Programming Cloud Service

Cloud Services Guide

Document version: v7.0 - 11/2017

INDEX

1. Introduction

1.1. Programming Cloud Service basis

This guide explains the Programming Cloud Service features and how to use them. This tool permits to create new binary programs for any Plug & Sense! v15 device. The generated programs follow a basic structure which is continuously repeated performing infinite cycles based on 3 steps:

- Step 1: Read sensor data
- Step 2: Send sensor data
- Step 3: Sleep mode

Therefore, this guide explains how the user can select different parameter options so as to modify the 3 main code blocks: reading, sending and sleeping.

Regarding the sensor options, the user can select the proper Plug & Sense! model and choose the available sensor probes for each socket.

In reference to communications, it is possible to choose any of the radio modules available for the Plug & Sense! devices. Depending on the network requirements, the user should use one type or another.

Finally, the user can select the sleeping time in number of days, hours, minutes and seconds. You must keep in mind that longer sleep periods permit to save energy increasing the life of the device. Depending on the battery charging methods applied or radio used, the user should adapt this time to fit their needs.

1.2. Licenses

There are different PCS license versions:

- PCS Basic
- PCS PRO
- PCS Elite

The main differences among licenses are explained in the table below:

	Basic	PRO	Elite
Nodes to program	5	20	100
Plug & Sense! models supported	All	All	All
Sensors supported	All	All	All
Actuators control (Relay on/off in Plug & Sense! Smart Security)	Yes	Yes	Yes
Sigfox/LoRaWAN settings setup	Yes	Yes	Yes
802.15.4 / RF 868/900 MHz settings setup	Yes	Yes	Yes
4G/WiFi settings setup	Yes	Yes	Yes
GPS settings setup	Yes	Yes	Yes
HTTP (4G/WiFi)	Yes	Yes	Yes
HTTPS (4G/WiFi)	No	Yes	Yes
RF link encryption (802.15.4 / RF 868/900)	No	Yes	Yes
AES-256 payload encryption	No	Yes	Yes
Industrial Protocols support (Modbus over RS232/RS485, CAN Bus)	No	Yes	Yes
Low battery warning	Yes	Yes	Yes
Templates manager	Yes	Yes	Yes
Batch programming (generate up to 100 binaries in one click)	No	No	Yes
Price (per year)	~~150€~~ Launch Offer: 105 € (for the first year when acquired before January 31st 2018)	~~300€~~ Launch offer: 279 € (for the first year when acquired before January 31st 2018)	~~1,499€~~ Launch offer: 1,049 € (for the first year when acquired before January 31st 2018)

A license is needed to use the PCS, so please be sure to purchase and activate your license and devices as explained in the Services Cloud Manager Guide.

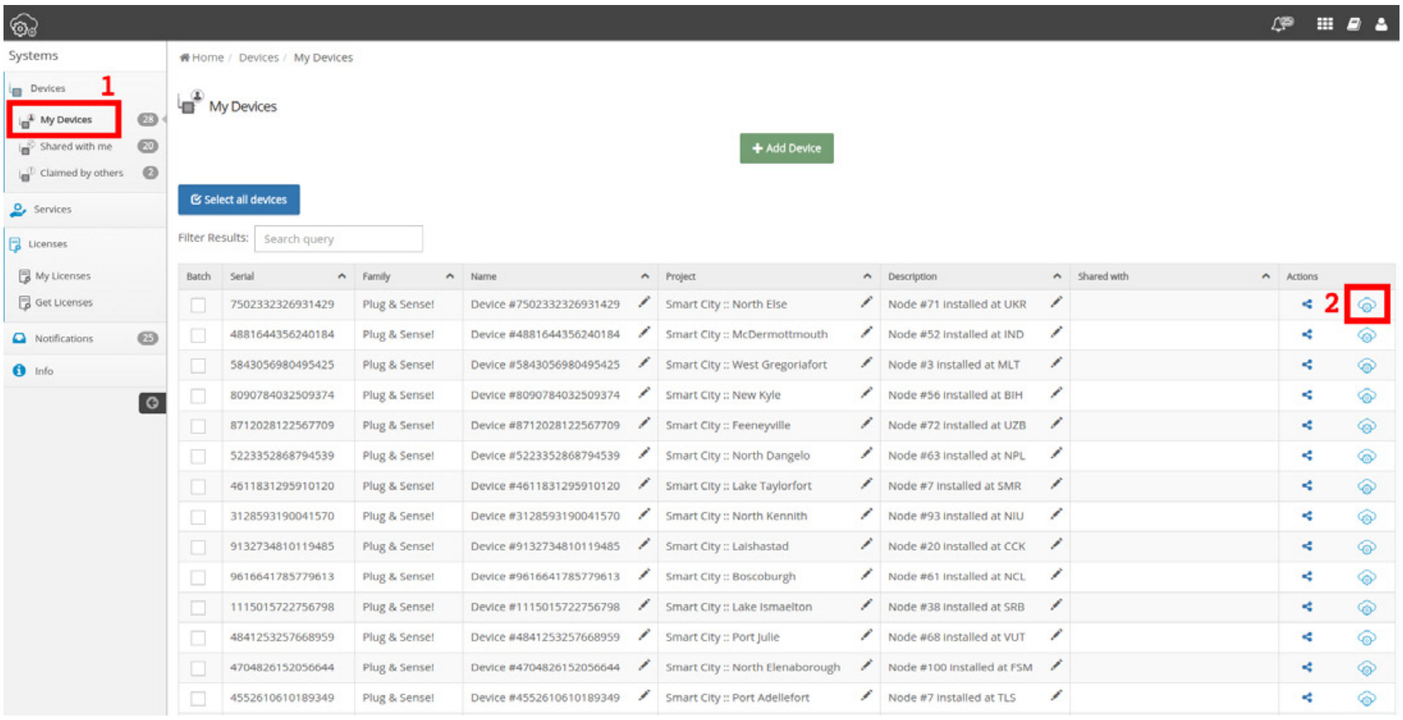
1.3. Purchasing licenses

You can acquire your licenses on the IoT Marketplace.

1.4. Accessing to the Programming Cloud Service

The user can access the PCS form at https://cloud.libelium.com/pcs.

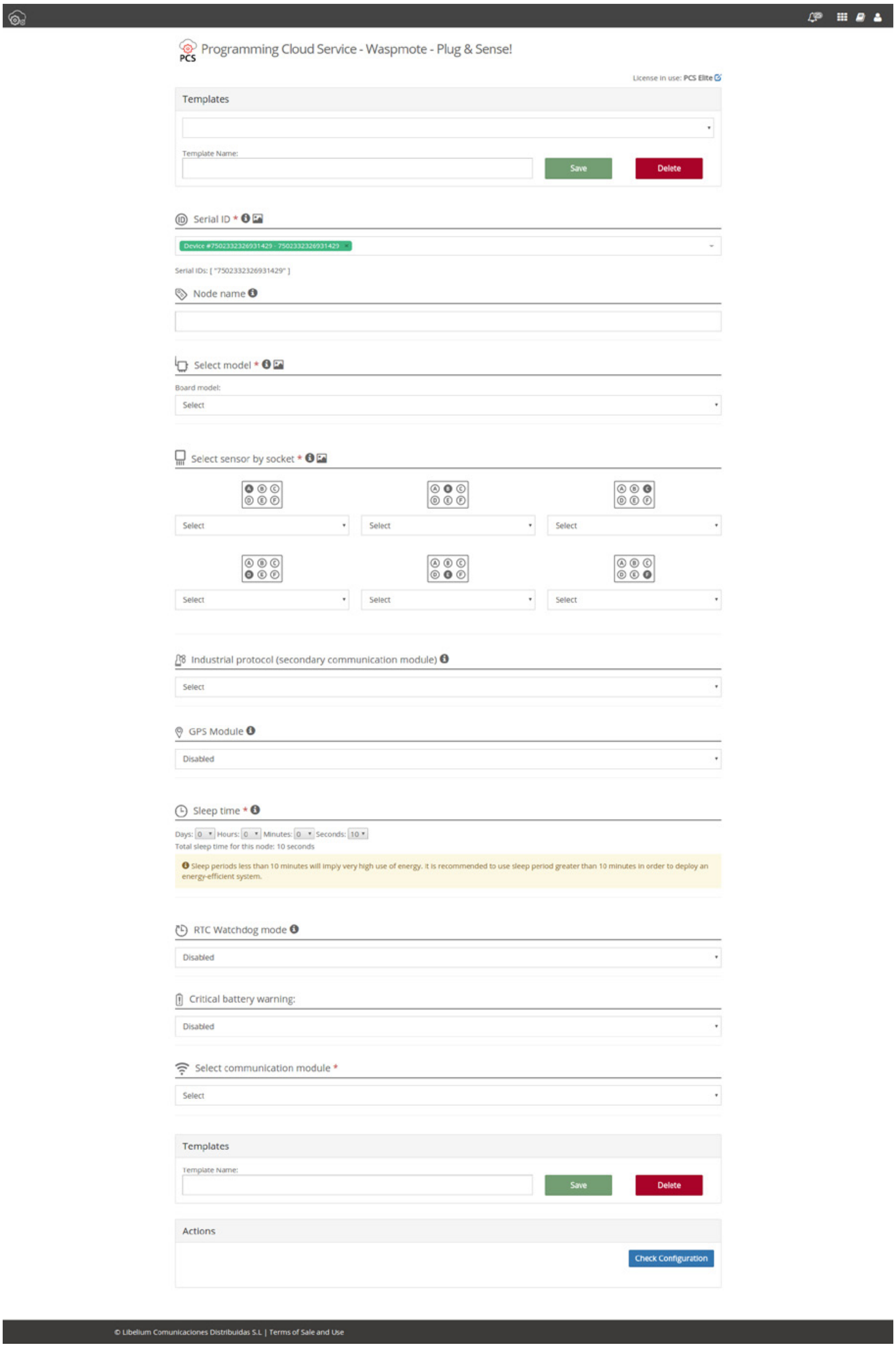
Besides, it is possible to access to the PCS form for a specific device registered in the SCM. The user must go to "My Devices" menu where all available devices will be shown. In this menu, it is possible to edit your devices. Also, if the PCS icon for a specific device is pressed, the browser will go to the PCS form for that device. So the device's serial ID will be filled in the corresponding form section.



Batch	Serial	Family	Name	Project	Description	Shared with	Actions
	7502332326931429	Plug & Sense!	Device #7502332326931429	Smart City :: North Else	Node #71 installed at UKR		
	4881644356240184	Plug & Sense!	Device #4881644356240184	Smart City :: McDermottmouth	Node #52 installed at IND		
	5843056980495425	Plug & Sense!	Device #5843056980495425	Smart City :: West Gregoriafort	Node #3 installed at MLT		
	8090784032509374	Plug & Sense!	Device #8090784032509374	Smart City :: New Kyle	Node #56 installed at BIH		
	8712028122567709	Plug & Sense!	Device #8712028122567709	Smart City :: Feeneyville	Node #72 installed at UZB		
	5223352868794539	Plug & Sense!	Device #5223352868794539	Smart City :: North Dangelo	Node #63 installed at NPL		
	4611831295910120	Plug & Sense!	Device #4611831295910120	Smart City :: Lake Taylorfort	Node #7 installed at SMR		
	3128593190041570	Plug & Sense!	Device #3128593190041570	Smart City :: North Kennith	Node #93 installed at NIU		
	9132734810119485	Plug & Sense!	Device #9132734810119485	Smart City :: Laishastad	Node #20 installed at CCK		
	9616641785779613	Plug & Sense!	Device #9616641785779613	Smart City :: Boscoburgh	Node #61 installed at NCL		
	1115015722756798	Plug & Sense!	Device #1115015722756798	Smart City :: Lake Ismaelton	Node #38 installed at SRB		
	4841253257668959	Plug & Sense!	Device #4841253257668959	Smart City :: Port Julie	Node #68 installed at VUT		
	4704826152056644	Plug & Sense!	Device #4704826152056644	Smart City :: North Elenaborough	Node #100 installed at FSM		
	4552610610189349	Plug & Sense!	Device #4552610610189349	Smart City :: Port Adellefort	Node #7 installed at TLS		

Figure: Access to the PCS form from a specific device

Finally, the user will access to the Programming Cloud Service configuration form which is explained in detail in the next section. The PCS form appearance is as follows:



Programming Cloud Service - Waspmote - Plug & Sense!

License in use: **PCS Elite**

Templates

Template Name:

Serial ID *

Device #7502332326931429 - 7502332326931429

Serial IDs: ["7502332326931429"]

Node name

Select model *

Board model: Select

Select sensor by socket *

Select	Select	Select
Select	Select	Select

Industrial protocol (secondary communication module): Select

GPS Module: Disabled

Sleep time *

Days: 0 Hours: 0 Minutes: 0 Seconds: 10

Total sleep time for this node: 10 seconds

Sleep periods less than 10 minutes will imply very high use of energy. It is recommended to use sleep period greater than 10 minutes in order to deploy an energy-efficient system.

RTC Watchdog mode: Disabled

Critical battery warning: Disabled

Select communication module *: Select

Templates

Template Name:

Actions

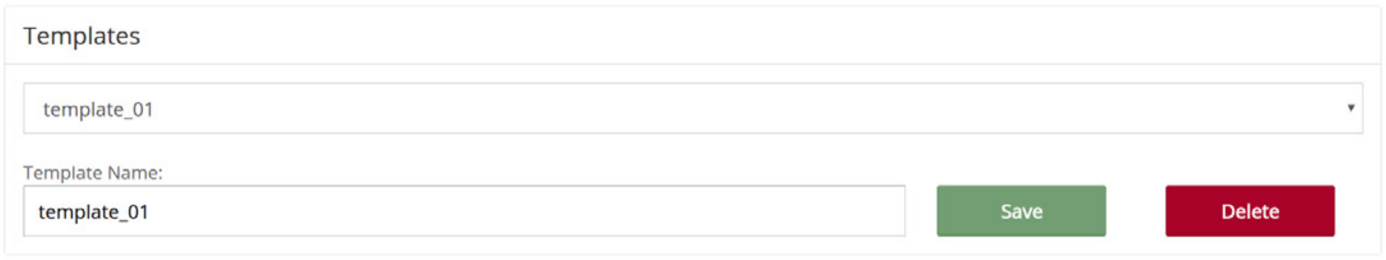
© Libelium Comunicaciones Distribuidas S.L | Terms of Sale and Use

Figure: PCS form

2. Programming Cloud Service configuration

2.1. Templates

The user can manage different program templates. So it is possible “save” a valid program and use it later by clicking the “Load” button. This is useful in order to program several nodes which belong to the same Plug & Sense! model. So the user can use the same template with different serial IDs.


Templates

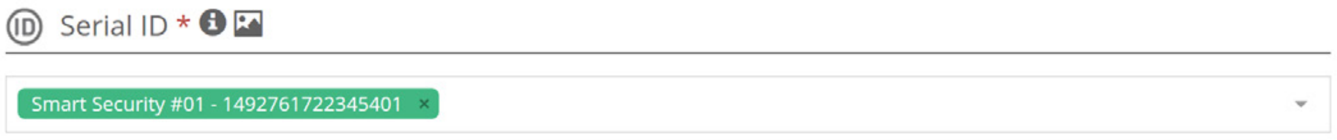
template_01

Template Name: template_01

Figure: Manage program templates

2.2. Serial ID

The “serial ID” is the unique identifier of the Plug & Sense! device. This serial ID can be found in the bottom-side sticker where the number is specified. It is a 16-digit identifier defined in hexadecimal digits. The PCS form allows the user to search for a specific serial ID among all registered devices. The user must keep in mind that the generated program will only work for the serial number specified in this field.

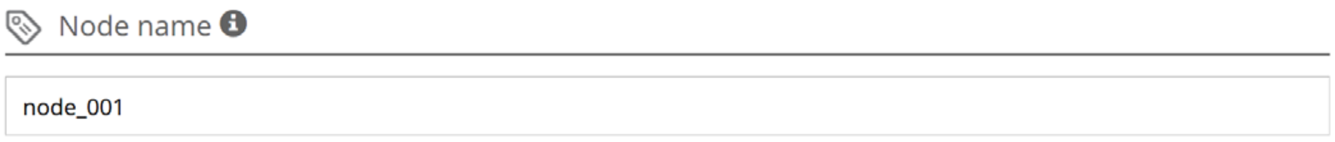

Serial ID *

Smart Security #01 - 1492761722345401

Figure: Serial identifier

2.3. Node name

This optional field permits to use a legible name for the device instead of using the serial ID only. It is possible to define a node identifier such as “node_001” which can be easier to be managed within a network of several devices. The user must keep in mind that if this field is defined, less payload size will be available for sensor data. This restriction specially applies for the XBee-PRO 802.15.4 radio. So it is recommended to rely on the “serial ID” instead of the “node name”.


Node name

node_001

Figure: Node name

2.4. Select model

The user must select the Plug & Sense! model related to the device to be programmed. The drop-down selector lists all Plug & Sense! models so the correct one must be chosen.

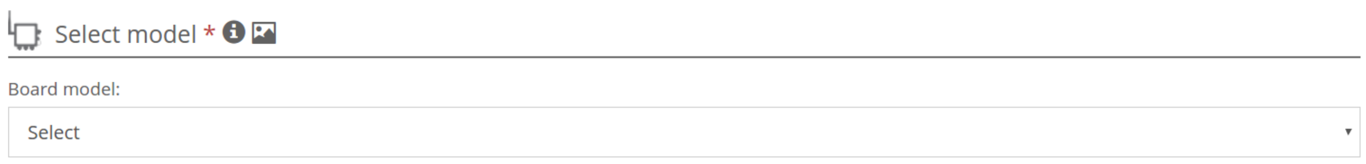


Figure: Select Plug & Sense! model

2.5. Select sensor by socket

The user must select the sensor probes to be configured in each socket. There are 6 sockets dedicated for sensor probes (from 'A' to 'F'). Depending on the Plug & Sense! Model, a different list of sensors is displayed in each drop-down selector when clicking the socket menu.

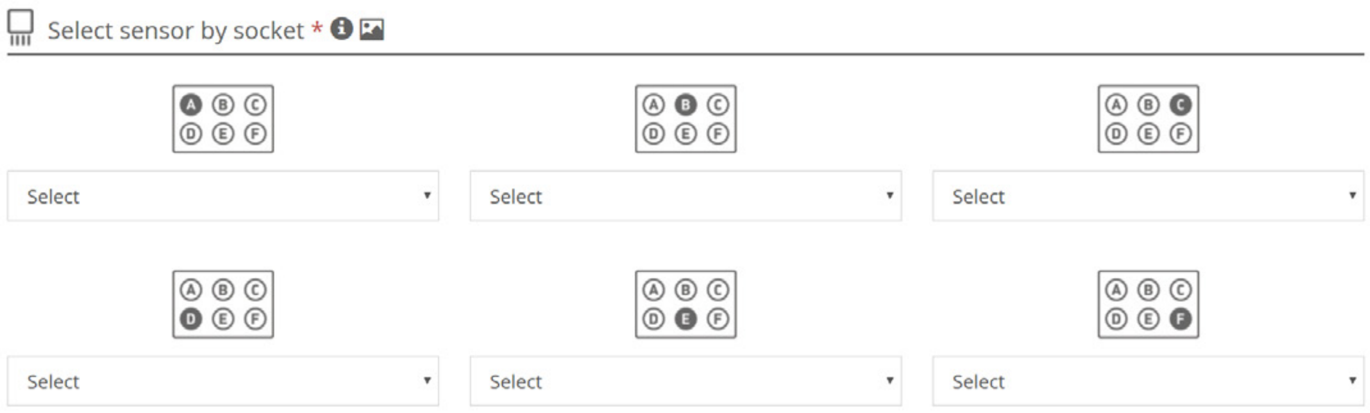
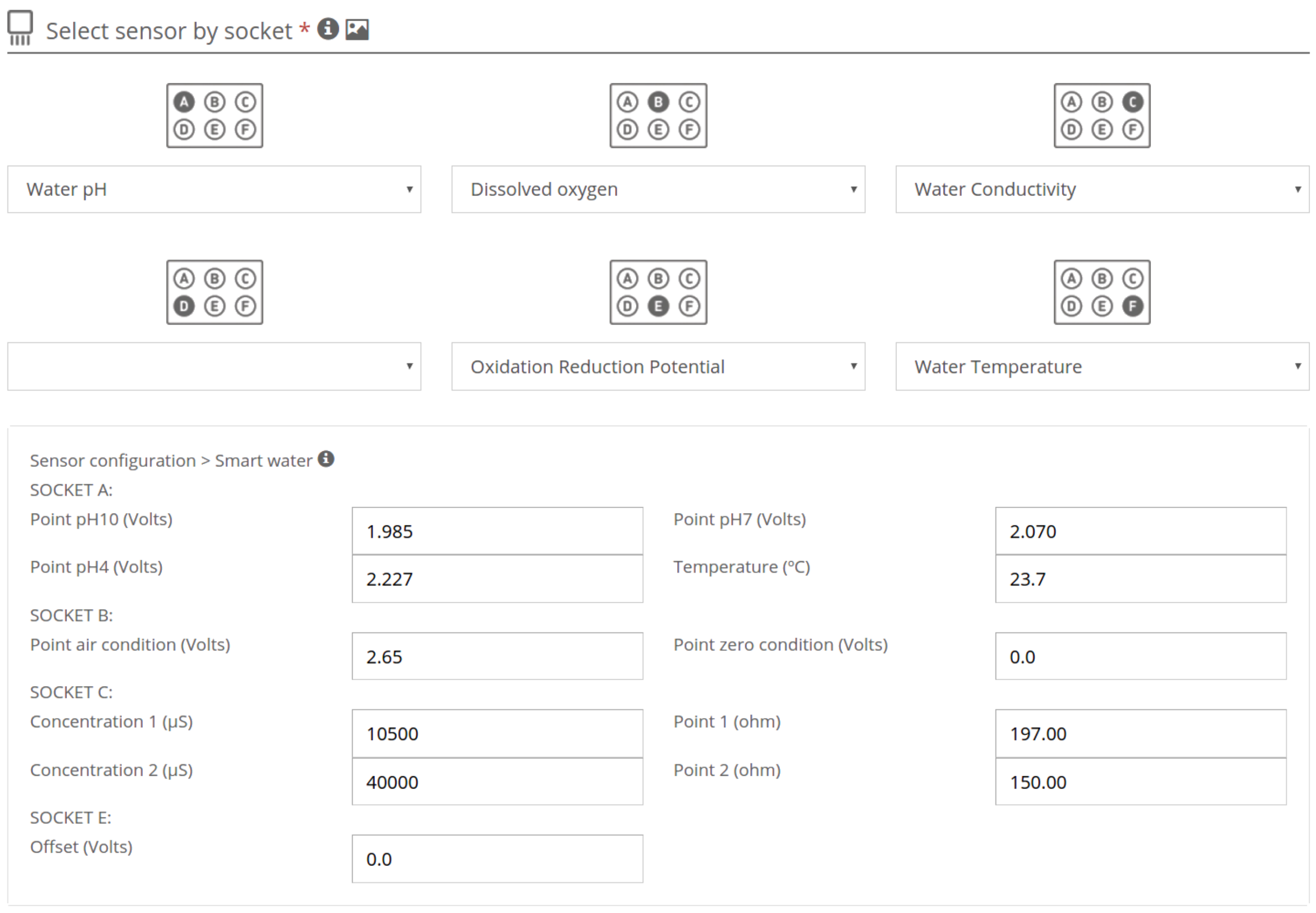


Figure: Select sensor by socket

There are specific cases when the sensor probes need special configuration parameters. The next chapters explain each special case.

