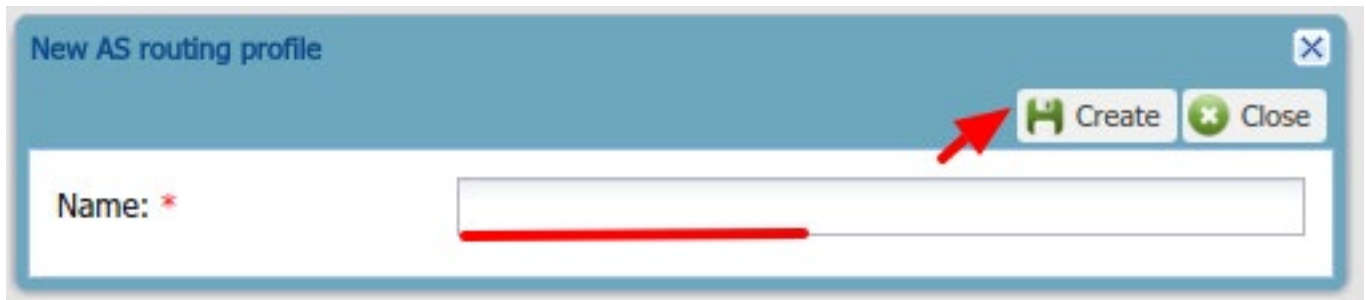
All existing AS routing profiles are listed in the “AS routing profiles” section. Click the “Create” button shown in the first section to create a new AS routing profile. A new AS routing profile gives the capability to add new AS routing profiles.



Name	ID	Is default
Test routing	TWA	True

Figure: Add routing profiles

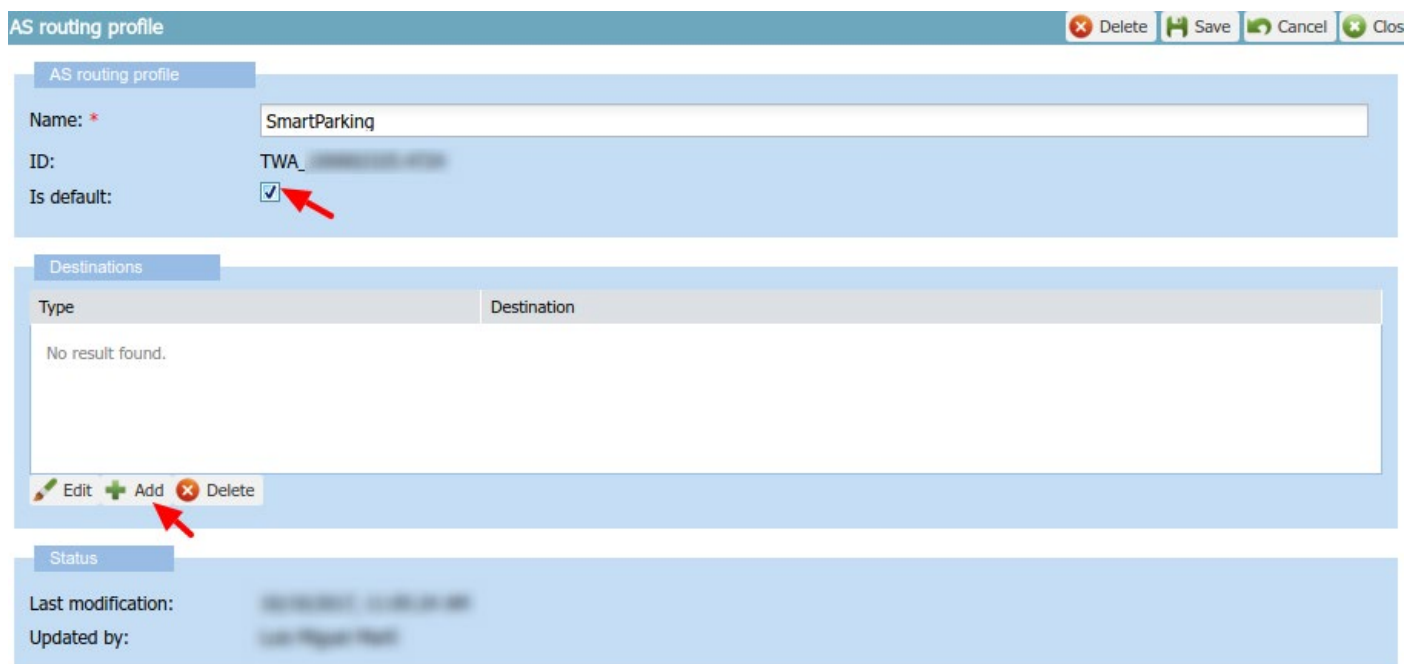
You must supply a name in the new window displayed. After that, click on “Create” to continue the process.



The dialog box titled "New AS routing profile" has a close button (X) in the top right corner. Below the title bar, there are two buttons: "Create" (with a green plus icon) and "Close" (with a green X icon). A red arrow points to the "Create" button. Below these buttons is a text input field labeled "Name: *". A red underline is visible under the input field.