2.5.1. Smart Water calibration points

When a Smart Water sensor is selected, the calibration points are displayed as a new option to be configured. Each sensor will need different calibration values depending on the points to be measured. The calibration values consist on different points to be determined. Depending on the sensor it can be a voltage level or a resistance value. The calibration points must be determined by the user through a calibration process which is described in the corresponding Smart Water Board Guide.


Select sensor by socket *

Socket A: Water pH	Socket B: Dissolved oxygen	Socket C: Water Conductivity
Socket D:	Socket E: Oxidation Reduction Potential	Socket F: Water Temperature

Sensor configuration > Smart water

SOCKET A:

Point pH10 (Volts)	1.985	Point pH7 (Volts)	2.070
Point pH4 (Volts)	2.227	Temperature (ºC)	23.7

SOCKET B:

Point air condition (Volts)	2.65	Point zero condition (Volts)	0.0

SOCKET C:

Concentration 1 (µS)	10500	Point 1 (ohm)	197.00
Concentration 2 (µS)	40000	Point 2 (ohm)	150.00

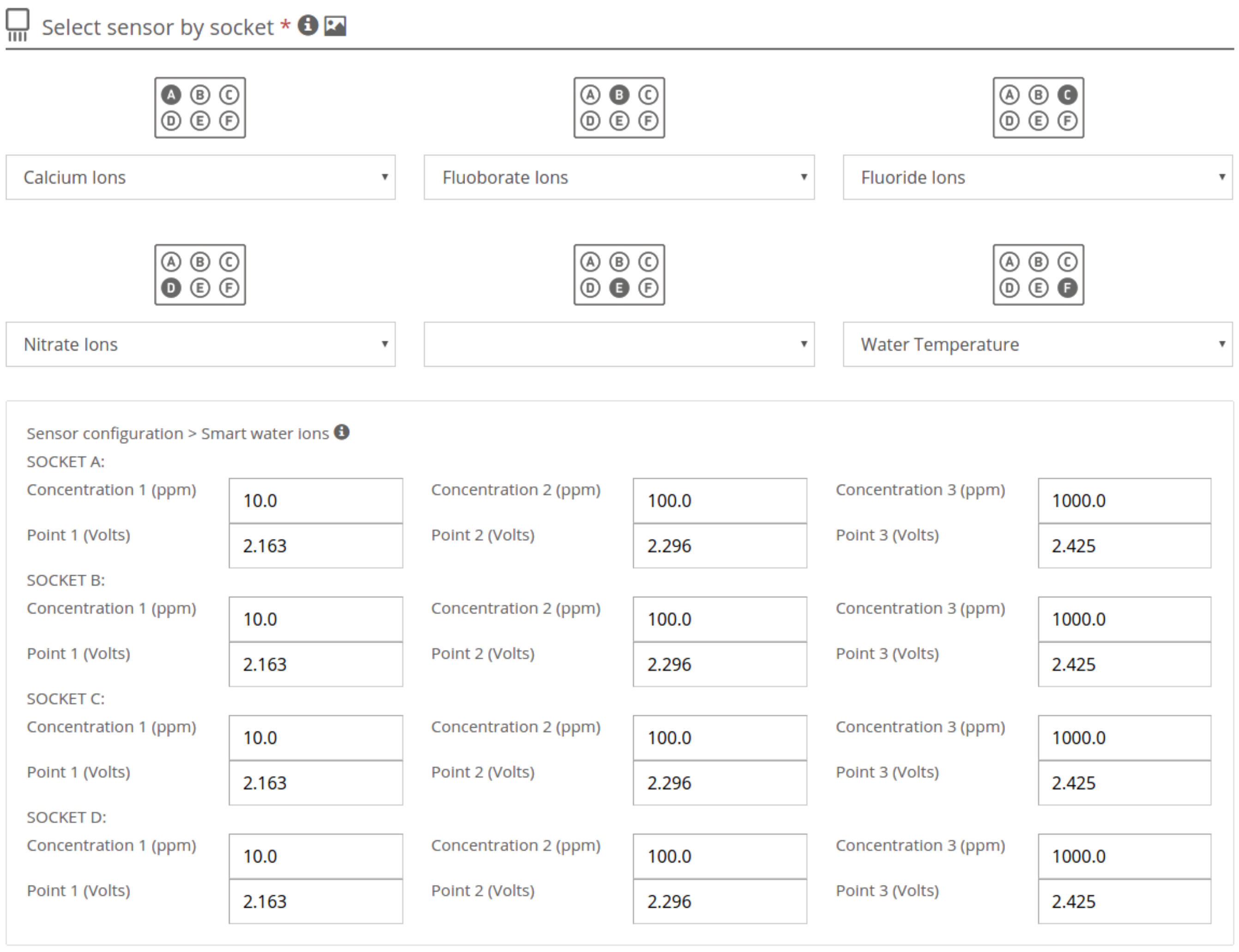
SOCKET E:

Offset (Volts)	0.0

Figure: Smart Water calibration points

2.5.2. Smart Water Ions calibration points

When a Smart Water Ions sensor is selected, the calibration points are displayed as a new option to be configured. Each sensor will need different calibration values depending on the points to be measured. The calibration values consist on 3 calibration points specified by 2 numbers: concentration in ppm units and related measured point in Volts. The calibration values must be determined by the user through a calibration process which is described on the corresponding Smart Water Ions Board Guide.



Select sensor by socket *

A: Calcium Ions	B: Fluoborate Ions	C: Fluoride Ions
D: Nitrate Ions	E:	F: Water Temperature

Sensor configuration > Smart water ions

SOCKET A:

Concentration 1 (ppm)	10.0	Concentration 2 (ppm)	100.0	Concentration 3 (ppm)	1000.0
Point 1 (Volts)	2.163	Point 2 (Volts)	2.296	Point 3 (Volts)	2.425

SOCKET B:

Concentration 1 (ppm)	10.0	Concentration 2 (ppm)	100.0	Concentration 3 (ppm)	1000.0
Point 1 (Volts)	2.163	Point 2 (Volts)	2.296	Point 3 (Volts)	2.425

SOCKET C:

Concentration 1 (ppm)	10.0	Concentration 2 (ppm)	100.0	Concentration 3 (ppm)	1000.0
Point 1 (Volts)	2.163	Point 2 (Volts)	2.296	Point 3 (Volts)	2.425

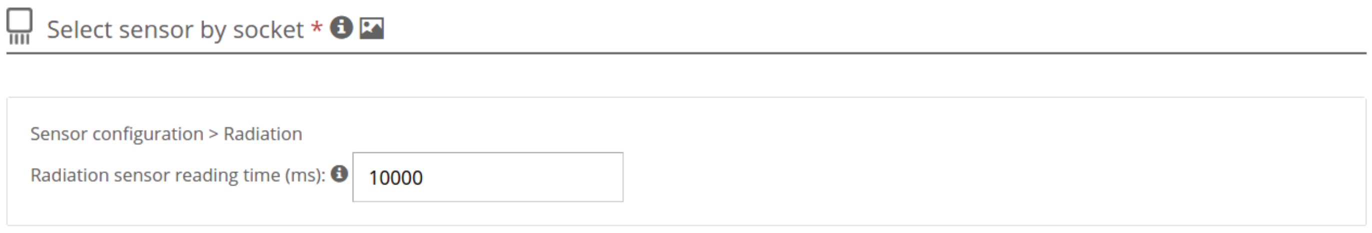
SOCKET D:

Concentration 1 (ppm)	10.0	Concentration 2 (ppm)	100.0	Concentration 3 (ppm)	1000.0
Point 1 (Volts)	2.163	Point 2 (Volts)	2.296	Point 3 (Volts)	2.425

Figure: Smart Water Ions calibration points

2.5.3. Radiation Control measuring period

For the Radiation Control model, it is mandatory to define the amount of time to be spent for the radiation sensor reading (in millisecond units).



Select sensor by socket *

Sensor configuration > Radiation

Radiation sensor reading time (ms): 10000

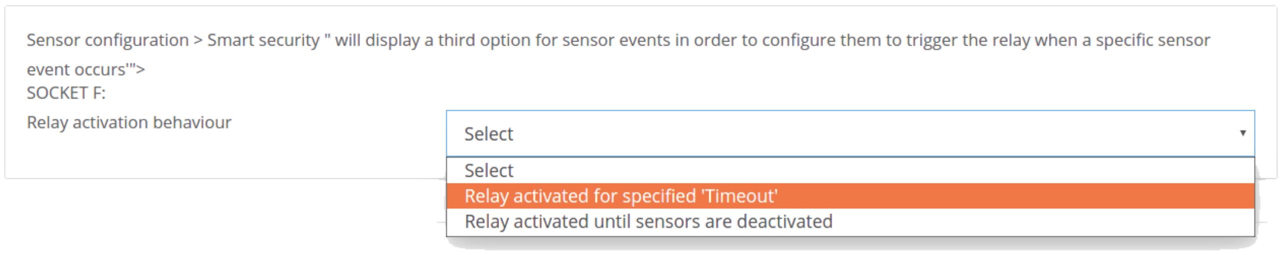
Figure: Radiation Control measuring time

2.5.4. Smart Security events configuration

This model permits to configure sensor events for certain sensors probes. The behavior of these events must be determined by the user in the Sensor Configuration window. Besides, the Smart Security model provides a relay output in order to manage an external system. This relay can be activated by any available sensor event programmed by the user.

The output relay (only available in socket F) supports 2 possible configuration modes:

- **Relay activated for specified 'Timeout':** When the relay is triggered by one of the sensor events, the device waits for the specified timeout before deactivating the relay. For instance: the water level in a tank rises up to the liquid level sensor, the interruption occurs and the relay is activated for the number of seconds defined in the form.
- **Relay activated until sensors are deactivated:** When the relay is triggered by one of the sensor events, the device waits for all sensor events to be deactivated before deactivating the relay. For instance: the PIR sensor stops detecting motion around the device.

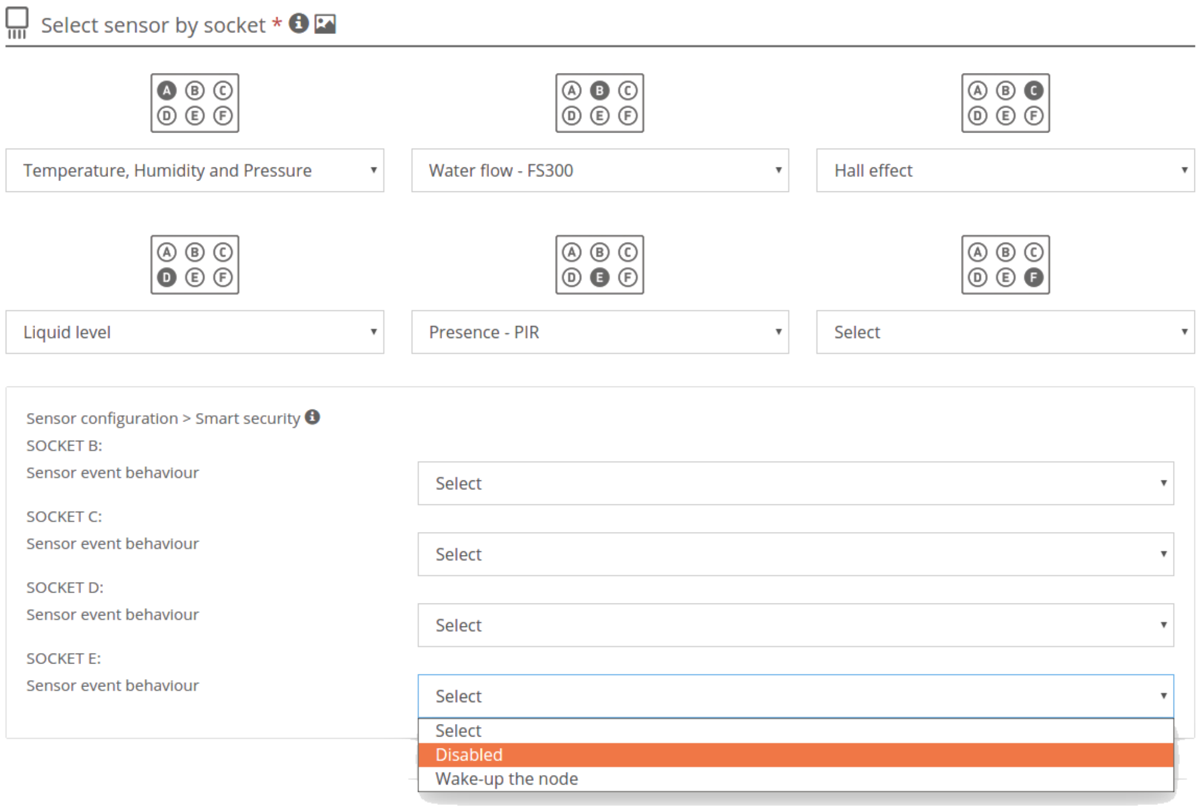

Sensor configuration > Smart security " will display a third option for sensor events in order to configure them to trigger the relay when a specific sensor event occurs'">
SOCKET F:
Relay activation behaviour: Select

Select
Relay activated for specified 'Timeout'
Relay activated until sensors are deactivated

Figure: Smart Security relay activation behavior

For each sensor event it is possible to configure the next options:

- **Disabled:** The sensor is not enabled for treating any generated event. Warning: if other sensors were enabled for waking-up the node, this disabled-sensor will be still powered on, so it will trigger the event although this sensor was not enabled for waking-up the node.
- **Wake-up the node:** If this option is selected for any of the sockets, it remains all sensors active within the sleep mode. So, it is possible to generate events and wake-up the node. When the platform wakes-up, a new sensor reading and sending is done.
- **Wake-up the node + trigger relay:** (Only available if the relay in socket 'F' was previously enabled by the user). This possibility is displayed as a new option for sensor event behavior in the drop-down menu. This means the sensor event wakes up the platform, sends sensor data and triggers the relay following the relay option selected.



Select sensor by socket *

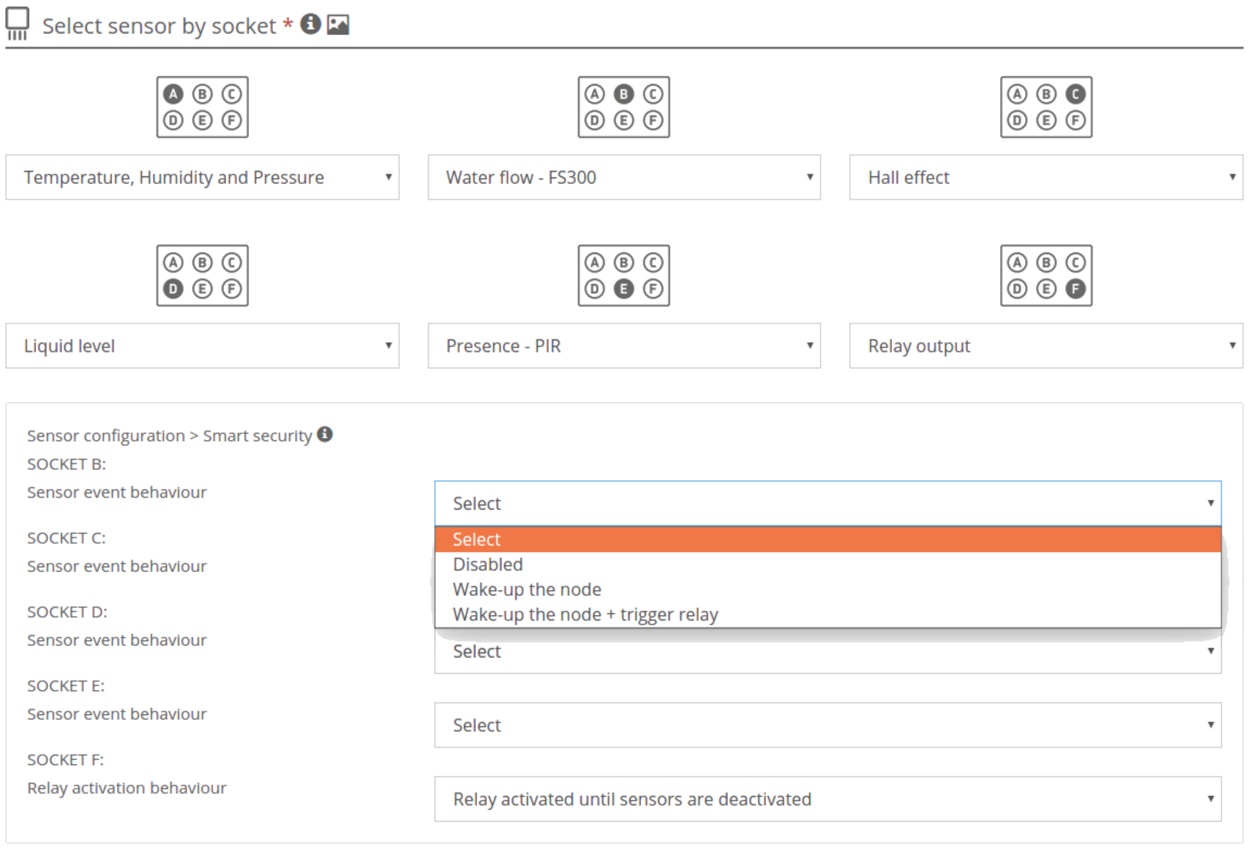
Temperature, Humidity and Pressure	Water flow - FS300	Hall effect
Liquid level	Presence - PIR	Select

Sensor configuration > Smart security

SOCKET B: Sensor event behaviour	Select
SOCKET C: Sensor event behaviour	Select
SOCKET D: Sensor event behaviour	Select
SOCKET E: Sensor event behaviour	Select

Select
Disabled
Wake-up the node

Figure: Smart Security sensor events configuration (without relay)



Select sensor by socket *

Temperature, Humidity and Pressure	Water flow - FS300	Hall effect
Liquid level	Presence - PIR	Relay output

Sensor configuration > Smart security

SOCKET B: Sensor event behaviour	Select
SOCKET C: Sensor event behaviour	
SOCKET D: Sensor event behaviour	Select
SOCKET E: Sensor event behaviour	Select
SOCKET F: Relay activation behaviour	Relay activated until sensors are deactivated

Select
Disabled
Wake-up the node
Wake-up the node + trigger relay

Figure: Smart Security sensor events configuration (with relay)

2.6. Industrial protocol

Note: *This feature is only supported by PRO and Elite license versions.*

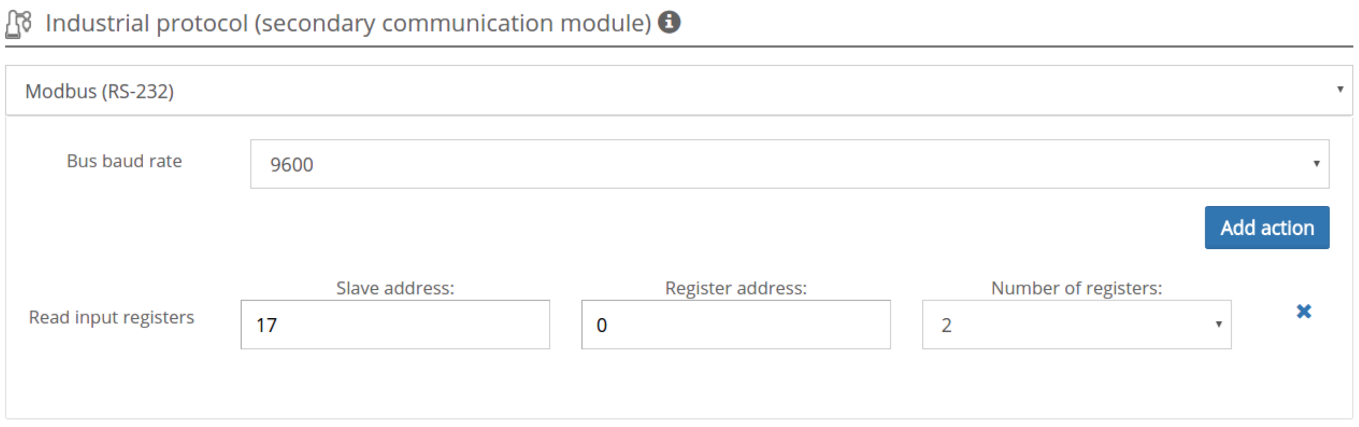
If the user purchased a Plug & Sense! device which includes an Industrial Protocol module, this means that this optional field should be managed. Besides, the PCS Basic license does not support the Industrial protocol configuration. A PRO or Elite license is needed for this purpose. The available Industrial protocols are:

- Modbus (over RS-232)
- Modbus (over RS-485)
- CAN Bus

2.6.1. Modbus over RS-232/485

Regarding the Modbus protocol, both RS-232 and RS-485 can be used in order to read input registers working with the Plug & Sense! device as master of the Modbus interface. The "input registers" function code is used to read contiguous input registers in a slave. Up to 3 different reading operations can be programmed. For each one of them, it is mandatory to define the slave address, the starting register address and the number of data (number of registers to read). Each register read from the Modbus interface is defined as 2 bytes. For each register, the first byte contains the MSB, and the second one the LSB.

Each successful reading is inserted into the Frame generated to be sent via the communication module. There is a particularity about the Modbus readings as it is explained in the "How data frames are generated" section.



Modbus (RS-232)

Bus baud rate: 9600

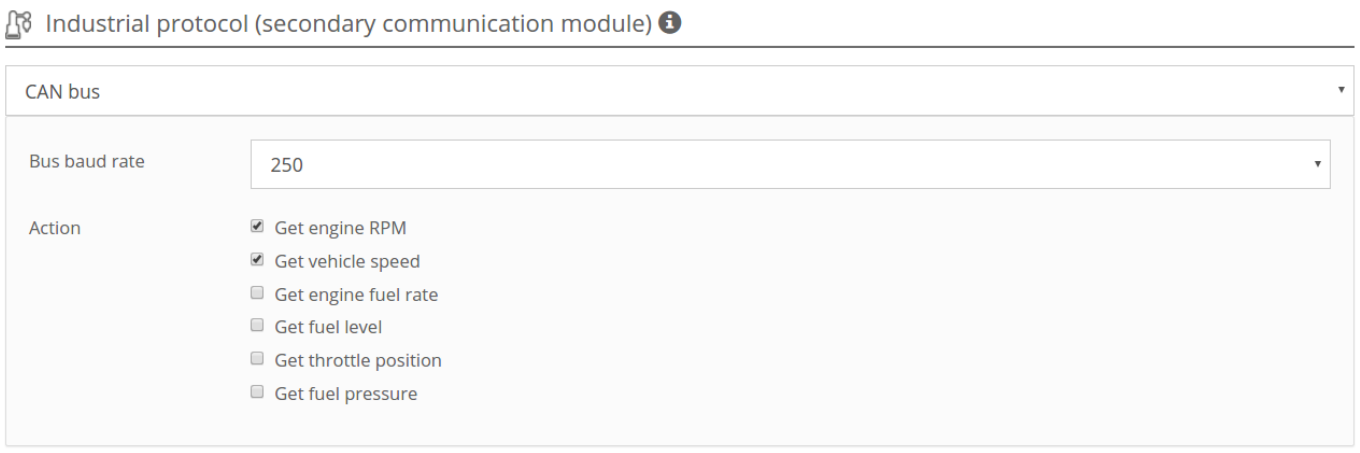
	Slave address:	Register address:	Number of registers:
Read input registers	17	0	2

Figure: Modbus protocol (RS-232 or RS-485)

2.6.2. CAN Bus

Regarding the CAN Bus protocol, several operations can be programmed according to the customer requirements:

- Get engine RPM
- Get vehicle speed (km/h)
- Get engine fuel rate (l/h)
- Get fuel level (%)
- Get throttle position (%)
- Get fuel pressure (kPa)


Industrial protocol (secondary communication module)

CAN bus

Bus baud rate: 250

Action:
- [✓] Get engine RPM
- [✓] Get vehicle speed
- [ ] Get engine fuel rate
- [ ] Get fuel level
- [ ] Get throttle position
- [ ] Get fuel pressure

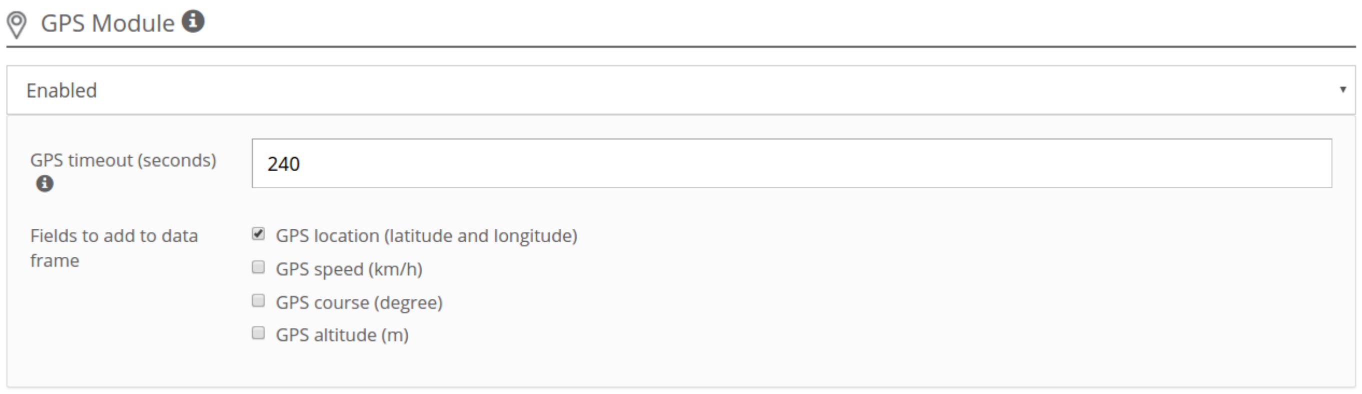
Figure: CAN Bus protocol

2.7. GPS module

If the Plug & Sense! device provides the GPS module, in this section it will be possible to enable this feature. When enabled, the timeout to wait for satellites signal must be defined in second units.

Besides, several GPS fields can be attached to the data frame:

- GPS location (latitude and longitude in degrees)
- GPS speed (km/h)
- GPS course (degree)
- GPS altitude (m)


GPS Module

Enabled

GPS timeout (seconds): 240

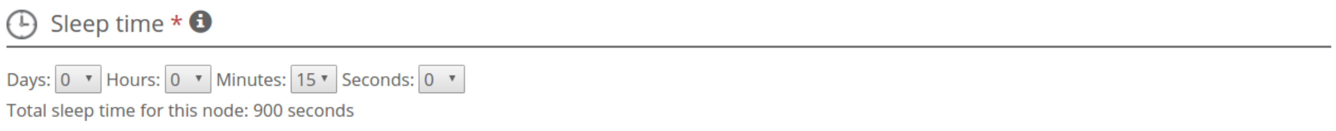
Fields to add to data frame:
- [✓] GPS location (latitude and longitude)
- [ ] GPS speed (km/h)
- [ ] GPS course (degree)
- [ ] GPS altitude (m)

Figure: GPS module

2.8. Sleep time

This field is related to the period of time the device will spent in a sleep mode before a new sensor reading and data sending are performed. This permits to keep the battery level. The user must keep in mind that the greater the sleep time is, the longer the battery life will be and vice-versa.

The user must define the period of time in the 4 fields related to this section: Days, hours, minutes and seconds.


Sleep time *

Days: 0 Hours: 0 Minutes: 15 Seconds: 0

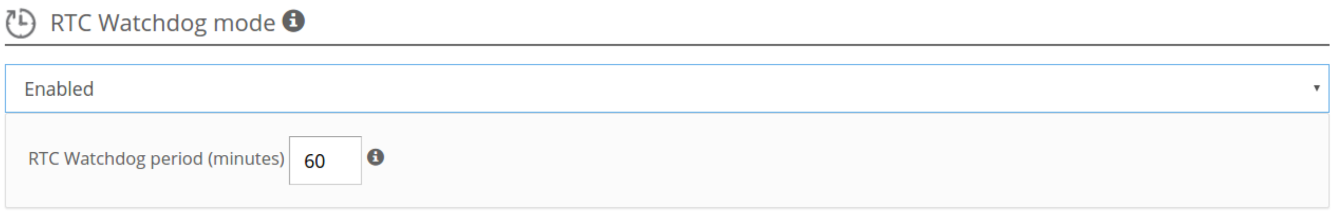
Total sleep time for this node: 900 seconds

Figure: Sleep time

Sleep periods less than 10 minutes will imply very high use of energy. It is recommended to use sleep period greater than 10 minutes in order to deploy an energy-efficient system.

2.9. RTC Watchdog

The RTC Watchdog allows the user to set up a watchdog reset as a backup feature. In the case the device experiences any kind of hanging issue, the RTC watchdog will reset the device after the watchdog period is consumed. The RTC Watchdog period must be defined in minute units. The application follows a simple rule to limit the minimum watchdog period. The minimum Watchdog period must be greater than the sleep time plus an offset of 10 minutes. The reason for applying this rule is to make sure all tasks (reading, sending and sleeping) are performed before triggering the Watchdog reset signal.


RTC Watchdog mode

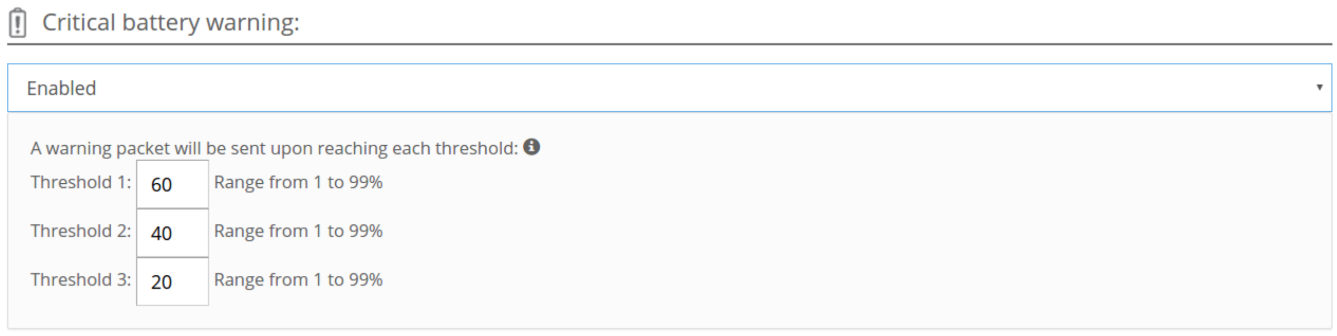
Enabled

RTC Watchdog period (minutes) 60

Figure: Select RTC watchdog settings

2.10. Critical battery warning

This optional setting permits to enable and configure a battery level warning message when battery thresholds are reached. When this field is enabled, the user must define 3 different thresholds following the next rule: Threshold 1 > Threshold 2 > Threshold 3. The thresholds are defined in % units related to battery level values. If the battery level decreases below one of these thresholds, a special warning packet is sent using the corresponding communication radio. The packet related to each threshold is sent once after crossing the threshold value and no more times. Only when the battery level gets over the threshold level, the program will be able to trigger this warning again.


Critical battery warning:

Enabled

A warning packet will be sent upon reaching each threshold:

Threshold 1:	60	Range from 1 to 99%
Threshold 2:	40	Range from 1 to 99%
Threshold 3:	20	Range from 1 to 99%

Figure: Select critical battery thresholds

The warning messages are different depending on the threshold. So, if the first threshold is exceeded, a string field is attached to the frame with the next contents: “LOW1”. The same happens with the second threshold (“LOW2”). And again the same with the third threshold (“LOW3”).

Example: if the thresholds are established to 60%, 40% and 20%. Thus, the system reads a 60% battery level in the loop ‘n’. And afterwards the loop ‘n+1’ reads 59%, then a “LOW1” warning message is sent as a string field (SENSOR_STR) within the frame.

2.11. Select communication module

The communication module provides the task of transmitting the frame with sensor data. All communication modules generate “binary” frames because this type of frame permits to insert more sensor fields in a single frame. The short-payload protocols (LoRaWAN and Sigfox) take advantage of the “tiny” frame.

For further information, refer to the Data Frame Guide.

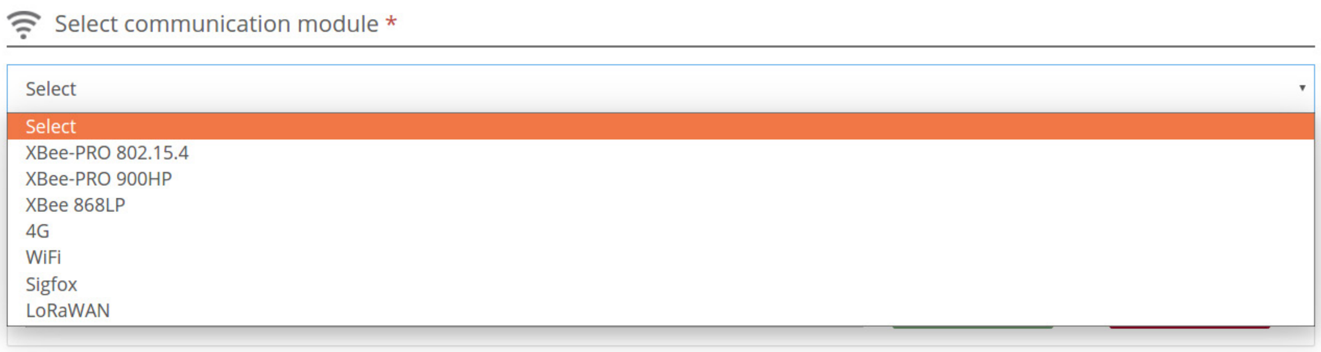


Figure: Select communication module

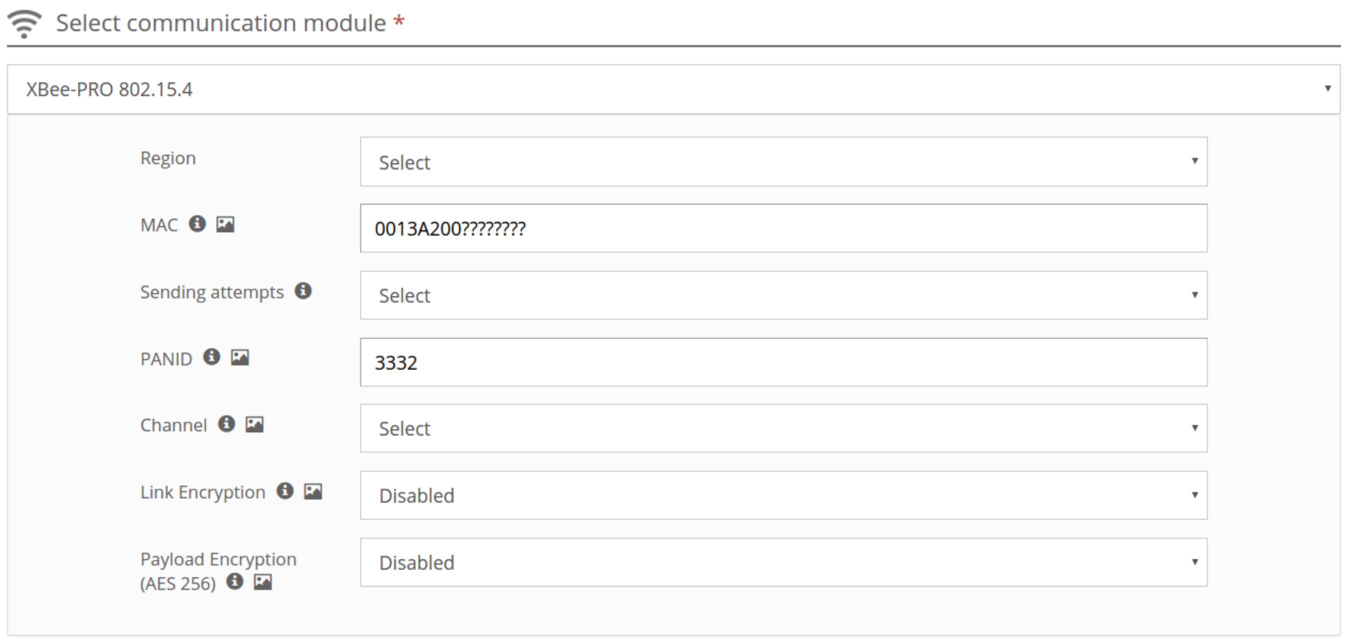
2.11.1. XBee-PRO 802.15.4

***Note:** The "Link encryption" and "Payload encryption" features are only supported by PRO and Elite license versions.*

The XBee-PRO 802.15.4 can send frames to another XBee radio. A Meshlium device should be used in order to receive the frames from the Plug & Sense! device. The PCS can configure several parameters:

- **Region:** It permits to select between XBee-PRO 802.15.4 and XBee-PRO 802.15.4 EU. You can see the radio type in the sticker in the bottom side of the Plug & Sense! device.
- **MAC:** Defines the destination MAC address. Where the packet is sent to. If you are using a Meshlium device to collect the data, the "RF modules" tab in the Manager System permits to read the Meshlium's XBee radio MAC address. The MAC addresses always start by 0013A200. The rest of the address must be correctly written down.
- **Sending attempts:** Number of sending attempts in the case there are any RF errors.
- **PANID:** Personal Area Network identifier. This is a 2-byte field. It is necessary to define it as 4 hexadecimal digits (from 0000 to FFFF). It must be the same as the receiver's.
- **Channel:** The frequency channel defined for the network. It must be the same as the receiver's channel.
- **Link encryption:** (Only available in PCS PRO/Elite licenses). This parameter enables/disables the XBee AES-128 link encryption layer. If enabled, a 16-byte encryption key must be defined for the AES-128 encryption process. It must be defined as 16 chars. The receiver must be configured with the same encryption key. If Meshlium is used as data collector, the "RF modules" tab in the Manager System permits to configure it. This encryption layer is related to "Device to Device" encryption level described in the IoT Security Infographic.
- **Payload encryption (AES-256):** (Only available in PCS PRO/Elite licenses).This parameter enables/disables the AES-256 application encryption layer. This encryption layer is related to "Device to Gateway" and "Device to Cloud" encryption level described in the IoT Security Infographic. So there are 3 possible values for this parameter: Disabled, Device to Cloud and Device to Gateway. If Meshlium is used as data collector, the "Encryption" tab in the Manager System permits to configure the "Device to Gateway" case.

For further information, refer to the 802.15.4 Networking Guide.