Figure: Create routing profiles

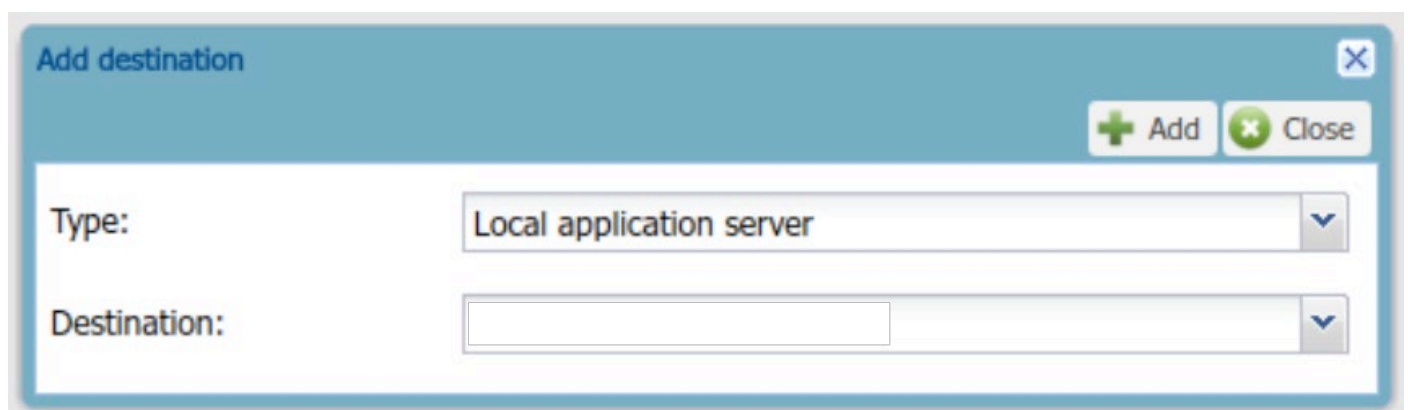
Uncheck the option "Is default" and click on "Add" on the route section:



The "AS routing profile" form has a top bar with "Delete", "Save", "Cancel", and "Close" buttons. The main form is divided into three sections: "AS routing profile", "Destinations", and "Status".
 In the "AS routing profile" section, there are three fields: "Name: *" with the value "SmartParking", "ID:" with the value "TWA_...", and "Is default:" with a checked checkbox. A red arrow points to the "Is default:" checkbox.
 In the "Destinations" section, there is a table with two columns: "Type" and "Destination". The table is empty, and the text "No result found." is displayed below it. Below the table are three buttons: "Edit" (with a pencil icon), "Add" (with a green plus icon), and "Delete" (with a red X icon). A red arrow points to the "Add" button.
 In the "Status" section, there are two fields: "Last modification:" and "Updated by:", both with blurred values.

Figure: "Is default" option

A new pop-up will appear with two fields. "Type" is the kind of application for our destination, in this case "Local application server". Then, in the "Destination" field, all application servers created will be loaded in the drop-down. The application server previously created must be selected.



The "Add destination" dialog box has a close button (X) in the top right corner. Below the title bar, there are two buttons: "Add" (with a green plus icon) and "Close" (with a green X icon). Below these buttons are two fields: "Type:" with a dropdown menu showing "Local application server", and "Destination:" with an empty dropdown menu. Both dropdown menus have a downward arrow icon on the right side.

Figure: Add destination

Navigate back to the AS routing profile form and click "Save" to create it.

Device Configuration (Create New Device)

At this point the user can add as many devices as they need by clicking on the “Devices” option on the left menu. Then, in the new form loaded on the right frame, create a new device by hitting on the “Create” button.

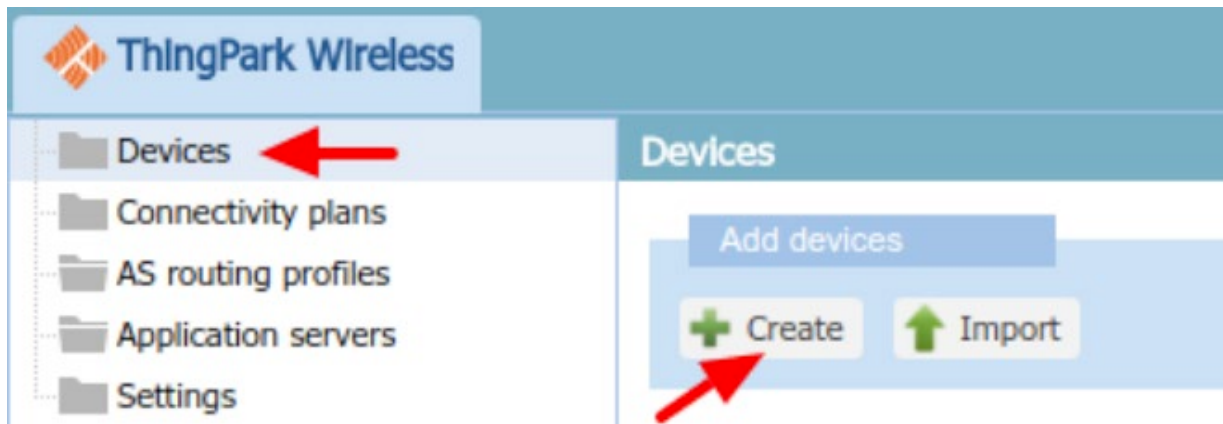
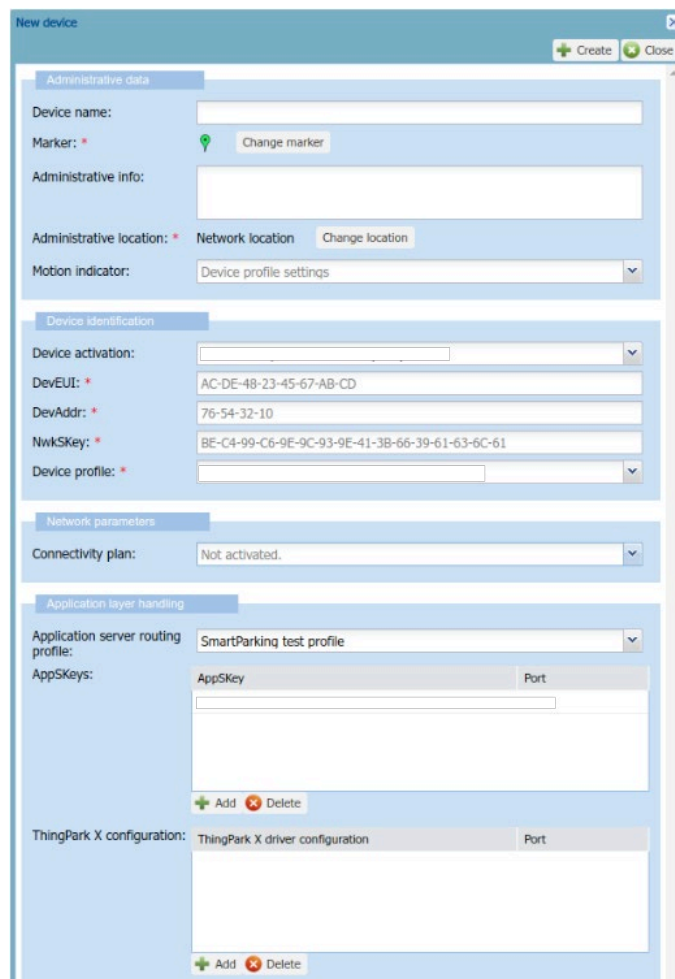