Select communication module *

XBee-PRO 802.15.4

Region	Select
MAC	0013A200????????
Sending attempts	Select
PANID	3332
Channel	Select
Link Encryption	Disabled
Payload Encryption (AES 256)	Disabled

Figure: XBee-PRO 802.15.4 configuration

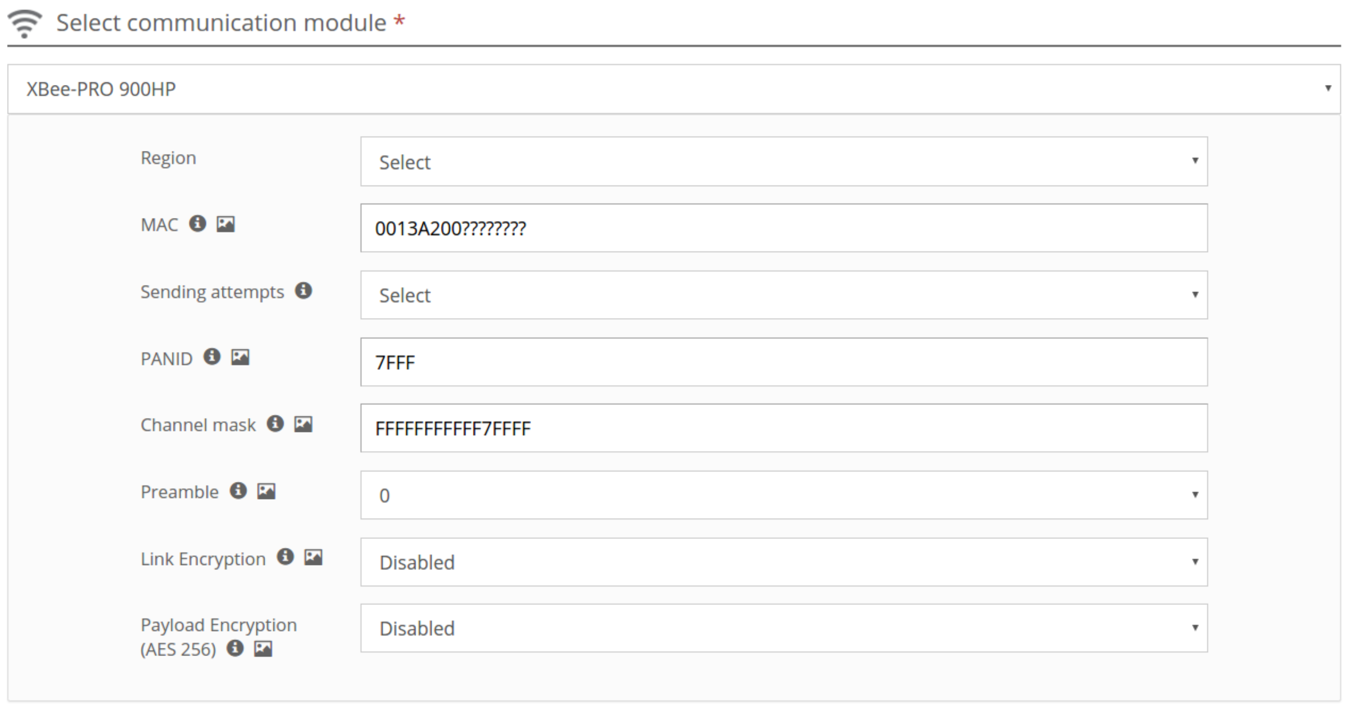
2.11.2. XBee-PRO 900HP

***Note:** The "Link encryption" and "Payload encryption" features are only supported by PRO and Elite license versions.*

The XBee-PRO 900HP can send frames to another XBee radio. A Meshlium device should be used in order to receive the frames from the Plug & Sense! device. The application can configure several parameters:

- **Region:** It permits to select between XBee-PRO 900HP US/BR/AU.
- **MAC:** Defines the destination MAC address. Where the packet is sent to. If you are using a Meshlium device to collect the data, the "RF modules" tab in the Manager System permits to read the Meshlium's XBee radio MAC address. The MAC addresses always start by 0013A200. The rest of the address must be correctly written down.
- **Sending attempts:** Number of sending attempts in the case there are any RF errors.
- **PANID:** Personal Area Network identifier. This is a 2-byte field. It is necessary to define it as 4 hexadecimal digits (from 0000 to FFFF). It must be the same as the receiver's.
- **Channel mask:** This is a bitmap to select the channels used for RF communications. It goes from 0000000001FFFFFF to FFFFFFFFFFFFFFFF. It must be the same as the receiver's.
- **Preamble:** The preamble defined for the network must be the same as the receiver's.
- **Link encryption:** (Only available for the PRO/Elite licenses). This parameter enables/disables the XBee AES-128 link encryption layer. If enabled, a 16-byte encryption key must be defined for the AES-128 encryption process. It must be defined as 16 chars. The receiver must be configured with the same encryption key. If Meshlium is used as data collector, the "RF modules" tab in the Manager System permits to configure it. This encryption layer is related to "Device to Device" encryption level described in the IoT Security Infographic.
- **Payload encryption (AES-256):** (Only available in PCS PRO/Elite licenses). This parameter enables/disables the AES-256 application encryption layer. This encryption layer is related to "Device to Gateway" and "Device to Cloud" encryption level described in the IoT Security Infographic. So there are 3 possible values for this parameter: Disabled, Device to Cloud and Device to Gateway. If Meshlium is used as data collector, the "Encryption" tab in the Manager System permits to configure the "Device to Gateway" case.

For further information, refer to the 900 Networking Guide.


Select communication module *

XBee-PRO 900HP

Region	Select
MAC	0013A200????????
Sending attempts	Select
PANID	7FFF
Channel mask	FFFFFFFFFFF7FFFF
Preamble	0
Link Encryption	Disabled
Payload Encryption (AES 256)	Disabled

Figure: XBee-PRO 900HP configuration

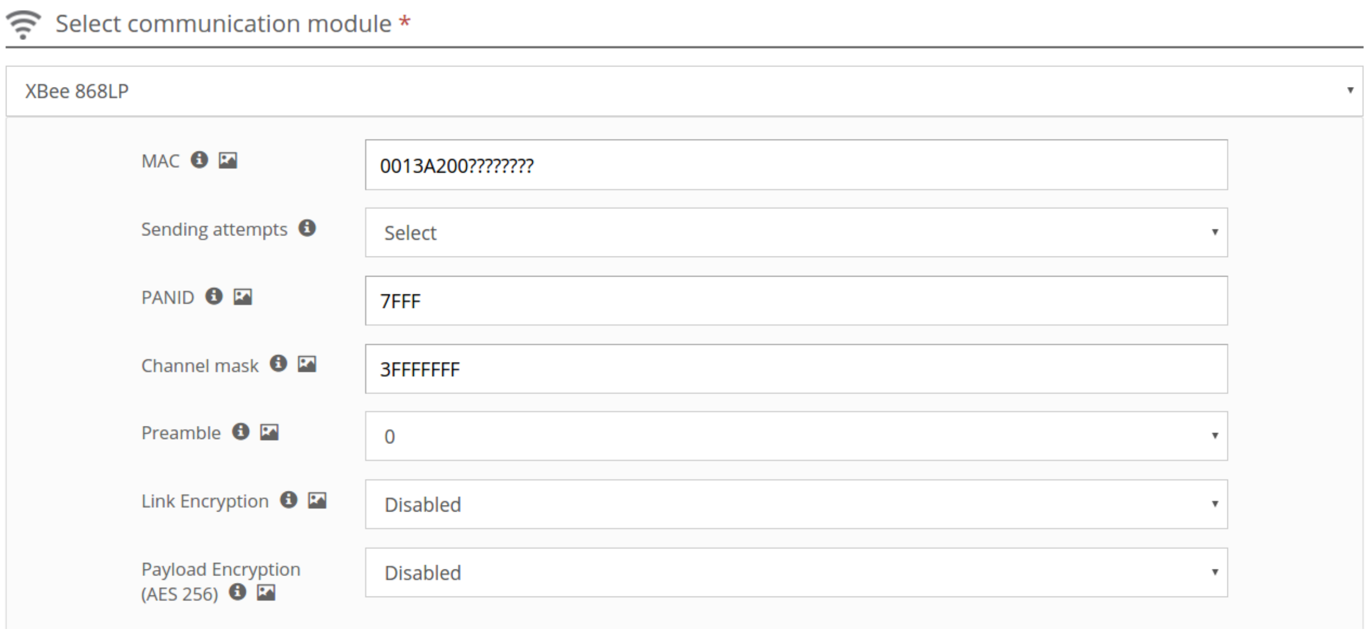
2.11.3. XBee 868LP

***Note:** The "Link encryption" and "Payload encryption" features are only supported by PRO and Elite license versions.*

The XBee 868LP can send frames to another XBee radio. A Meshlium device should be used in order to receive the frames from the Plug & Sense! device. The application can configure several parameters:

- **MAC:** Defines the destination MAC address. Where the packet is sent to. If you are using a Meshlium device to collect the data, the "RF modules" tab in the Manager System permits to read the Meshlium's XBee radio MAC address. The MAC addresses always start by 0013A200. The rest of the address must be correctly written down.
- **Sending attempts:** Number of sending attempts in the case there are any RF errors.
- **PANID:** Personal Area Network identifier. This is a 2-byte field. It is necessary to define it as 4 hexadecimal digits (from 0000 to FFFF). It must be the same as the receiver's.
- **Channel mask:** This is a bitmap to select the channels used for RF communications. It goes from 00000000 to 3FFFFFFF.
- **Preamble:** The preamble defined for the network must be the same as the receiver's.
- **Link encryption:** (Only available in PCS PRO/Elite licenses). This parameter enables/disables the XBee AES-128 link encryption layer. If enabled, a 16-byte encryption key must be defined for the AES-128 encryption process. It must be defined as 16 chars. The receiver must be configured with the same encryption key. If Meshlium is used as data collector, the "RF modules" tab in the Manager System permits to configure it. This encryption layer is related to "Device to Device" encryption level described in the IoT Security Infographic.
- **Payload encryption (AES-256):** (Only available in PCS PRO/Elite licenses). This parameter enables/disables the AES-256 application encryption layer. This encryption layer is related to "Device to Gateway" and "Device to Cloud" encryption level described in the IoT Security Infographic. So there are 3 possible values for this parameter: Disabled, Device to Cloud and Device to Gateway. If Meshlium is used as data collector, the "Encryption" tab in the Manager System permits to configure the "Device to Gateway" case.

For further information, refer to the 868 Networking Guide.


Select communication module *

XBee 868LP

MAC	0013A200????????
Sending attempts	Select
PANID	7FFF
Channel mask	3FFFFFFF
Preamble	0
Link Encryption	Disabled
Payload Encryption (AES 256)	Disabled

Figure: XBee 868LP configuration

2.11.4. 4G

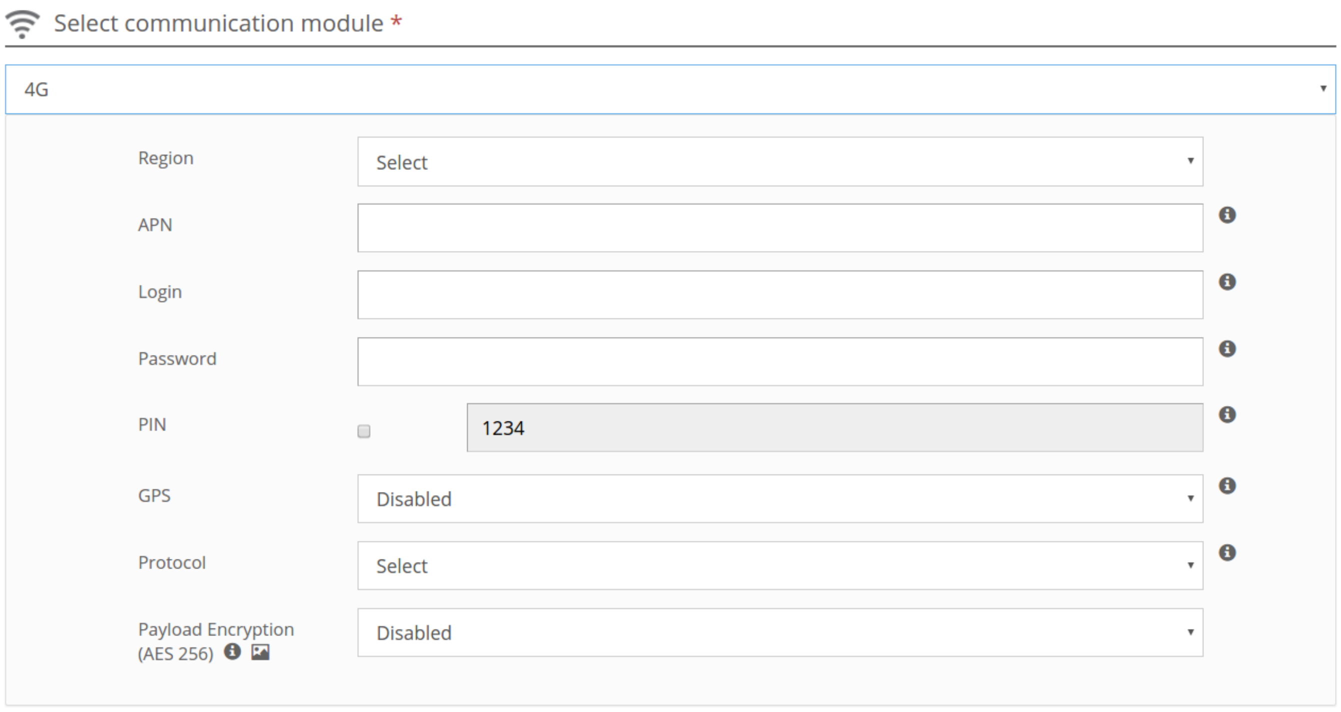
Note: *The "Payload encryption" and "HTTPS" features are only supported by PRO and Elite license versions.*

The 4G radio supports different protocols: SMS, TCP, HTTP and HTTPS. It permits to send data to an external system.

The application can configure several parameters:

- **Region:** It permits to select the radio version between Europe and Brazil, Americas or Australia.
- **APN:** Access Point Name of the Mobile Network Operator network.
- **Login:** Login of the Mobile Network Operator network.
- **Password:** Password of the Mobile Network Operator network.
- **PIN:** Defines the SIM card's PIN number.
- **GPS:** It permits to enable/disable the GPS feature. Not available for Australian version.
- **Protocol:**
  - **SMS:** The data is sent to the defined phone number as a text message via SMS. The SMS body consists on the frame as hexadecimal digits.
  - **HTTP:** The data is sent to a specific server as an HTTP GET request. There are several parameters to fill. The request will be sent as http://<host>:<port><path>?frame=<data>. Where <data> is the sensor data frame created by the device. It follows the structure defined in the Data Frame Guide. The rest of the parameters must be defined by the user:
    - Host: Consists on the host's URL or IP address.
    - Port: The remote port the server is listening for.
    - Path: The resource in order to access to the server.
  - **HTTPS:** (Only available in PCS PRO/Elite licenses). The data is sent to a specific server as an HTTP GET request. There are several parameters to fill. The request will be sent as https://<host>:<port><path>?frame=<data>. Where <data> is the sensor data frame created by the device. It follows the structure defined in the Data Frame Guide. The rest of the parameters must be defined by the user:
    - Host: Consists on the host's URL or IP address.
    - Port: The remote port the server is listening for.
    - Path: The resource in order to access to the server.
    - Certificate: The CA certificate that must be installed in the 4G radio in order to open the secure connection to the server.
  - TCP: In the case of the TCP connection, the frame is sent to the host in raw format (hexadecimal digits) with no headers. The rest of the parameters must be defined by the user:
    - Host: Consists on the host's URL or IP address.
    - Port: The remote port the server is listening for.
- **Payload encryption (AES-256):** (Only available in PCS PRO/Elite licenses). This parameter enables/disables the AES-256 application encryption layer. This encryption layer is related to "Device to Gateway" and "Device to Cloud" encryption level described in the IoT Security Infographic. So there are 3 possible values for this parameter: Disabled, Device to Cloud and Device to Gateway. If Meshlium is used as data collector, the "Encryption" tab in the Manager System permits to configure the "Device to Gateway" case.

For further information, refer to the 4G Networking Guide.


Select communication module *

4G

Region	Select
APN	
Login	
Password	
PIN	☐ 1234
GPS	Disabled
Protocol	Select
Payload Encryption (AES 256)	Disabled

Figure: 4G configuration

2.11.5. WiFi

Note: *The "Payload encryption" and "HTTPS" features are only supported by PRO and Elite license versions.*

The WiFi radio supports different protocols: TCP, HTTP, HTTPS and send a frame to Meshlium. It permits to send data to an external system.

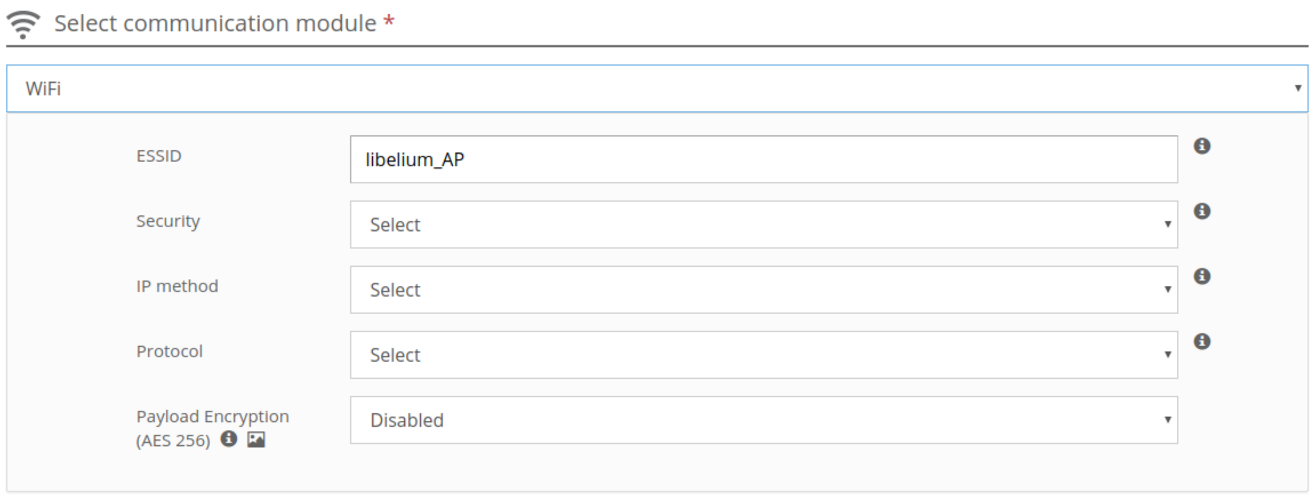
The application can configure several parameters:

- **ESSID:** Extended service set identifier of the Access Point to join.
- **Security:** It permits to configure the security settings for OPEN, WEP64, WEP128, WPA and WPA2. Depending on the security option, a different format of password is required.
  - OPEN: no security.
  - WEP64: 10 hexadecimal digits.
  - WEP128: 26 hexadecimal digits.
  - WPA/WPA2: at least 8 characters for the password.
- **IP method:** It permits to select Static of DHCP methods.
  - Static: IP address, DNS, gateway and netmask addresses must be defined
  - DHCP: no parameters required.
- **Protocol:**
  - **HTTP:** The data is sent to a specific server as an HTTP GET request. There are several parameters to fill. The request will be sent as http://<host>:<port><path>?frame=<data>. Where <data> is the sensor data frame created by the device. It follows the structure defined in the Data Frame Guide. The rest of the parameters must be defined by the user:
    - Host: Consists on the host's URL or IP address.
    - Port: The remote port the server is listening for.
    - Path: The resource in order to access to the server.

- **HTTPS:** (Only available in PCS PRO/Elite licenses). The data is sent to a specific server as an HTTP GET request. There are several parameters to fill. The request will be sent as https://<host>:<port><path>?frame=<data>. Where <data> is the sensor data frame created by the device. It follows the structure defined in the Data Frame Guide. The rest of the parameters must be defined by the user:
  - Host: Consists on the host's URL or IP address.
  - Port: The remote port the server is listening for.
  - Path: The resource in order to access to the server.
  - Certificate: The CA certificate that must be installed in the WiFi radio in order to open the secure connection to the server.
- **TCP:** In the case of the TCP connection, the frame is sent to the host in raw format (hexadecimal digits) with no headers. The rest of the parameters must be defined by the user:
  - Host: Consists on the host's URL or IP address.
  - Port: The remote port the server is listening for.

• **Payload encryption (AES-256):** (Only available in PCS PRO/Elite licenses). This parameter enables/disables the AES-256 application encryption layer. This encryption layer is related to "Device to Gateway" and "Device to Cloud" encryption level described in the IoT Security Infographic. So there are 3 possible values for this parameter: Disabled, Device to Cloud and Device to Gateway. If Meshlium is used as data collector, the "Encryption" tab in the Manager System permits to configure the "Device to Gateway" case.

For further information, refer to the WiFi Networking Guide.


Select communication module *

WiFi

ESSID	libelium_AP
Security	Select
IP method	Select
Protocol	Select
Payload Encryption (AES 256)	Disabled

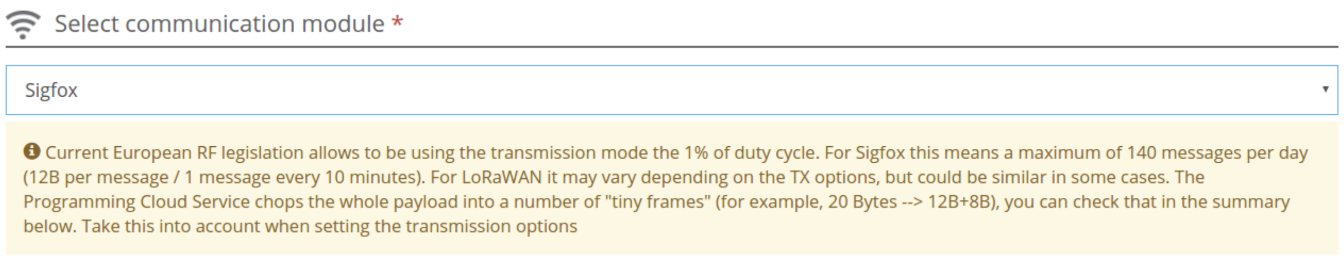
Figure: WiFi configuration

2.11.6. Sigfox

The Sigfox radio permits to send 12-byte packets to the Sigfox Cloud wherever the coverage and Sigfox standard is supported. For further information about coverage, please refer to Sigfox website. No special configuration is needed for this protocol.

The maximum payload is 12 bytes. In order to send data frames, the “tiny frame” format was designed. It permits to create several tiny frames from an original binary frame.

For further information, refer to the Sigfox Networking Guide.



Select communication module *

Sigfox

Current European RF legislation allows to be using the transmission mode the 1% of duty cycle. For Sigfox this means a maximum of 140 messages per day (12B per message / 1 message every 10 minutes). For LoRaWAN it may vary depending on the TX options, but could be similar in some cases. The Programming Cloud Service chops the whole payload into a number of "tiny frames" (for example, 20 Bytes --> 12B+8B), you can check that in the summary below. Take this into account when setting the transmission options

Figure: Sigfox configuration

2.11.7. LoRaWAN

The LoRaWAN radio permits to send variable-size packets to the LoRaWAN cloud system used by the user. For this purpose, a special LoRaWAN gateway is needed on the user side to be able to connect the Plug & Sense! device to the network.

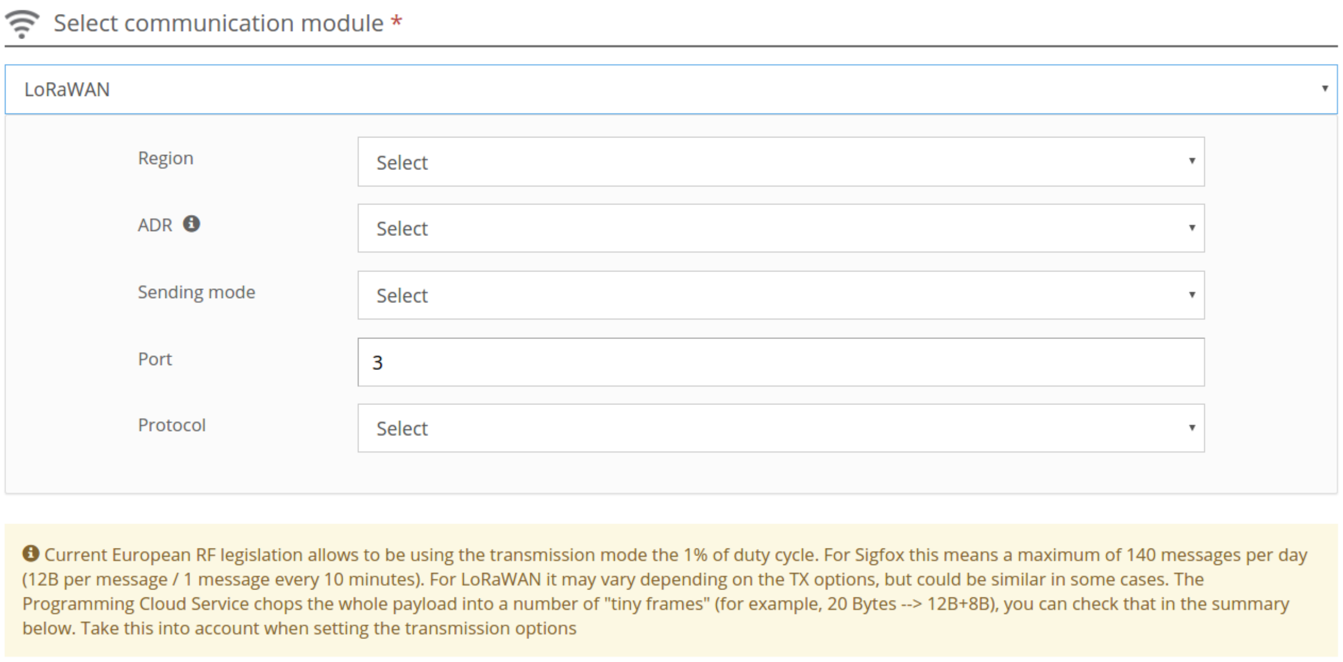
The maximum payload is variable and it depends on the network conditions. In the worst case, the payload can decrease down to 11 bytes in the American version. In order to send data frames, the “tiny frame” format was designed. It permits to create several tiny frames from an original binary frame.

The application can configure several parameters:

- Region: It permits to select between LoRaWAN EU and LoRaWAN US.
- ADR: It permits to enable/disable the Adaptive Data Rate scheme which optimizes data rates, air-time and energy consumption in the network.
- Sending mode: It permits to select between confirmed (with ACK) or unconfirmed (without ACK) transmissions.
- Port: Defines the port used for data reception in the LoRaWAN back-end. Range: 1 to 223.
- Protocol: Defines the protocol preferred by the user:
  - ABP (Association By Personalization):
    - DevEUI: The Device EUI identifies the LoRaWAN radio in the global network. It is defined by a 8-byte identifier. The user must insert the 16 hexadecimal digits that define the identifier. It must be the same as the one registered in the user back-end.
    - DevAddress: The Device Address identifies the LoRaWAN radio in the local network. It must be the same as the one registered in the user back-end.
    - NwkSKey: The Network Session Key is used for generating the message integrity check. It is defined by a 16-byte key. The user must insert the 32 hexadecimal digits that define the key. It must be the same as the one registered in the user back-end.
    - AppSKey: The Application Session Key is used for payload encryption. It is defined by a 16-byte key. The user must insert the 32 hexadecimal digits that define the key. It must be the same as the one registered in the user back-end.
  - OTAA (Over The Air Activation):
    - DevEUI: The Device EUI identifies the LoRaWAN radio in the global network. It is defined by an 8-byte identifier. The user must insert the 16 hexadecimal digits that define the identifier. It must be the same as the one registered in the user back-end.

- AppEUI: The Application EUI identifies the user's LoRaWAN back-end application. It is defined by a 8-byte identifier. The user must insert the 16 hexadecimal digits that define the identifier. It must be the same as the one registered in the user back-end.
- AppKey: The Application Key is used for the Over The Air Activation. It is defined by a 16-byte key. The user must insert the 32 hexadecimal digits that define the key. It must be the same as the one registered in the user back-end.

For further information, refer to the LoRaWAN Networking Guide.


Select communication module *

LoRaWAN

Region	Select
ADR	Select
Sending mode	Select
Port	3
Protocol	Select

Current European RF legislation allows to be using the transmission mode the 1% of duty cycle. For Sigfox this means a maximum of 140 messages per day (12B per message / 1 message every 10 minutes). For LoRaWAN it may vary depending on the TX options, but could be similar in some cases. The Programming Cloud Service chops the whole payload into a number of "tiny frames" (for example, 20 Bytes --> 12B+8B), you can check that in the summary below. Take this into account when setting the transmission options

Figure: LoRaWAN configuration

2.12. Check configuration

After filling the form, the user must press the "Check Configuration" button which will provide the list of the selected preferences. In the end, a text file is generated according to all choices given by the user.

Besides, information about the data frames will be also provided including the different sensor fields to be added during the program execution. For more information refer to the Data Frame Guide.

2.13. Compile and Download Binary

The "Compile and Download Binary" button generates the binary file to be upgraded into the Plug & Sense! device. When the button is pressed, a zip file is downloaded to the computer. After extracting the contents of the zip file, you will find a summary of the options selected in the PCS form and the corresponding binary file which name follows the next format: <serial_id>-<yy>-<mm>-<dd>-<hh>-<mm>-<ss>.hex.

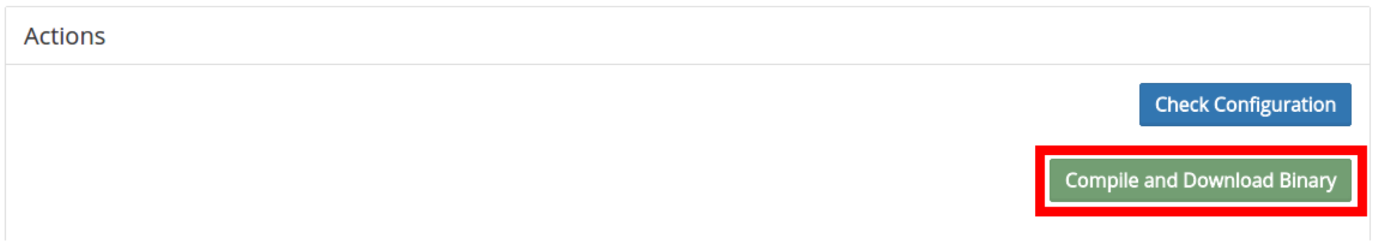


Figure: Compile and download binary

2.14. Upload a program to Plug & Sense!

Libelium Smart Devices App is an important tool developed by Libelium that allows users install programs to Plug & Sense! devices.

2.14.1. Smart Devices App installation

First of all and before installing anything, users have to take into account the platform where the application is going to be installed. To install the Libelium Smart Devices App, it is compulsory to have installed the JDK 1.8. If it is not installed in the computer, you can follow the steps and download it from this website:

https://docs.oracle.com/javase/8/docs/technotes/guides/install/install_overview.html

Once installed JDK, users can download the application using the appropriate link depending on the operative system:

- Ubuntu: http://downloads.libelium.com/smart_device_app/SmartDeviceApp_linux64.zip
- Windows: http://downloads.libelium.com/smart_device_app/SmartDeviceApp_windows32.zip
- Mac: http://downloads.libelium.com/smart_device_app/SmartDeviceApp_macosx64.zip

Then customers only have to extract the content of the SmartDeviceApp zip file downloaded in a place with the right permissions, and finally execute the file called "SmartDeviceApp" that will initialize the application. Please, note that the extension of this file will depend on the operating system the user is using at the moment (.sh for Linux and OSX, and .bat for Windows).

2.14.2. Binary update

When the Smart Devices App is launched the user must go to the "Plug&Sense!" tab. In this window, the user can select the firmware they would like to upgrade to the device.

Figure: Smart Devices App

Besides, the Plug & Sense! device must be connected to the USB port and switched on (press the On/Off button to turn it on).

Figure: Connect the Plug & Sense! device via USB port

Finally, in the "USB settings" section it is possible to refresh the available USB ports where the Plug & Sense! port identifier should be found. The user must press the "Install" button in order to upgrade the binary file.

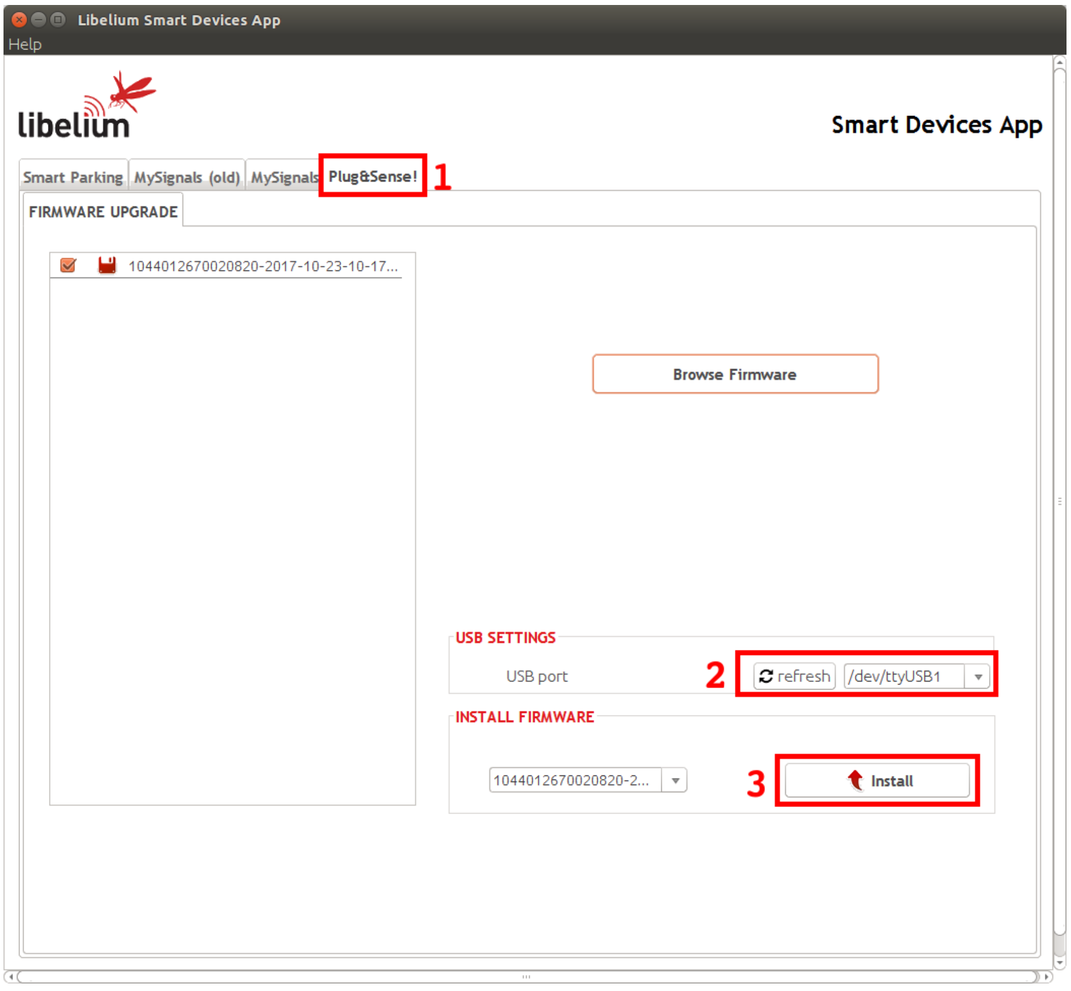
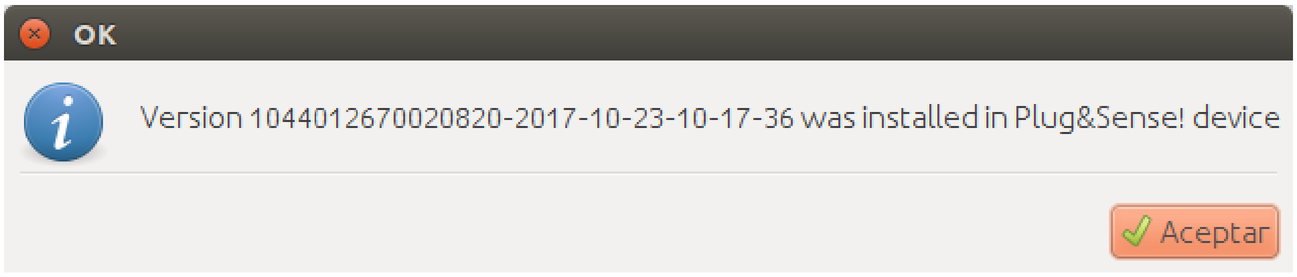


Figure: Upgrading binary to Plug & Sense!

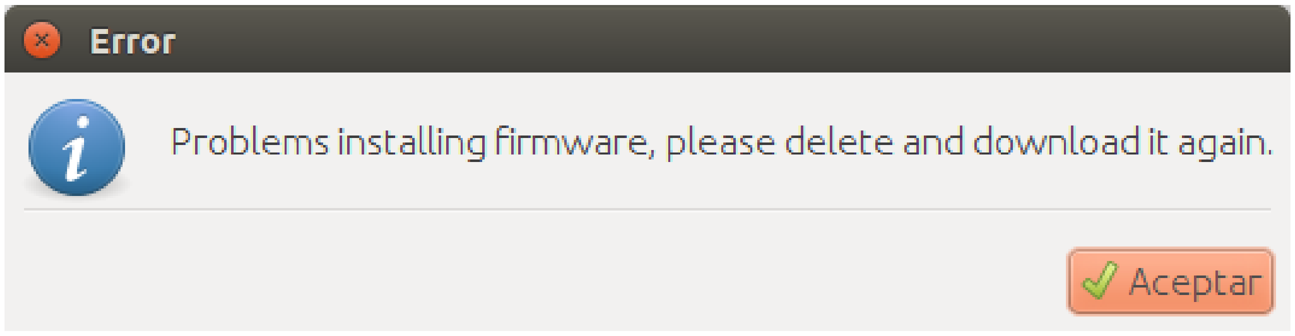
After few seconds, a small window should appear indicating that the program was uploaded successfully:


OK

Version 1044012670020820-2017-10-23-10-17-36 was installed in Plug&Sense! device

Figure: Smart Devices App successful message

On the other hand, the user might experience some issue. In that case, the following message will be displayed in the Smart Devices App. The most common errors are due to a bad USB port selection or a device powered-off state. So keep in mind the previous steps in order to use the Smart Devices App properly.


Error

Problems installing firmware, please delete and download it again.

Figure: Smart Devices App error message

2.15. Limitations

2.15.1. Sensor probe repetition

The PCS does not allow the user to repeat the same sensor probe in more than one socket. For instance, regarding the Smart Security model, the form does not permit to select a "luxes" sensor in both 'A' and 'C' sockets at the same time. This limitation applies to the following Plug and Sense! models and sensors:

- Smart Cities PRO: BME280, luxes, ultrasound and all gases sensors.
- Smart Environment PRO: All gases sensors.
- Smart Security: PIR, hall effect, liquid level, liquid presence, BME280, luxes and ultrasound sensors.
- Smart Water: ORP sensor.
- Smart Water Ions Single/Double/PRO: All ions and pH sensors.

However, there are some exceptions. The Smart Agriculture and Smart Agriculture PRO models allow the user to select up to three "soil moisture" sensors in sockets 'B', 'C' and 'E'.

2.15.2. Memory issues

As the PCS form permits to enable and configure so many possibilities, there could be some memory issues if the user considers using all PCS features in the same device. This limitation is not due to the PCS, it is due to the Waspmote's microcontroller: its flash memory allows binaries of a certain size. So this limitation applies to any Waspmote or Plug & Sense! unit, not only to those Plug & Sense! devices programmed with the PCS.

This case will happen only for very rare and complex selections. So it is not regular to find this type of restriction. For instance, besides using a Plug and Sense! model and probes, the user desires to enable the GPS module, an industrial protocol module and 4G HTTPS protocol at the same time. This combination would require a lot of memory allocation and will lead to memory issues.

3. How data frames are generated

3.1. Data frame types used by the PCS

As you know, the programs generated by the PCS consist on 3 basic steps:

- Read sensor data
- Send sensor data
- Sleep mode

The Waspmote Frame was designed in order to create sensor data frames with a specific format. This data protocol is supported by Meshlium (Meshlium can decode these data frames), so this is the format used by the PCS in order to transmit data. For more information, refer to the Data Frame Guide.

The PCS creates "binary frames" because this type of frame permits to add more sensor data within the same frame buffer. Each sensor value defines a sensor field which is added to the frame structure. All sensor fields can be found in the Data Frame Guide.

HEADER							PAYLOAD			
<=>	Frame Type	Num of bytes	Serial ID	Waspmote ID	#	Sequence	Sensor_1	Sensor_2	...	Sensor_n

Figure: Binary frame format

Besides, a special frame format was designed in order to send sensor data via low bit-rate protocols with short payload size. This frame type is called "tiny frame". The user must keep in mind that this protocol is not integrated into Meshlium (in fact, this frame type is mainly designed for constrained radios like Sigfox or LoRaWAN, and when operating with these protocols the receiver is not Meshlium but a Sigfox or LoRaWAN base station). In this case, the Programming Cloud Service chops the whole "binary" frame payload into a number of "tiny frames" as explained in the Data Frame Guide.

HEADER		PAYLOAD			
Sequence	Lenght	Sensor_1	Sensor_2	...	Sensor_n

Figure: Tiny frame format for LoRaWAN and Sigfox

Also, payload encryption feature (AES-256) can only be applied to radios using the binary frame format. If this feature is enabled, encrypted binary frames will be used. The format of the encrypted frames is explained in detail in the Data Frame Guide. In case of enabling the payload encryption feature, the user must define whether "Device to Cloud" or "Device to Gateway" is used. The only difference between both encryption options is the frame type identifier. Also, Meshlium will manage the frames differently. On one hand, the "Device to Cloud" packet will be stored in the Meshlium's database as it is received. On the other hand, the "Device to Gateway" packet will be decrypted and then stored in the Meshlium's database as any other unencrypted frame.

HEADER				PAYLOAD
<=>	Frame Type	Num of bytes	Serial ID	Encrypted Payload

Figure: Encrypted frame format

To sum up, depending on the radio protocol a specific frame format is used:

Radio	Frame type	Payload encryption (AES-256) feature?
XBee-PRO 802.15.4	Binary	Yes (only in PRO and Elite licenses)
XBee-PRO 900HP	Binary	Yes (only in PRO and Elite licenses)
XBee 868LP	Binary	Yes (only in PRO and Elite licenses)
4G	Binary	Yes (only in PRO and Elite licenses)
WiFi	Binary	Yes (only in PRO and Elite licenses)
Sigfox	Tiny	No
LoRaWAN	Tiny	No

3.2. Sensor fields definition

Normally, each sensor data consists on:

- Sensor identifier: this is the first byte to identify the sensor. For instance, the temperature sensor.
- Sensor value: the rest of the sensor data (one or more bytes) are dedicated to define the value of the sensor data.