Figure: Create device

A new form will be displayed in a pop-up to fulfil the device information:


 The screenshot shows a 'New device' form with the following sections:

- Administrative data:** Includes fields for 'Device name', 'Marker' (with a location pin icon and 'Change marker' button), 'Administrative info', 'Administrative location' (with a dropdown menu set to 'Network location' and a 'Change location' button), and 'Motion indicator' (with a dropdown menu set to 'Device profile settings').
- Device identification:** Includes fields for 'Device activation' (dropdown), 'DevEUI' (text field with value 'AC-DE-48-23-45-67-AB-CD'), 'DevAddr' (text field with value '76-54-32-10'), 'NwkSKey' (text field with value 'BE-C4-99-C6-9E-9C-93-9E-41-3B-66-39-61-63-6C-61'), and 'Device profile' (dropdown).
- Network parameters:** Includes a 'Connectivity plan' dropdown menu set to 'Not activated'.
- Application layer handling:** Includes an 'Application server routing profile' dropdown menu set to 'SmartParking test profile', and a table for 'AppSKeys' with columns 'AppSKey' and 'Port'. Below the table are 'Add' and 'Delete' buttons.
- ThingPark X configuration:** Includes a table for 'ThingPark X configuration' with columns 'ThingPark X driver configuration' and 'Port'. Below the table are 'Add' and 'Delete' buttons.

 The form has a title bar with 'New device' and buttons for '+ Create' and 'Close'.

Figure: Create device form

After this configuration you are ready to send data to Libelium Cloud Bridge service. It is not necessary to create a JSON payload because Actility creates a JSON structure that the Libelium Cloud Bridge service can process.

5.1.1.6. LoRaWAN MultiTech Base Station

Libelium Application Installation

The user can install the application from the www.libelium.com site. It does not require a big knowledge, just create a new folder into the MultiTech base station and download the file into the MultiTech base station, unzip the downloaded file. It is important to unzip all the files in the same folder.

Configuration

The Libelium application needs some parameter configuration, The user should know that some parameters are mandatory to execute the application properly. The configuration parameters are:

- **log:** This parameter can have the value 0 or 1. This parameter enables (value 1) or disables (value 0) the log file. If it is disabled the user just has an on-screen log. Default value is 0. This is a mandatory parameter to configure Libelium application.
- **host:** MultiTech Base Station host. As the Libelium application is installed onto a base station, the value will be localhost, but the user can also pull the data from other base station. Default value is localhost. This is a mandatory parameter to configure Libelium Application.
- **username:** MultiTech Base Station username to login into the web interface. Default value is admin. This is a mandatory parameter to configure Libelium Application.
- **password:** MultiTech Base Station password to login into the web interface. Default value is admin. This is a mandatory parameter to configure Libelium Application.
- **appeui:** The user can register a new appeui using the MultiTech Base Station web interface to get only the frames from the given appeui. Default value is empty. This is an optional parameter to configure Libelium Application.
- **libelium_cloud_token:** This is a token given by the Libelium Cloud Bridge service., The user must create an account in the Libelium Cloud Bridge service to get this token. There is no default value as the user must get the token and set it in the configuration file. This is a mandatory parameter to configure Libelium Application.

All these parameters should be in a file called config.ini. There is a file called config.in.template where the user can rename to config.ini and fulfil the configuration parameters.

This is a sample of configuration for a config.ini file:

```
{
[credentials]
log="0"
host="localhost"
username="admin"
password="admin"
appeui=""
libelium_cloud_token="YOUR_LIBELIUM_CLOUD_TOKEN"
}
```

Cron Configuration

The Libelium application can be executed periodically as a cron job. To install a new cron job with the Libelium application you can use the script `install_cloud_meshlium_multitech.sh`, which installs a new line on your cron application and run the main Libelium application every 5 minutes.

Grant execution permission and execute the script with the following commands:

```
{  
  chmod +x  install_cloud_meshlium_multitech.sh  
  ./install_cloud_meshlium_multitech.sh  
}
```

Do not forget to restart your cron application to apply the changes with the following command:

```
{  
  /etc/init.d/crond restart  
}
```

Execute Libelium Application For MultiTech Base Stations

In this chapter the user will learn how to run the Libelium application for MultiTech Base Stations and how the application works.

Execute the application. Once you have configured the application you must give execution permission to the main application:

```
{  
  chmod +x  cloud_meshlium_multitech.sh  
}
```

Before running the application, it is very important to set the MultiTech Base Station as **NETWORK SERVER**. You can change this setting on your MultiTech Base Station web interface on "LoRaWAN" → "Network Settings".

Now the user is ready to execute the application:

```
{  
  ./cloud_meshlium_multitech.sh  
}
```

The application shows an on-screen log with all operations. The application performs these steps in the MultiTech Base Station:

- The application logs in the base station
- The application gets a login token given by the base station
- The application queries the frames provided by the base stations
- The application processes all the frames to upload them to the Libelium Cloud Bridge service and split the frame to reduce request time between the base station and the Libelium Cloud Bridge service
- The application starts uploading frames to Libelium Cloud Bridge service
- The application logs out from the base station and the application ends execution

This is an on-screen sample log that shows all the processes from the beginning to the end of the execution where the data is uploaded to Libelium Cloud Bridge service:

```
{
[2018-10-01 07:36:08] - Starting Libelium Cloud application for Multitech Stations...
[2018-10-01 07:36:08] - -----
[2018-10-01 07:36:08] - ** Remember to configure your Multitech Base Station as NETWORK SER-
VER in your Network Settings **
[2018-10-01 07:36:08] - Configuration file found...
[2018-10-01 07:36:08] - Extracting values...
[2018-10-01 07:36:09] - Configuration elements: 6
[2018-10-01 07:36:09] - File log: DISABLED
[2018-10-01 07:36:09] - Host: localhost
[2018-10-01 07:36:09] - Username: admin
[2018-10-01 07:36:09] - Password: admin
[2018-10-01 07:36:09] - Configuration values OK...
[2018-10-01 07:36:09] - File token found!
[2018-10-01 07:36:10] - Token found: 40D85DB1868FB123DC53AE3AC3EFD766
[2018-10-01 07:36:15] - User logged out
[2018-10-01 07:36:15] - Script continues...
[2018-10-01 07:36:19] - {
  "code" : 200,
  "result" : {
    "address" : "127.0.0.1",
    "permission" : "admin",
    "port" : "60791",
    "timestamp" : "7:36:19:784",
    "token" : "8B4A6B9C7FA852F1234E09633AEC771",
    "user" : "admin"
  },
  "status" : "success"
}
[2018-10-01 07:36:21] - User logged in
[2018-10-01 07:36:21] - Getting login token...
[2018-10-01 07:36:25] - Token is: 8B4A6B9C7FA852F1234E09633AEC771
[2018-10-01 07:36:25] - Getting last frame values...
[2018-10-01 07:36:33] - Found last frames...
[2018-10-01 07:36:33] - Processing last frames...
[2018-10-01 07:36:42] - Frame result packets up not empty
[2018-10-01 07:36:48] - Libelium Cloud application will upload 100 frames to Libelium
Cloud...
[2018-10-01 07:36:54] - Creating request to upload frames to Libelium Cloud...
[2018-10-01 07:36:54] - Frames processed, now sending to cloud, please wait...
[2018-10-01 07:39:10] - Elapsed time on sending to cloud: 136 seconds
[2018-10-01 07:39:10] - SUCCESS - Frames uploaded to Libelium cloud!!!
[2018-10-01 07:39:10] - Logging out from system...
[2018-10-01 07:39:14] - {
  "code" : 200,
  "result" : {
    "address" : "127.0.0.1",
    "permission" : "admin",
    "port" : "60791",
    "timestamp" : "7:36:19:784",
    "token" : "8B4A6B9C7FA852F1234E09633AEC771",
    "user" : "admin"
  },
  "status" : "success"
}
}
```

You can run the Libelium application every time you need to upload new frames to the Libelium Cloud Bridge service.

All this information is summarized on the README file.

Troubleshooting

First of all, to avoid network connection or connectivity problems between the MultiTech Base Station and the Libelium Cloud Bridge service, make sure that your internet connection is working.

If you experience some low performance on your Base Station using the Libelium application, make sure that it does not have a stressful application running that drains the CPU usage.

If the MultiTech Base Station is not able to get new data frames, make sure that your devices are sending new data and the configuration of the LoRaWAN Network Settings on the MultiTech Base Station is set as NETWORK SERVER.

If you obtain a misconfiguration error, please review your configuration setting on config.ini and fill the missing parameters.

If you obtain a bad response from the Libelium Cloud Bridge service once you have upload your new frames, please review your configuration, specially your Libelium Cloud Token.

If you experience a lack of connection between the MultiTech Base Station and the Libelium Cloud Bridge service and your network settings are correct, please write a post at the Libelium Forum and we will help you to solve your issue: <https://www.libelium.com/forum>.

5.1.2. Buffering Data

Data is not stored in the Libelium Cloud Bridge service, in the process of syncing data with the cloud services only a limited buffer is needed.