Sensor data	
Sensor ID	Sensor value (one or more fields)

Figure: Sensor data structure

In the Data Frame Guide you can find the explanation for the binary frames payload. There are different sensor data types depending on the meaning of the data which integrates the sensor value. In the mentioned guide you will find 3 binary sensor data types:

- Simple data: sensor ID + one sensor field. For instance, the temperature sensor value.
- Complex data: sensor ID + several sensor fields. For instance, the GPS (latitude and longitude).
- String data: sensor ID + string length + string message.

For more information, refer to the Data Frame Guide.

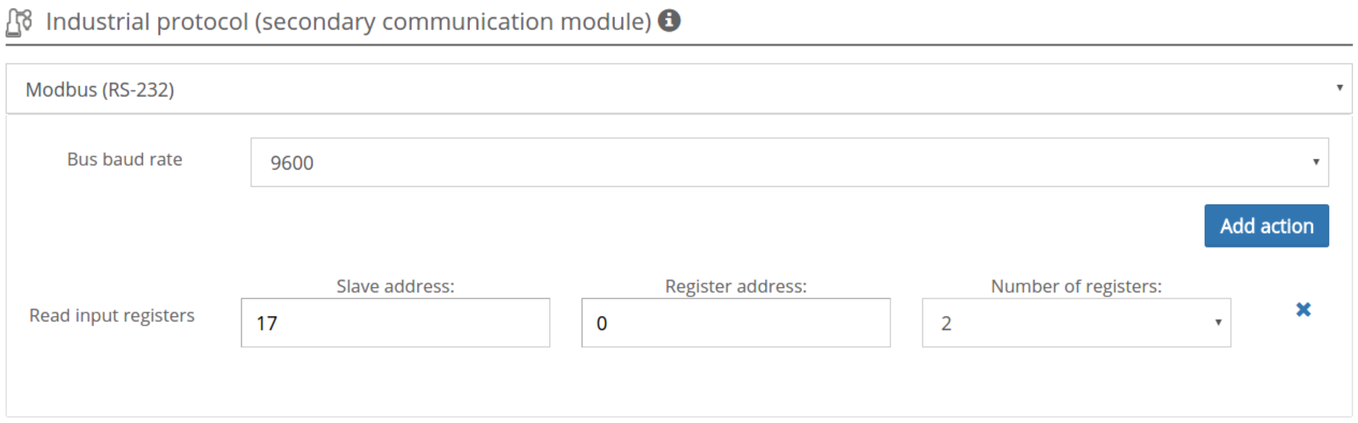
3.2.1. Modbus sensor data

The Modbus sensor data is a special case. The PCS features the "read input register" Modbus function. Each reading defines a new SENSOR_MODBUS_INPUT_REGS sensor data. This is a complex sensor data of three different uint16_t fields. The first field is divided into 2 bytes for the device address and the register address respectively. As the PCS can read up to 2 registers per action, the second field will be the first register and the third field will be the second register.

Example:

- Industrial protocol: Modbus (over RS-232)
- Baudrate: 9600
- Device address: 0x11 (Decimal 17)
- Register address: 0x00
- Number of registers to read: 2

In the form you should enter the next settings:


Industrial protocol (secondary communication module)

Modbus (RS-232)

Bus baud rate	9600			
	Slave address:	Register address:	Number of registers:	
Read input registers	17	0	2	✖

Figure: Industrial protocol settings

Imagine the first register read from the RS-232 module is 0x0192 and the second register is 0x0115. In that case, the final sensor data inserted within the payload would be:

Sensor data						
Sensor ID	**Sensor field 1**		**Sensor field 2**		**Sensor field 3**	
Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**
0xB7	0x11	0x00	0x92	0x11	0x15	0x01

Figure: Sensor data structure

4. How to get the CA certificate for HTTPS connections

Note: *The "Payload encryption" and "HTTPS" features are only supported by PRO and Elite license versions*

4.1. SSL / TLS basis

During the SSL/TLS negotiation process, the server identifies itself to the client by sending the server certificate. The server certificate's main purpose is to allow the client to determine that the server is indeed the server it claims to be.

The certificate authority (CA) is an entity that issues digital certificates. A digital certificate certifies the ownership of a public key by the named subject of the certificate. So, the CA assures you that no one possesses a certificate for a domain which is not their own. This allows clients to rely upon signatures made about the private key that corresponds to the certified public key. In this model of trust relationships, a CA is a third-party trusted both by the client and the server.

To fulfill its purpose, the server certificate contains the server's ID information (name, address, description, etc.) and its public key. It also contains a digital signature, signed by the CA, which authenticates this information. The client must trust the CA in order to accept its signature on a certificate. Furthermore, the trust relationship between the client and the CA must be established prior to the communication session. Usually, a client software (for example, Internet browsers as Google Chrome) include a set of trusted CA certificates. This makes sense, as many users need to trust their client software. Therefore, once a trusted CA's certificate is stored on the client, it will accept certificates signed by that CA from the SSL/TLS server it connects to.

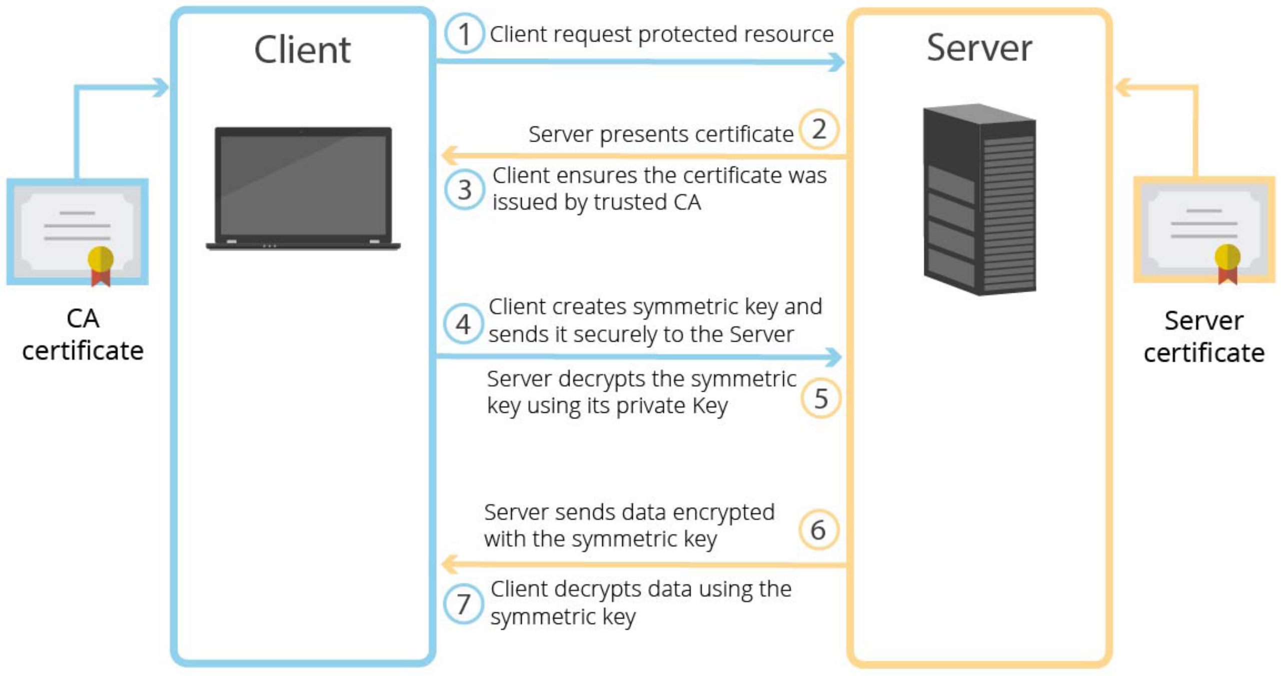

Client
Server
CA certificate
Server certificate
1 Client request protected resource
2 Server presents certificate
3 Client ensures the certificate was issued by trusted CA
4 Client creates symmetric key and sends it securely to the Server
5 Server decrypts the symmetric key using its private Key
6 Server sends data encrypted with the symmetric key
7 Client decrypts data using the symmetric key

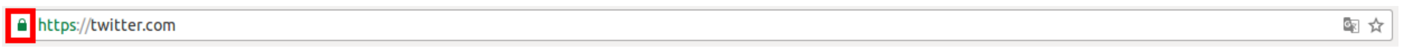
Figure: SSL/TLS negotiation process between Client and Server

Before using secure connections with WiFi or 4G radios, you must make sure the CA certificate is correctly installed on the radio.

4.2. Certificate management

In you set up your own system and create your own certificates, you should already have the CA certificate. If you are using an external third-party server, you must follow the next steps in order to obtain the CA certificate from your web browser application (i.e. Google Chrome). Imagine we want to access to https://twitter.com and we need to install the corresponding CA certificate:

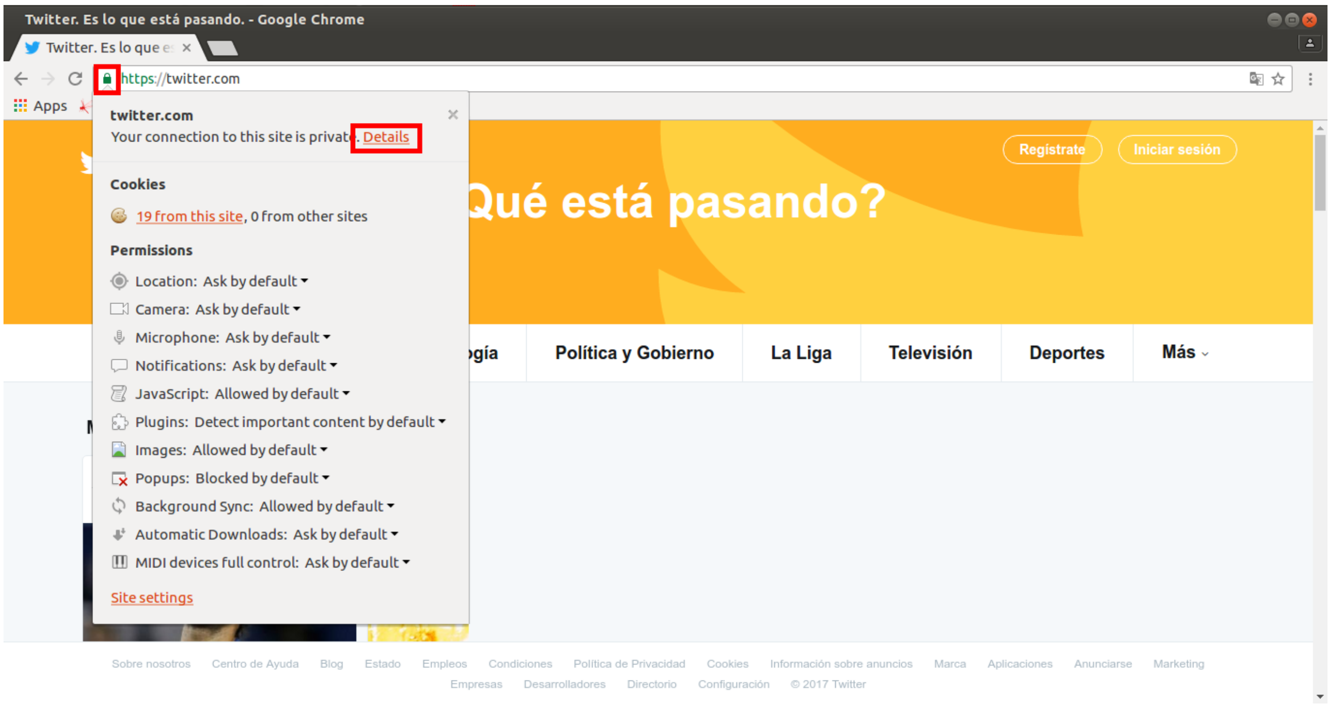
Step 1: Use the browser to access to the website.



https://twitter.com

Figure: Access to the website

Step 2: Get the details of the Server's certificate.



Twitter. Es lo que está pasando. - Google Chrome

https://twitter.com

twitter.com
Your connection to this site is private. Details

Cookies
19 from this site, 0 from other sites

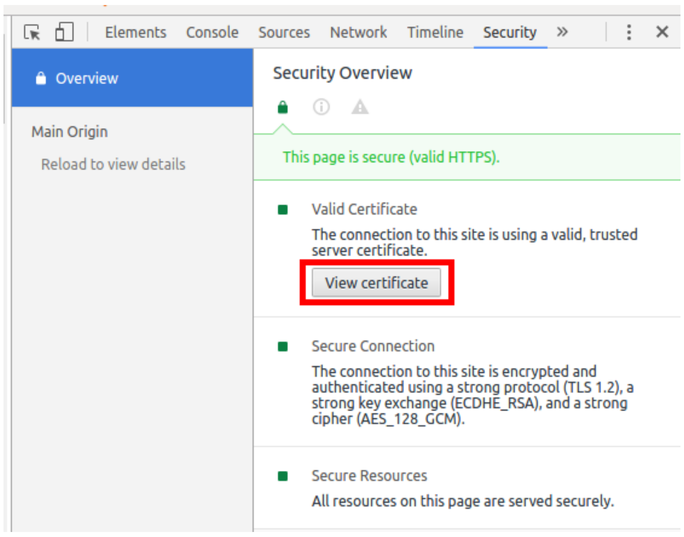
Permissions
Location: Ask by default
Camera: Ask by default
Microphone: Ask by default
Notifications: Ask by default
JavaScript: Allowed by default
Plugins: Detect important content by default
Images: Allowed by default
Popups: Blocked by default
Background Sync: Allowed by default
Automatic Downloads: Ask by default
MIDI devices full control: Ask by default

Site settings

Apps
Regístrate
Iniciar sesión
Qué está pasando?
Política y Gobierno
La Liga
Televisión
Deportes
Más
Sobre nosotros Centro de Ayuda Blog Estado Empleos Condiciones Política de Privacidad Cookies Información sobre anuncios Marca Aplicaciones Anunciarse Marketing
Empresas Desarrolladores Directorio Configuración © 2017 Twitter

Figure: Get the details of the Server's certificate

Step 3: View Server's certificate.



Security Overview

This page is secure (valid HTTPS).

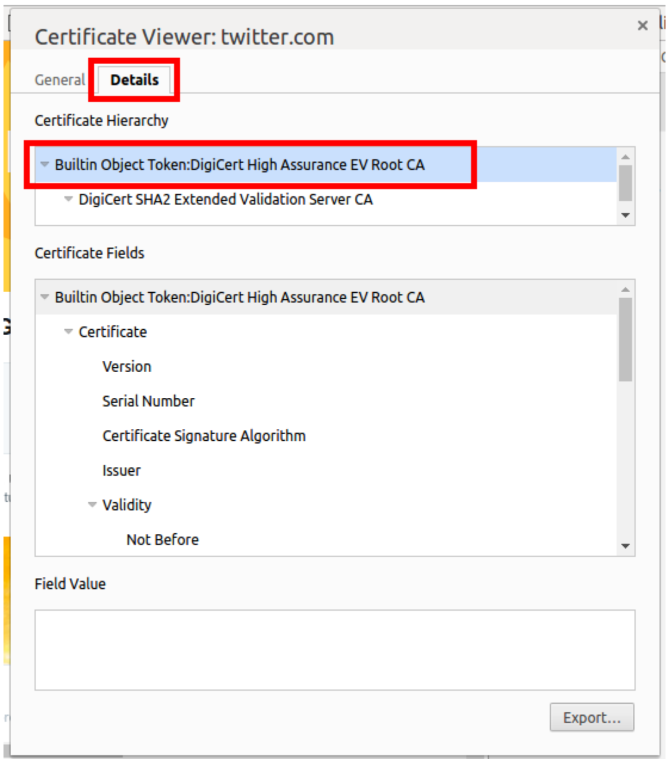
Valid Certificate
The connection to this site is using a valid, trusted server certificate.
View certificate

Secure Connection
The connection to this site is encrypted and authenticated using a strong protocol (TLS 1.2), a strong key exchange (ECDHE_RSA), and a strong cipher (AES_128_GCM).

Secure Resources
All resources on this page are served securely.

Figure: View Server's certificate

Step 4: Extract information about the Certificate Authority.


Certificate Viewer: twitter.com

General | Details

Certificate Hierarchy

- Builtin Object Token:DigiCert High Assurance EV Root CA
  - DigiCert SHA2 Extended Validation Server CA

Certificate Fields

- Builtin Object Token:DigiCert High Assurance EV Root CA
  - Certificate
    - Version
    - Serial Number
    - Certificate Signature Algorithm
    - Issuer
    - Validity
      - Not Before

Field Value

Figure: Extract information about the Certificate Authority

Step 5: Go to web browser settings to look for the installed CA certificate.

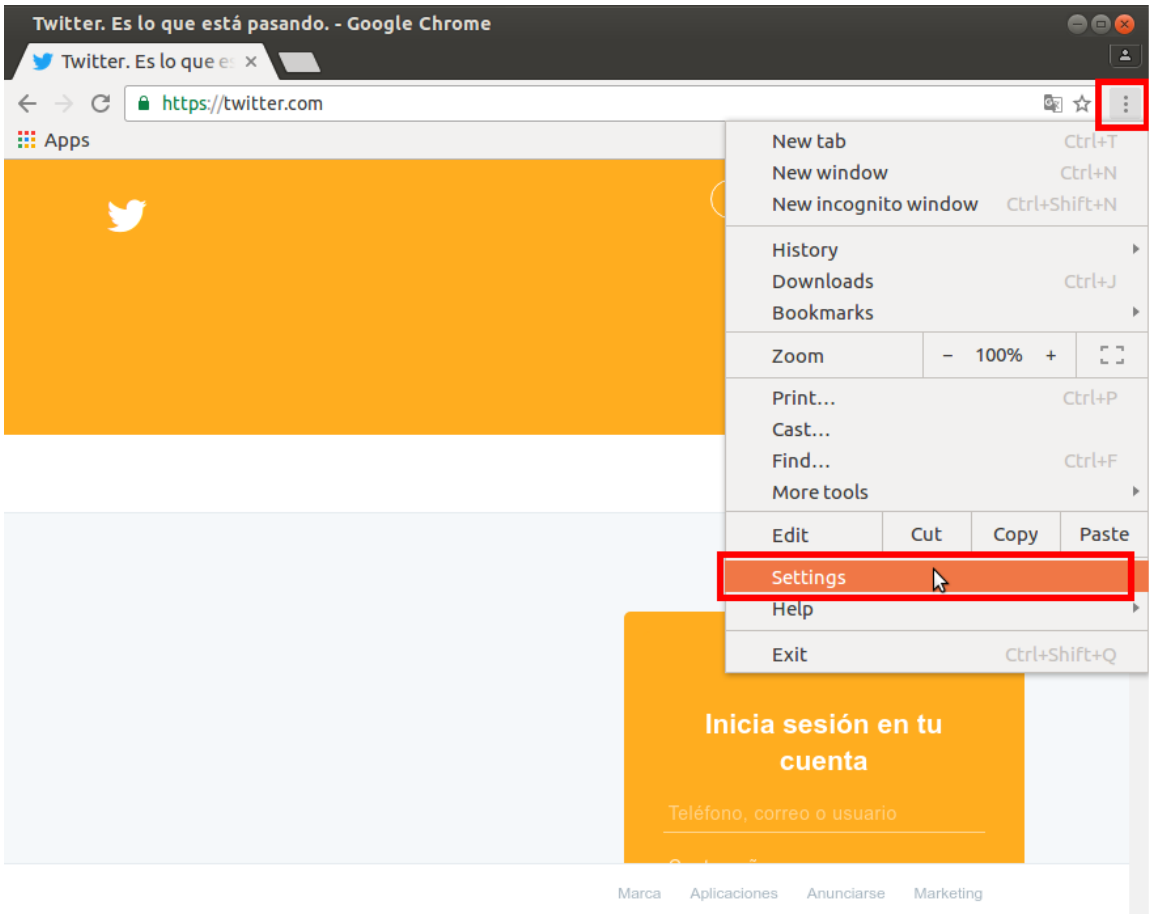


Figure: Web browser settings

Step 6: Go to “HTTPS/SSL” settings and enter into “Manage certificates” section.