The synchronization process is also optimized to group values before sending them to the cloud services and to run parallelized when needed. The buffering process is design to work properly with a huge amount of sensor nodes sending data.

Before programming any single sensor node to send data to the Libelium Cloud Bridge service, it is recommended to configure at least one cloud service to synchronize data to. If sensor nodes are sending data to the Libelium Cloud Bridge service but there is not any cloud service properly configured, the buffer will get full and the sensor nodes will start receiving errors when trying to send data.

It is possible to check the current percentage of the buffer usage in the Libelium Cloud Service license panel.

5.1.3. Syncing With The Cloud Services

The Libelium Cloud Bridge service implements connectors for the following cloud services: Alibaba Cloud, AWS IoT, Cumulocity, IBM Bluemix, Microsoft Azure Event Hubs, Microsoft Azure IoT Hubs, MQTT, SAP Hana and Telefónica IoT Cloud.

Several instances of any of them may be configured. Data received in the Libelium Cloud Bridge service will be buffered waiting to be synchronized with all the active connectors instances. Only values that can be synchronized with all the active connector instances will be cleared from the buffer. A misconfiguration problem in any single connector instance may lead to fill the buffer.

Data is sent to the cloud services implementing the HTTPS REST API calls and MQTT protocols.

The Libelium Cloud Bridge service measures the synchronized values with the cloud services. The current percentage of the synchronization limit of the active license may be checked in the Libelium Cloud Service license panel.

5.2. Encryption In Depth

There are several encryption levels used when communicating the sensor nodes with the Libelium Cloud Bridge service using TCP / IP protocol:

- Application level encryption: **AES256** (symmetric key known by the sensor node and the final cloud service)
- Application level encryption: **AES128** (secret symmetric key stored in the sensor node and the Libelium Cloud Bridge service. This encryption key in the sensor node cannot be read "under any circumstances".
- **HTTPS** encryption: Some sensor node communication modules (WiFi or 4G) implement HTTPS with the Libelium Cloud Bridge service

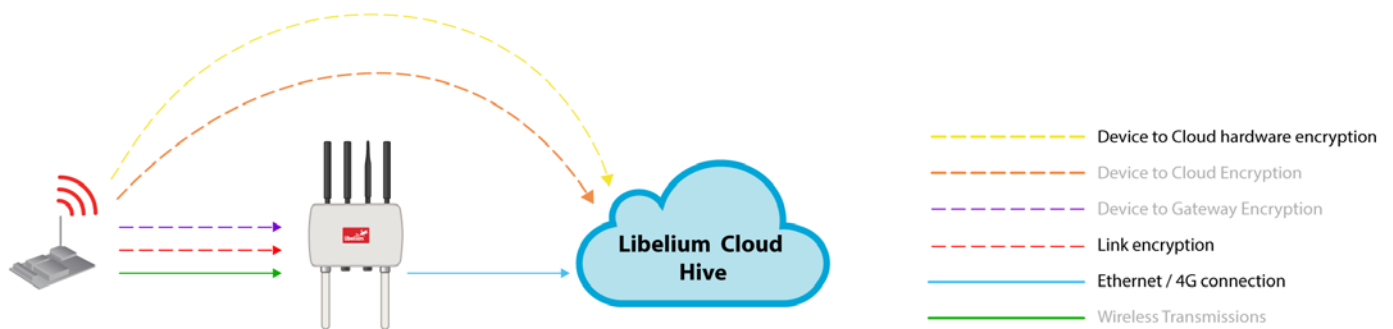


Figure: Encryption diagram

6. Gateway Configuration

This section lists all the Meshlium gateways registered in your user account. The main view is a panel showing a grid with basic information to identify the Meshlium gateways.

≡ More actions

Configure gateways

Smart City :: South Carriefurt

 ID: 18299115224520

Meshlium #18299115224520

MANAGE

Smart City :: North Kevinburgh

 ID: 1784829569418383

Meshlium #1784829569418383

MANAGE


Smart City :: Morissetteside

 ID: 5624509990845677

Meshlium #5624509990845677

MANAGE

Smart City :: Ludwigbury

 ID: 5585420945401236

Meshlium #5585420945401236

MANAGE

Smart City :: Danielshire

 ID: 1734417157540894

Meshlium #1734417157540894

MANAGE

Figure: Gateway configuration grid

Click the “Setup” button to manage and configure the Meshlium gateway. It is possible to remotely access to the Manager System panel of a Meshlium gateway to configure all the parameters just as if we were locally accessing its Manager System.

Smart City :: South Carriefurt

 ID: 18299115224520

Meshlium #18299115224520

MANAGE

Figure: Gateway configuration setup

6.1. Remote Gateway Management

To remotely manage the Meshlium gateway, a reverse communication tunnel will be established between the Meshlium gateway and the Libelium Cloud Bridge service.

The field “Status” shows the current state of the tunnel; possible values are:

- Offline: there is no tunnel currently active
- Online: the tunnel is opened and the remote Manager System will be accessible in the window

A “Connect” button is also available to open the tunnel.

◀ CONFIGURE GATEWAYS > CONNECT TO GATEWAY

☰ More actions

Smart City :: East Shanon

Status: **Offline**



ID: 18299115224520

Meshlium #18299115224520

Connect

Figure: Remote connection

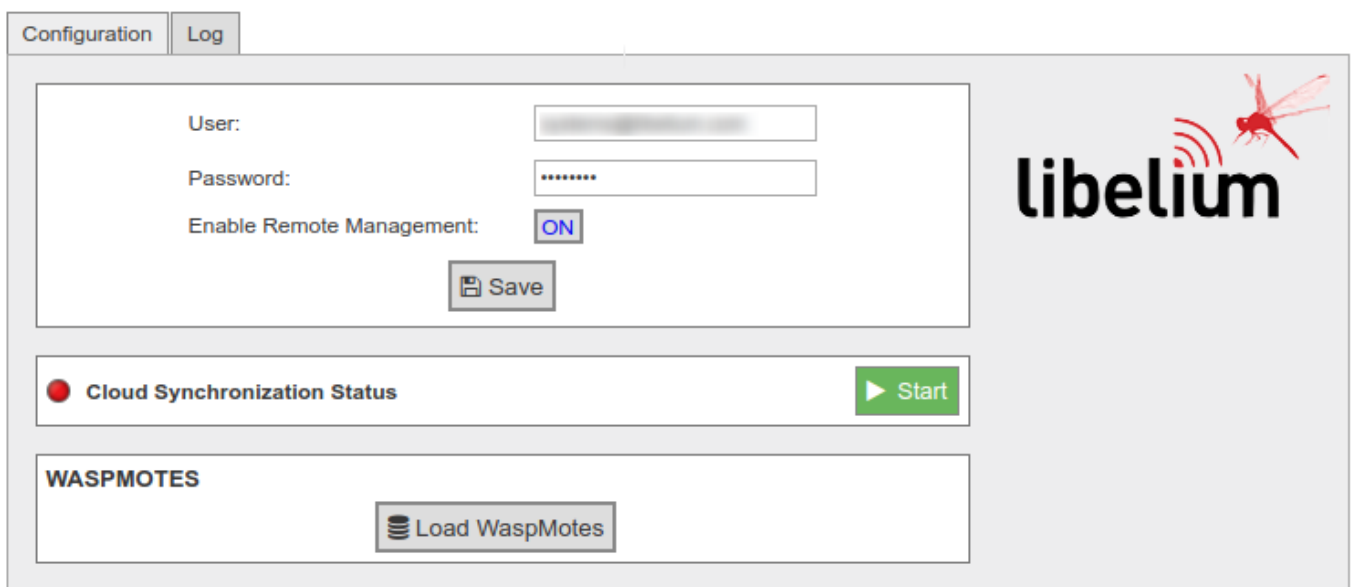
6.1.1. Enable Remote Management

The remote gateway management feature must be enabled in the Meshlium gateway when configuring the Libelium Cloud Bridge connector:

“Cloud Connector” → “Premium Cloud Partner” → “Libelium Cloud Bridge” connector

Libelium Cloud user credentials should be properly configured and the “Enable Remote Management” option must be set to “On” as indicated by the image. This option enables the Manager System panel to be remotely displayed from the Libelium Cloud Bridge service.

IoT Premium > Libelium Cloud

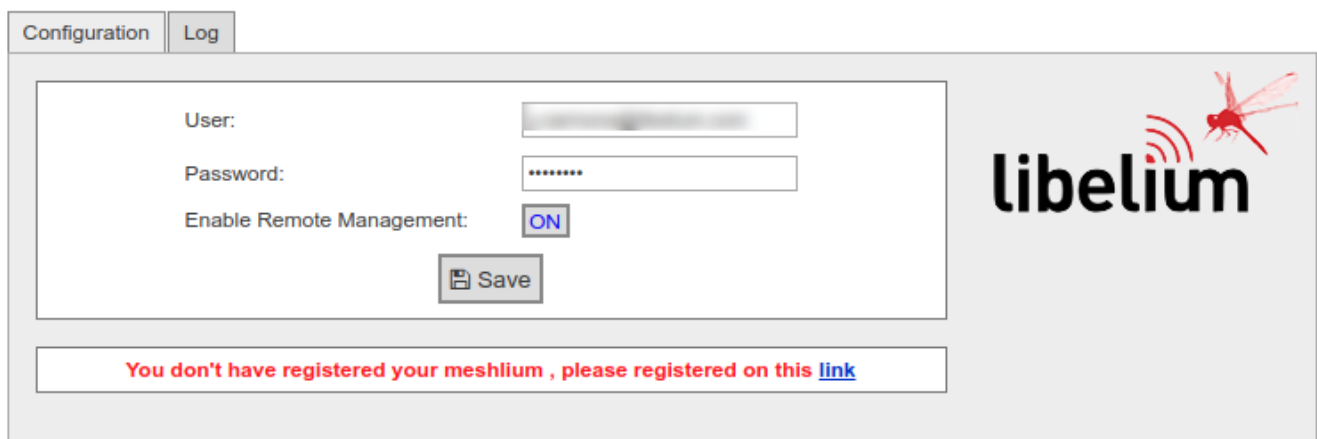


The screenshot shows the 'Configuration' tab of the Libelium Cloud Bridge service connector. It includes fields for 'User' and 'Password', and a toggle for 'Enable Remote Management' which is currently set to 'ON'. A 'Save' button is located below these fields. To the right, there is a 'Cloud Synchronization Status' section with a 'Start' button, and a 'WASPMOTES' section with a 'Load WaspMotes' button. The Libelium logo is visible on the right side of the interface.

Figure: Libelium Cloud Bridge service connector, Enable Remote Management

The option “Enable Remote Management” will only be available if the Meshlium gateway has previously been registered. Meshlium gateways that have not been registered in the Services Cloud Manager will show the message “You don’t have registered your Meshlium, please register on this link” instead of the check option.

IoT Premium > Libelium Cloud



The screenshot shows the same configuration interface as the previous one, but with an error message displayed at the bottom: “You don’t have registered your meshlium , please registered on this [link](#)”. The 'Enable Remote Management' option is still set to 'ON', but the 'Save' button is disabled.

Figure: Meshlium not registered error message

6.1.2. Connect

Click the “Connect” button to start the process of establishing the reverse communication tunnel. The process may take up to three minutes to complete. Meshlium gateways are pooling every 60 seconds to the Libelium Cloud Bridge service to check if the tunnel should be opened.

◀ CONFIGURE GATEWAYS > CONNECT TO GATEWAY

☰ More actions

Smart City :: East Shanon

Status: **Offline**



ID: 18299115224520

Meshlium #18299115224520

Cancel

This process can take up to 3 minutes.

🔄 Connecting ...

Figure: Connecting info message

Unpredictable network errors may take it longer to establish the communication. An error message will be shown when the maximum timeout is reached and it has not been possible to open the tunnel successfully. The process may be canceled anytime with the “Cancel” button.



Figure: Cancel connection process

When the process is finished correctly, the Manager System of the remote Meshlium gateway will be displayed in the box, and the “Status” field will turn to “Online”. User and password are required as if you were accessing locally. All the features of the Meshlium gateway can be configured.

◀ CONFIGURE GATEWAYS > CONNECT TO GATEWAY

☰ More actions

Smart City :: East Shanon

Status: **Online**

ID: 18299115224520

Meshlium #18299115224520

Disconnect



Figure: Access remote Manager System

Click the “Disconnect Button” to close the connection.



Figure: Disconnect button

If the process is finished correctly the “Status” field will turn to “Offline”, and the box showing the remote Manager Systems will be hidden.

6.2. Connection Timeout

Properly established remote management connections will timeout after 30 minutes and a warning message will be displayed on screen.

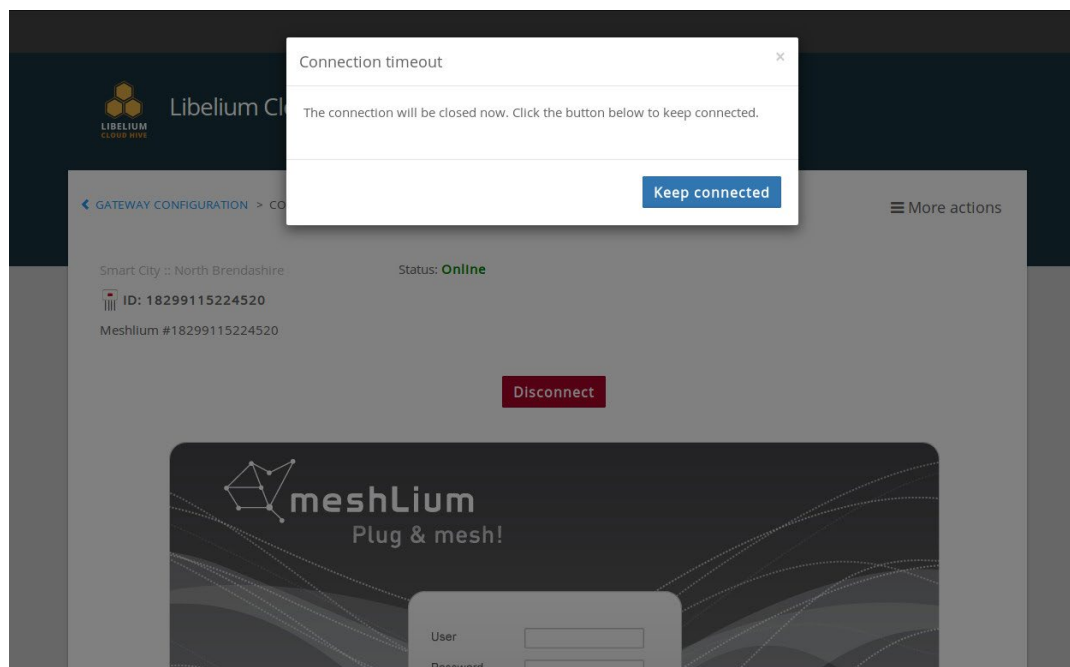


Figure: Connection timeout

The warning message is displayed for 30 seconds, in this time the connection may be kept alive by clicking the button "Keep connected". Keeping the connection alive means another 30 minutes accessing the Manager System to remotely manage your gateway.

If the button is not clicked before those 30 seconds, the warning message will be hidden and the connection with the Meshlium gateway will be closed. After the connection is closed, a new "Connect" process may be established again.

Closing the browser window does not close the connection. Navigating to the "Connect to Gateway" section of the Gateway Configuration feature using the "Setup" button will show directly the Manager System and the Online status for those connections previously established that have not been closed yet and remain active.

7. Programming Sensor Nodes To Use The Libelium Cloud Bridge Service

7.1. Using The Programming Cloud Service

Sensor nodes have to be programmed to send data to the Libelium Cloud Bridge service. There are two options for that: using the Programming Cloud Service or manually writing code on the Wasmote IDE.


The Programming Cloud Service has the option “Send to Libelium Cloud Bridge” in the “Protocol and destination” section to generate valid programs that will connect directly to the Libelium Cloud Bridge service. All the security and encryption layers available in the Libelium security scheme will be implemented in those programs.

The information needed in the Programming Cloud Service to generate the specific binary program of each sensor node is shared from the Services Cloud Manager and the Libelium Cloud Bridge service. All the services available in the Libelium Cloud have access to the user information (active license type) and to the sensor nodes properties (unique device ID and MAC address).

The Programming Cloud Services has the section “Communication module” to choose what will be the radio used by the sensor node to send data. The available options are:

- XBee-PRO 802.15.4
- XBee-PRO 900HP
- XBee-PRO 868LP
- XBee ZigBee 3
- 4G
- WiFi
- Sigfox
- LoRaWAN

Depending on the communication module used in the sensor node, there will be different possibilities to send data to the Libelium Cloud Bridge service. In the section “Protocol and destination” the available options will change according to the selection done in the “Communication module” section.


 Select communication module *

WiFi

ESSID


libelium_AP

Security

Select

IP method

Select


 Select protocol and destination *

Send to Libelium Cloud Hive

Use Payload Encryption (AES 256)

☐

Figure: Programming Cloud Service , send to the Libelium Cloud Bridge service

In all cases, the recommended option is “Send to Libelium Cloud Bridge”. In some cases data will need to go through the Meshlium gateway and in other cases the data will be sent directly to the Libelium Cloud Bridge service.

Nodes using LPWAN communication modules will always send the data through the LPWAN backend servers, in the section “Receiving Data” of this guide it is explained how to configure the callbacks properly to redirect the data to the Libelium Cloud Bridge service.

In the next table we show the possible choices:

Communication module	Protocol and destination	Plain frame	AES-128	AES-256	Meshlium connector	Libelium Cloud Bridge service
XBee-PRO 802.15.4	Send to Libelium Cloud Bridge	Yes	frame	frame	Yes	Yes
	Send to Meshlium Gateway	Yes	Yes	Yes/No	No ¹	No ¹
XBee-PRO 900HP	Send to Libelium Cloud Bridge	Yes	No	Yes/No	Yes	Yes
	Send to Meshlium Gateway	Yes	Yes	Yes/No	No ¹	No ¹
Xbee 868LP	Send to Libelium Cloud Bridge	Yes	No	Yes/No	Yes	Yes
	Send to Meshlium Gateway	Yes	Yes	Yes/No	No ¹	No ¹
XBee ZigBee 3	Send to Libelium Cloud Bridge or Meshlium not available	Yes	No	No	No	No
	Send to generic ZigBee receiver available					
4G	Send to Libelium Cloud Bridge	Yes	No	Yes/No	No	Yes
	Send to Meshlium Gateway	Yes	Yes	Yes/No	No ¹	No ¹
	HTTPS	Yes	No	Yes/No	No ²	No ²
	HTTP	Yes	No	Yes/No	No ³	No ³
	TCP	Yes	No	Yes/No	No	No
	SMS	Yes	No	Yes/No	No	No
WiFi	Send to Libelium Cloud Bridge	Yes	No	Yes/No	No	Yes
	Send to Meshlium Gateway	Yes	Yes	Yes/No	No ¹	No ¹
	HTTPS	Yes	No	Yes/No	No ²	No ²
	HTTP	Yes	No	Yes/No	No ³	No ³
	TCP	Yes	No	Yes/No	No	No
Sigfox	Sigfox backend	Tiny	No	Yes/No	No	Yes
LoRaWAN	Loriot backend	Tiny	No	No	No	Yes
	Actility backend	Tiny	No	No	No	Yes
	MultiTech base station	Tiny	No	No	No	Yes

Figure: Programming Cloud Service “Send to Libelium Cloud Bridge”

¹ If the destination Meshlium gateway has the Libelium Cloud Bridge service connector properly configured, the data will be redirected to the Libelium Cloud Bridge service.

² It may be possible to configure the HTTPS parameters with the values of an active and running Meshlium gateway. If the destination Meshlium gateway has the Libelium Cloud Bridge service connector properly configured the

data will be redirected to the Libelium Cloud Bridge service, this option is discouraged in favour of using the “Send to Meshlium Gateway”. It may be possible to configure HTTPS parameters with the values of the Libelium Cloud Bridge service, this option is discouraged in favour of using the “Send to Libelium Cloud Bridge”.

³ It may be possible to configure the HTTP parameters with the values of an active and running Meshlium gateway having the HTTP protocol manually activated. If the destination Meshlium gateway has the Libelium Cloud Bridge service connector properly configured the data will be redirected to the Libelium Cloud Bridge service, this option is totally discouraged because all the security layers are disabled with this configuration.

7.2. Using Wasmote API (HTTPS Example)

In the Wasmote API documentation there is a sketch available (the HTTPS example) that may be used for the user as the basis for coding on the Wasmote IDE, a valid call to the Libelium Cloud Bridge service.

Parameters needed for the HTTPS call to the Libelium Cloud Bridge service are:

- **Host:** hw.libelium.com
- **Action:** GET /hw/ps?frame=[FRAME TO SEND] HTTP/1.1
- **Authorization:** Bearer [API KEY FROM Bridge]

Example using “curl” tool to simulate the HTTPS call:

```
{  
curl -X GET 'https://hw.libelium.com/hw/ps?frame=[FRAME TO SEND]' -H 'Authorization: Bearer  
[API KEY FROM Bridge]' \  
}
```

8. Documentation Changelog

From 7.1 to 7.2

- Added references to the new XBee ZigBee 3 module

From 7.0 to 7.1

- Renamed from Hive to Bridge