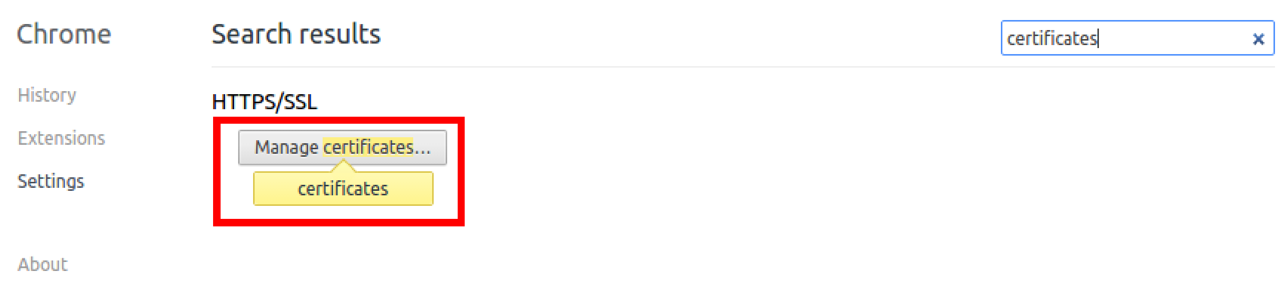
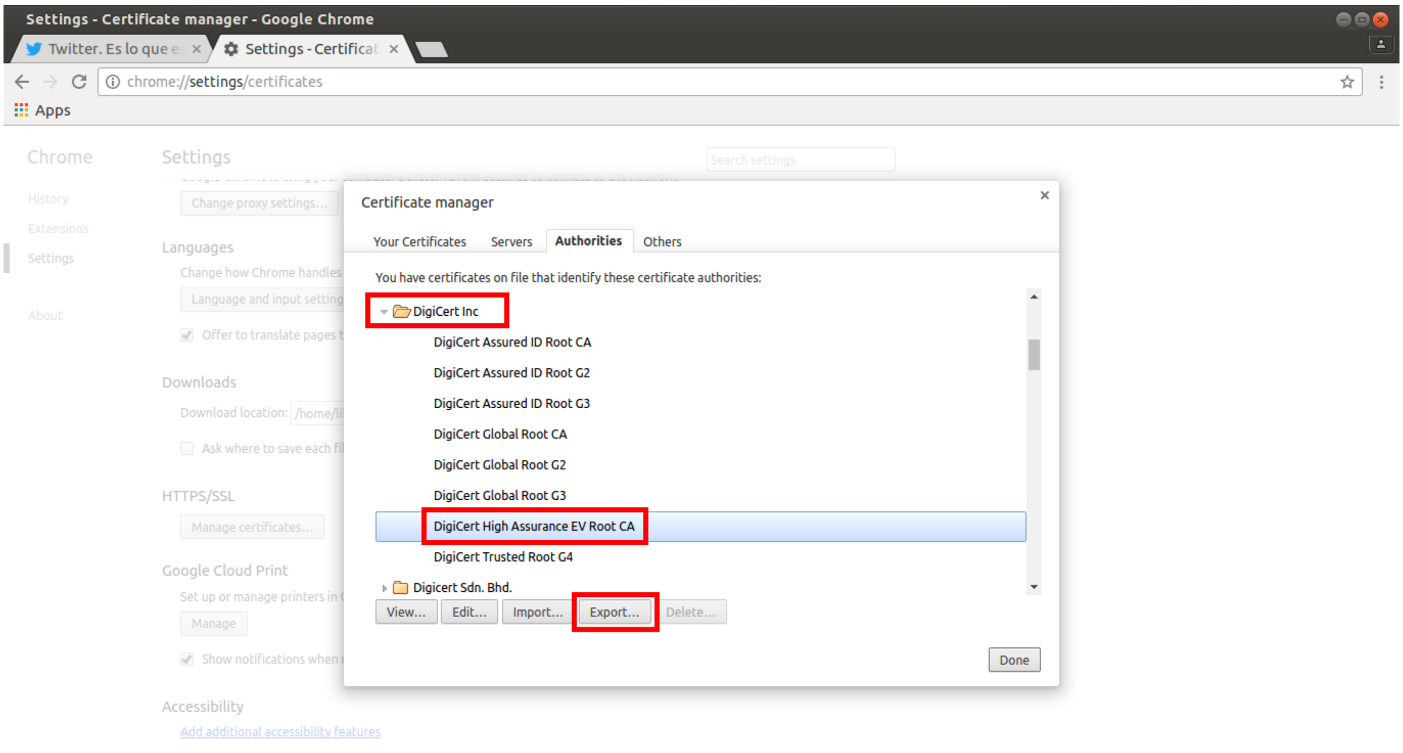


Figure: HTTPS/SSL settings

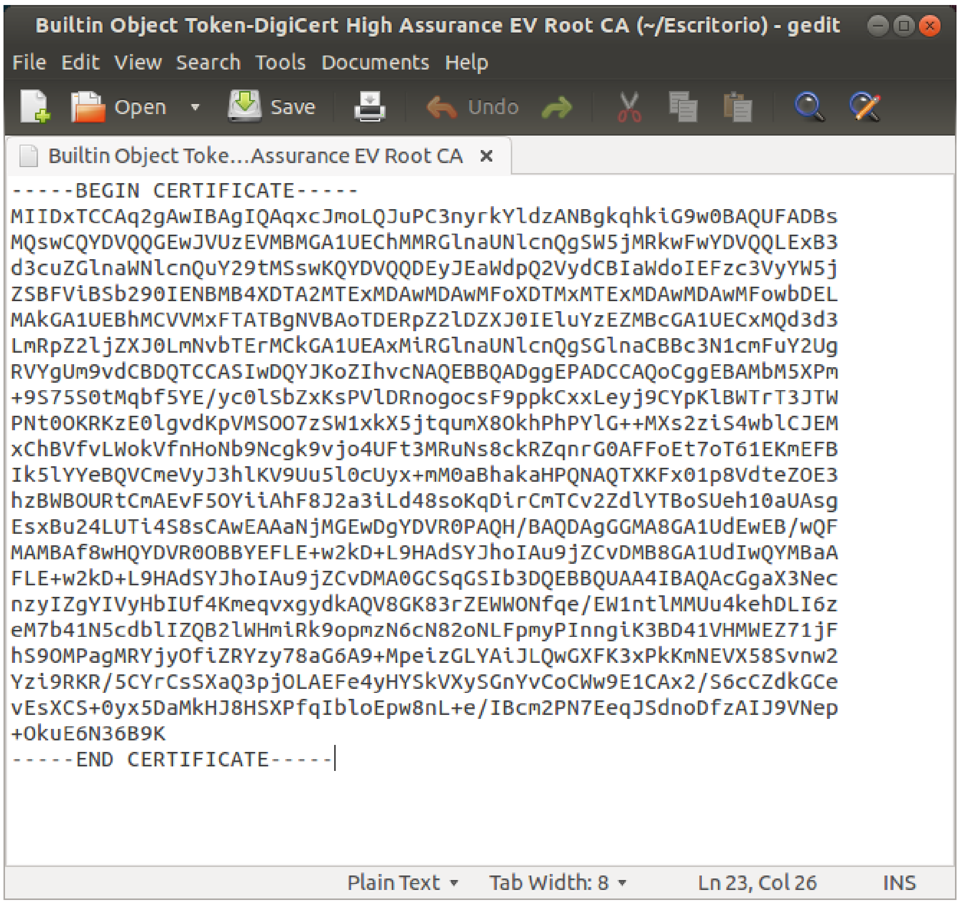
Step 7: Export the correct certificate from the list of Authorities.



You have certificates on file that identify these certificate authorities:

Figure: Export CA certificate

Step 8: Now you can open the certificate with a text editor application and copy it into the PCS form.


Builtin Object Token-DigiCert High Assurance EV Root CA (~/Escritorio) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
Builtin Object Toke…Assurance EV Root CA
Plain Text
Tab Width: 8
Ln 23, Col 26
INS

```
-----BEGIN CERTIFICATE-----
MIIDxTCCAq2gAwIBAgIQAqxcJmoLQJuPC3nyrkYldzANBgkqhkiG9w0BAQUFADBs
MQswCQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMRkwFwYDVQQLExB3
d3cuZGlnaWNlcnQuY29tMSswKQYDVQQDEyJEaWdpQ2VydCBIaWdoIEFzc3VyYW5j
ZSBFViBSb290IENBMB4XDTA2MTExMDAwMDAwMFoXDTMxMTExMDAwMDAwMFowbDEL
MAkGA1UEBhMCVVMxFTATBgNVBAoTDERpZ2lDZXJ0IEluYzEZMBcGA1UECxMQd3d3
LmRpZ2ljZXJ0LmNvbTErMCkGA1UEAxMiRGlnaUNlcnQgSGlnaCBBc3N1cmFuY2Ug
RVYgUm9vdCBDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMbM5XPm
+9S75S0tMqbf5YE/yc0lSbZxKsPVlDRnogocsF9ppkCxxLeyj9CYpKlBWTrT3JTW
PNt0OKRKzE0lgvdKpVMSOO7zSW1xkX5jtqumX8OkhPhPYlG++MXs2ziS4wblCJEM
xChBVfvLWokVfnHoNb9Ncgk9vjo4UFt3MRuNs8ckRZqnrG0AFFoEt7oT61EKmEFB
Ik5lYYeBQVCmeVyJ3hlKV9Uu5l0cUyx+mM0aBhakaHPQNAQTXKFx01p8VdteZOE3
hzBWBOURtCmAEvF5OYiiAhF8J2a3iLd48soKqDirCmTCv2ZdlYTBoSUeh10aUAsg
EsxBu24LUTi4S8sCAwEAAaNjMGEwDgYDVR0PAQH/BAQDAgGGMA8GA1UdEwEB/wQF
MAMBAf8wHQYDVR0OBBYEFLE+w2kD+L9HAdSYJhoIAu9jZCvDMB8GA1UdIwQYMBaA
FLE+w2kD+L9HAdSYJhoIAu9jZCvDMA0GCSqGSIb3DQEBBQUAA4IBAQAcGgaX3Nec
nzyIZgYIVyHbIUf4KmeqvxgydkAQV8GK83rZEWWONfqe/EW1ntlMMUu4kehDLI6z
eM7b41N5cdblIZQB2lWHmiRk9opmzN6cN82oNLFpmyPInngiK3BD41VHMWEZ71jF
hS9OMPagMRYjyOfiZRYzy78aG6A9+MpeizGLYAiJLQwGXFK3xPkKmNEVX58Svnw2
Yzi9RKR/5CYrCsSXaQ3pjOLAEFe4yHYSkVXySGnYvCoCWw9E1CAx2/S6cCZdkGCe
vEsXCS+0yx5DaMkHJ8HSXPfqIbloEpw8nL+e/IBcm2PN7EeqJSdnoDfzAIJ9VNep
+OkuE6N36B9K
-----END CERTIFICATE-----
```

Figure: Copy CA certificate contents

5. Sending to the Cloud

5.1. Send data to Meshlium

Several communication modules in the Plug & Sense! line permit to send sensor data to Meshlium:

- XBee-PRO 802.15.4
- XBee-PRO 900HP
- XBee 868LP
- WiFi
- 4G

LoRaWAN and Sigfox protocols cannot send data directly to Meshlium as they send sensor frames to the customer's LoRaWAN Cloud and the Sigfox Cloud, respectively.

The XBee radios send data directly to Meshlium as there is another XBee radio inside Meshlium that allows reception.

Regarding the WiFi radios, they can connect to the Meshlium's Access Point (AP) and proceed with the HTTP request. However, it is also possible to send data to Meshlium via Ethernet or 4G interface if another AP different from the Meshlium's is used. The user only needs an external WiFi AP with Internet connectivity and the public Meshlium's IP address and port.

Regarding the 4G radios, they always use the mobile network operator infrastructure, so they might use the Meshlium's Ethernet or 4G interfaces in order to receive sensor data.

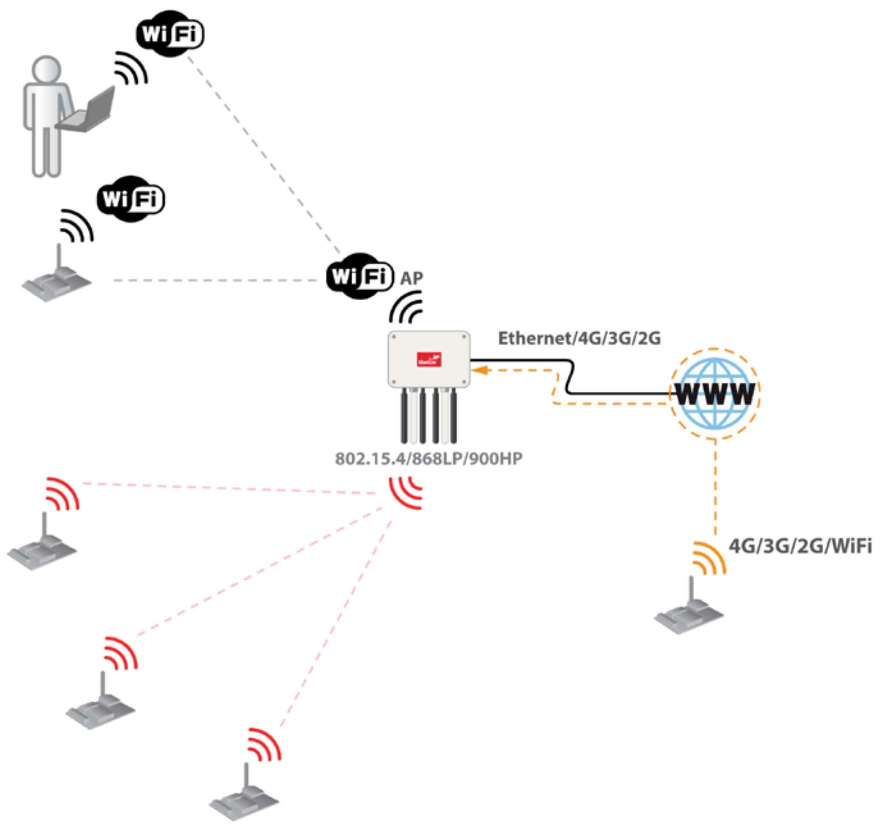


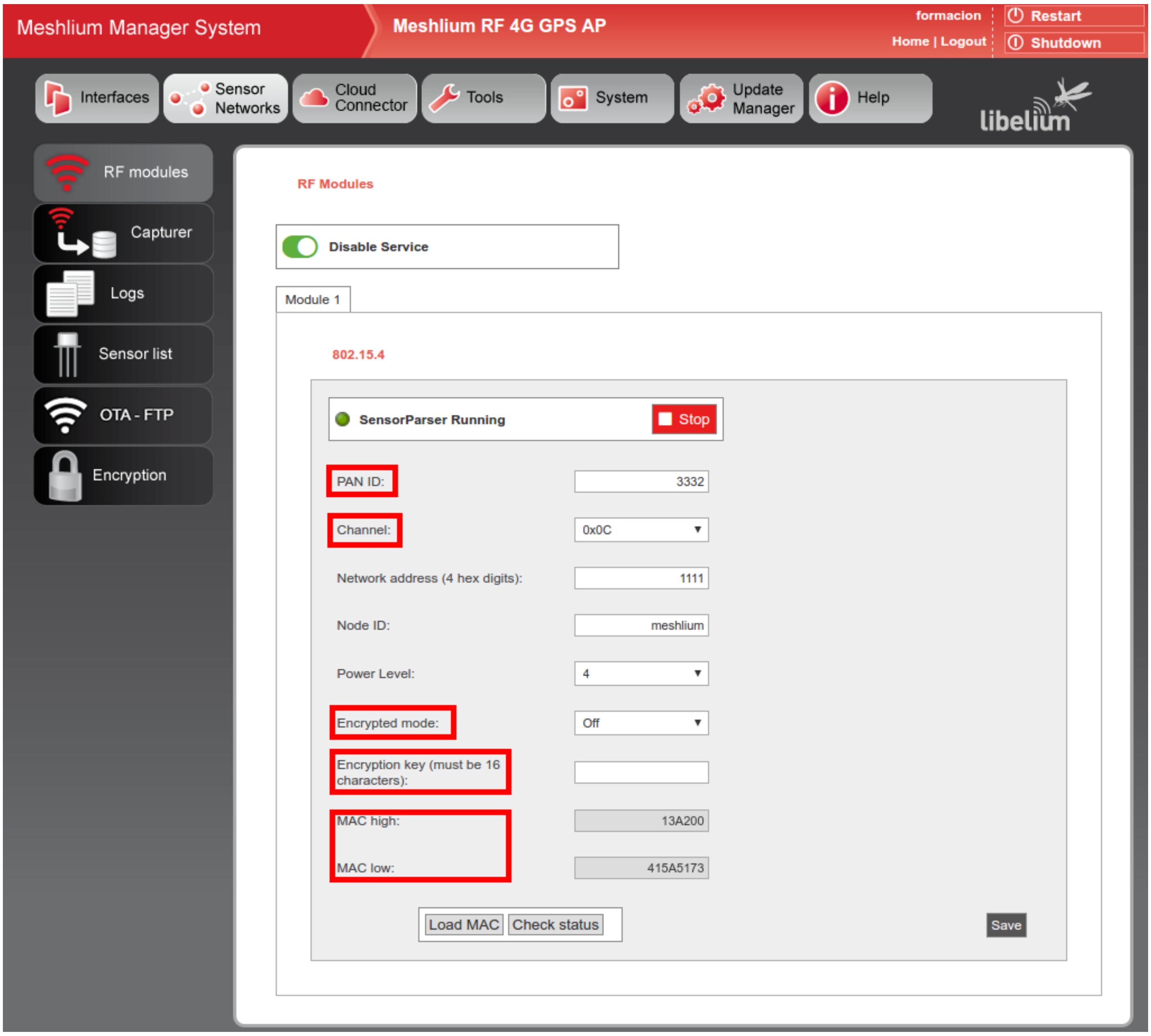
Figure: Send data to Meshlium

The PCS form allows the user to configure these types of communication radios to send data to Meshlium. Let's see all cases step by step.

5.1.1. XBee-PRO 802.15.4

This protocol needs to define the same parameters on all nodes in the network (this includes the Plug & Sense! device and the Meshlium device). Besides, the Meshlium's XBee radio MAC address must be provided in order to send data properly.

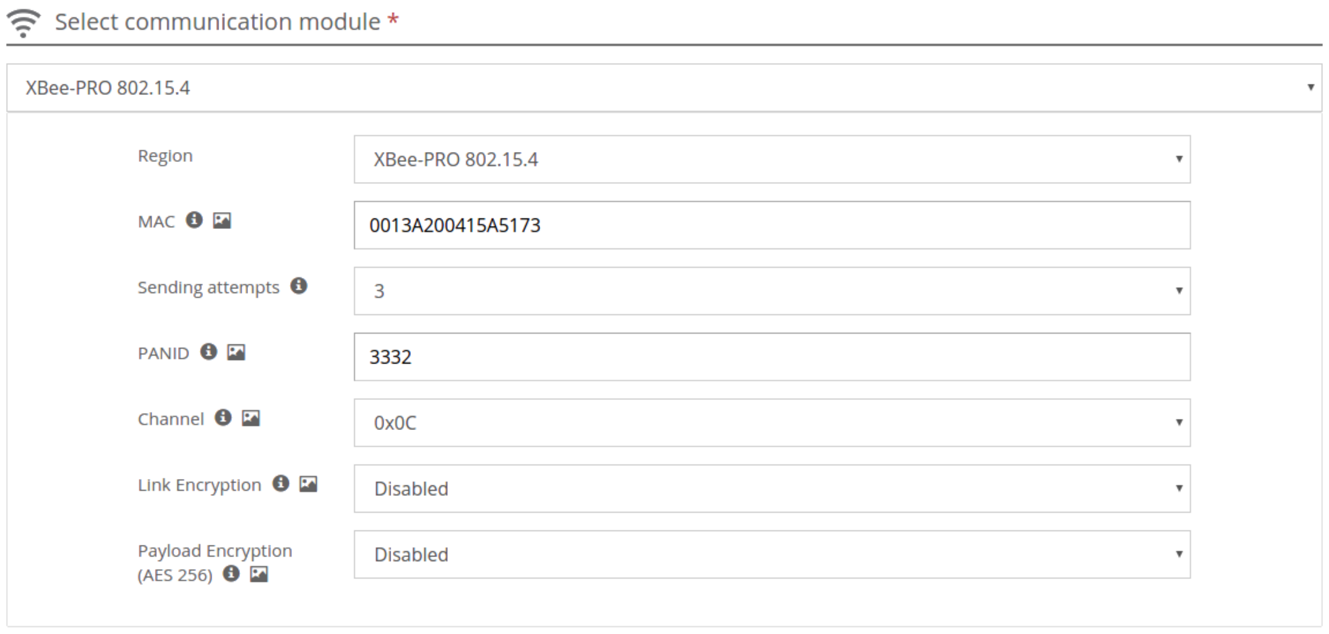
The Meshlium Manager System interface permits to configure the receiving XBee radio as follows:



PAN ID:	3332
Channel:	0x0C
Network address (4 hex digits):	1111
Node ID:	meshlium
Power Level:	4
Encrypted mode:	Off
Encryption key (must be 16 characters):	
MAC high:	13A200
MAC low:	415A5173

Figure: Meshlium Manager System (XBee-PRO 802.15.4)

Given the Meshlium configuration, the PCS form should be filled as follows:



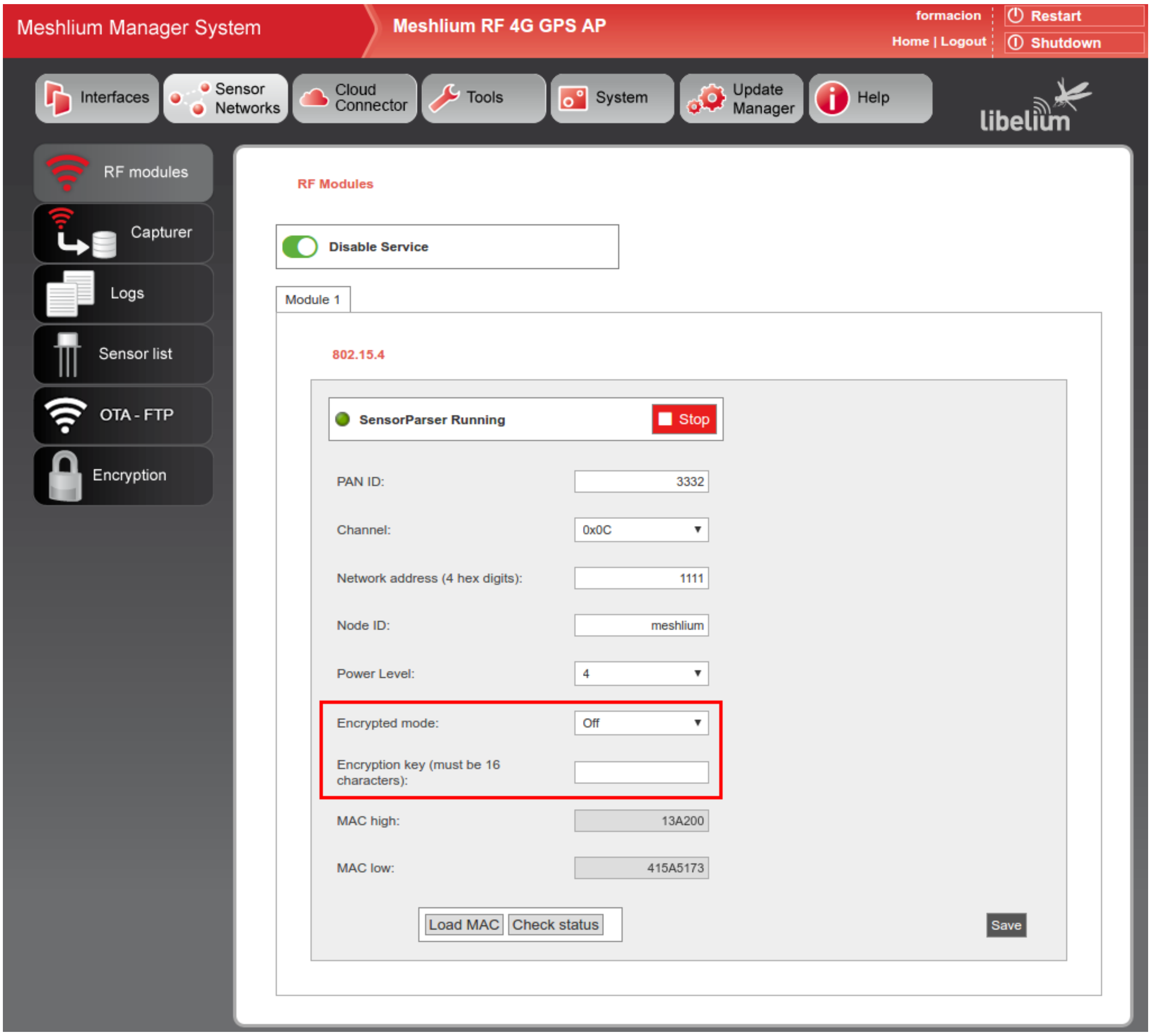
Select communication module *

XBee-PRO 802.15.4

Region	XBee-PRO 802.15.4
MAC	0013A200415A5173
Sending attempts	3
PANID	3332
Channel	0x0C
Link Encryption	Disabled
Payload Encryption (AES 256)	Disabled

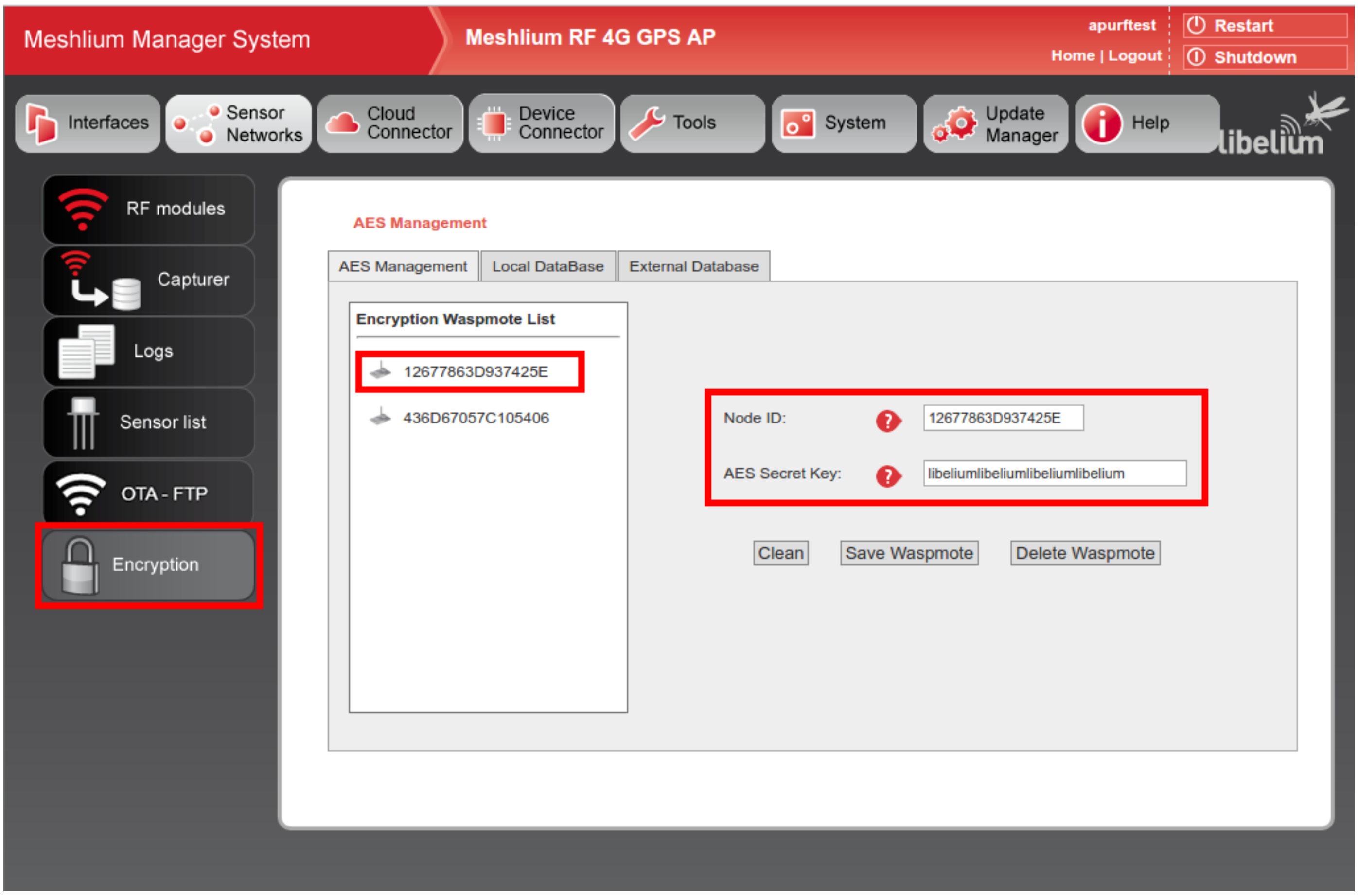
Figure: PCS form (XBee-PRO 802.15.4)

No encryption was configured in the previous example. If the user desires to enable the "Link encryption" or the "Payload encryption" layers, these features are also configured in both PCS form and Meshlium Manager System. Please refer to the "Programming Cloud Service configuration" section to see how you should configure the PCS form. On the other hand, the Meshlium Manager System provides the interfaces to enable both encryption layers:



PAN ID:	3332
Channel:	0x0C
Network address (4 hex digits):	1111
Node ID:	meshlium
Power Level:	4
Encrypted mode:	Off
Encryption key (must be 16 characters):	
MAC high:	13A200
MAC low:	415A5173

Figure: Meshlium Manager System "link encryption"



Encryption Waspmote List

12677863D937425E

436D67057C105406

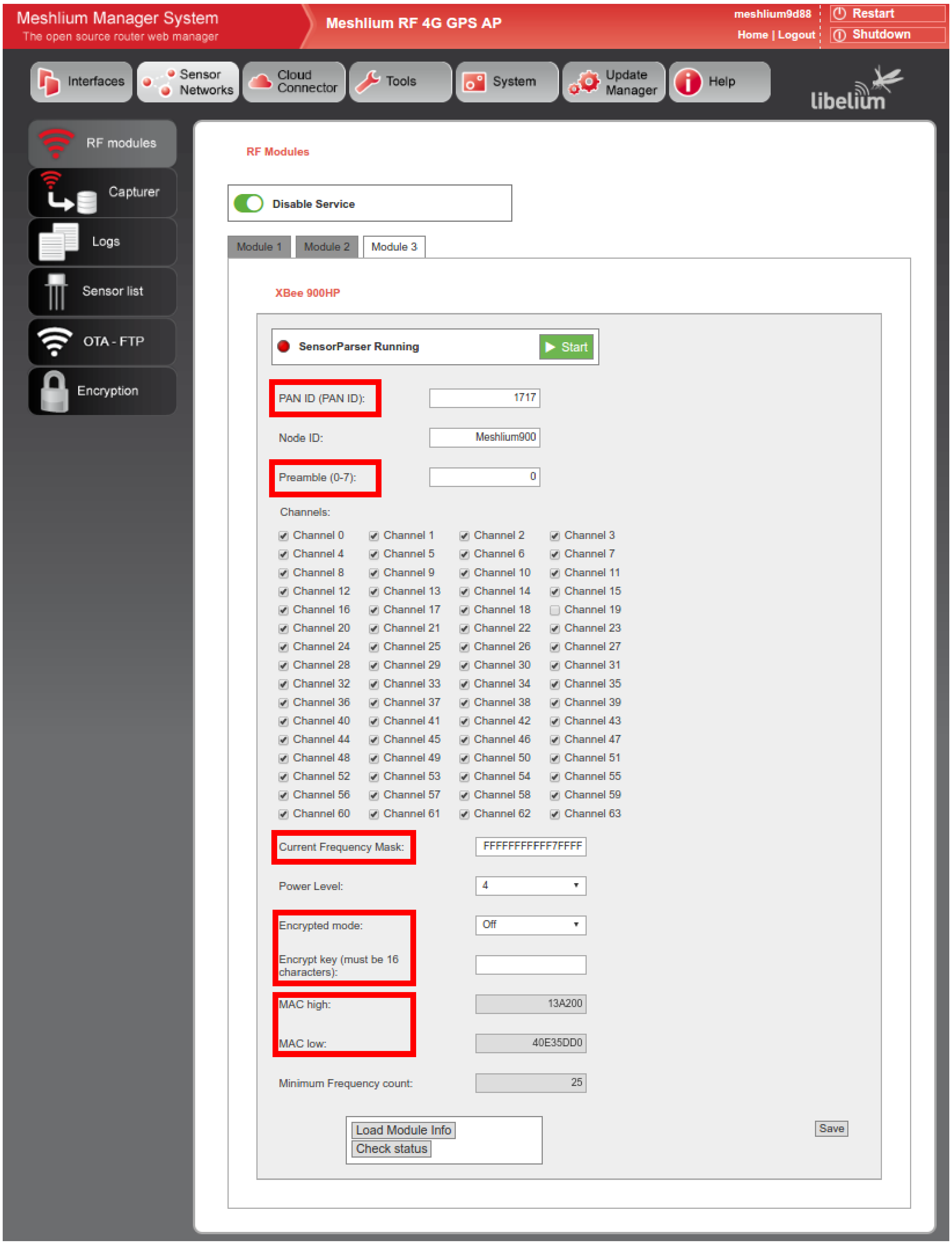
Node ID:	12677863D937425E
AES Secret Key:	libeliumlibeliumlibeliumlibelium

Figure: Meshlium Manager System "payload encryption"

5.1.2. XBee-PRO 900HP

This protocol needs to define the same parameters on all nodes in the network (this includes the Plug & Sense! device and the Meshlium device). Besides, the Meshlium's XBee radio MAC address must be provided in order to send data properly.

The Meshlium Manager System interface permits to configure the receiving XBee radio as follows:



PAN ID (PAN ID): 1717

Node ID: Meshlium900

Preamble (0-7): 0

Channels:

✓ Channel 0	✓ Channel 1	✓ Channel 2	✓ Channel 3
✓ Channel 4	✓ Channel 5	✓ Channel 6	✓ Channel 7
✓ Channel 8	✓ Channel 9	✓ Channel 10	✓ Channel 11
✓ Channel 12	✓ Channel 13	✓ Channel 14	✓ Channel 15
✓ Channel 16	✓ Channel 17	✓ Channel 18	☐ Channel 19
✓ Channel 20	✓ Channel 21	✓ Channel 22	✓ Channel 23
✓ Channel 24	✓ Channel 25	✓ Channel 26	✓ Channel 27
✓ Channel 28	✓ Channel 29	✓ Channel 30	✓ Channel 31
✓ Channel 32	✓ Channel 33	✓ Channel 34	✓ Channel 35
✓ Channel 36	✓ Channel 37	✓ Channel 38	✓ Channel 39
✓ Channel 40	✓ Channel 41	✓ Channel 42	✓ Channel 43
✓ Channel 44	✓ Channel 45	✓ Channel 46	✓ Channel 47
✓ Channel 48	✓ Channel 49	✓ Channel 50	✓ Channel 51
✓ Channel 52	✓ Channel 53	✓ Channel 54	✓ Channel 55
✓ Channel 56	✓ Channel 57	✓ Channel 58	✓ Channel 59
✓ Channel 60	✓ Channel 61	✓ Channel 62	✓ Channel 63

Current Frequency Mask: FFFFFFFFFFF7FFFF

Power Level: 4

Encrypted mode: Off

Encrypt key (must be 16 characters):

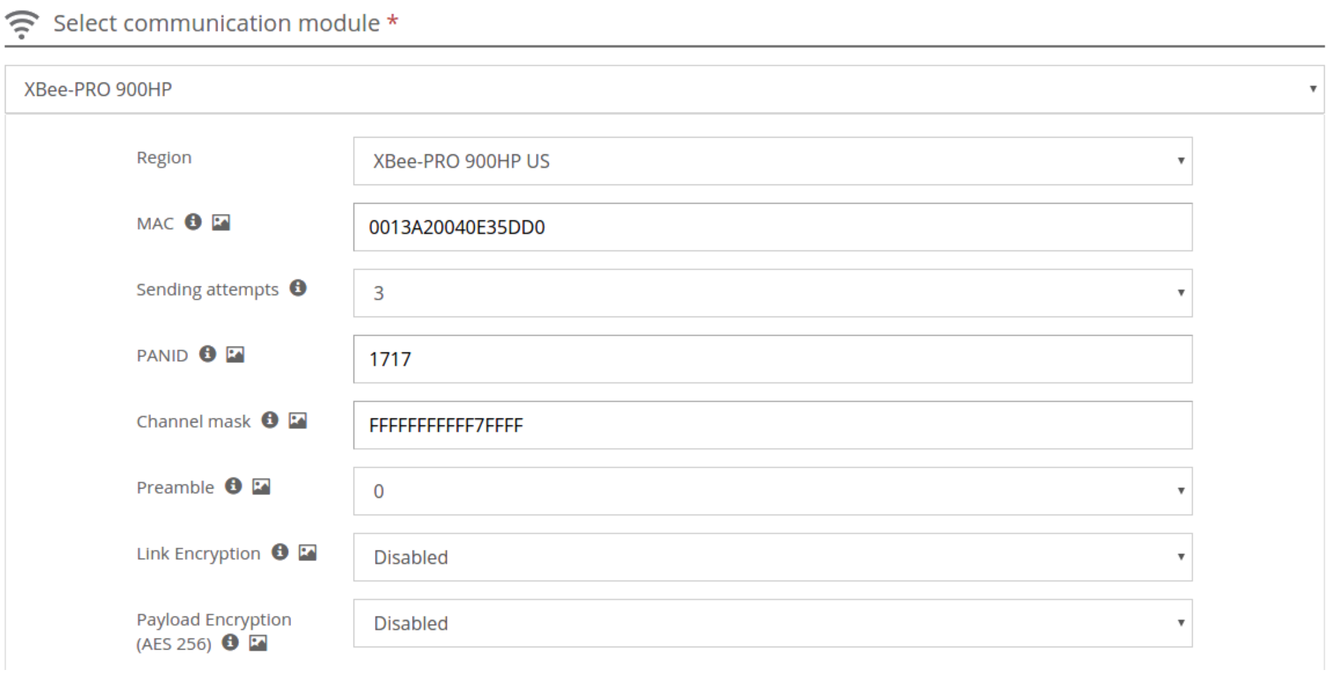
MAC high: 13A200

MAC low: 40E35DD0

Minimum Frequency count: 25

Figure: Meshlium Manager System (XBee-PRO 900HP)

Given the Meshlium configuration, the PCS form should be filled as follows:



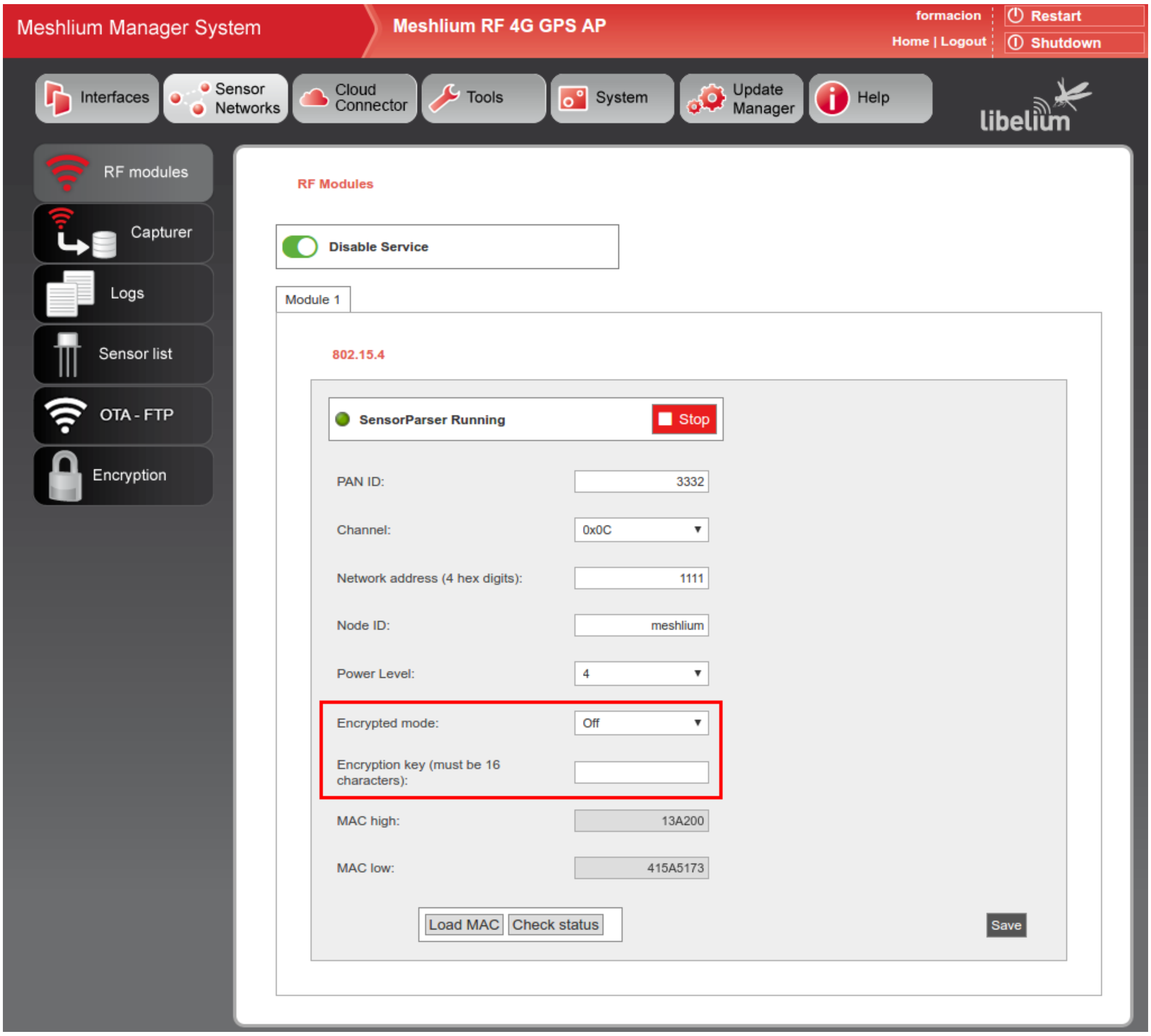
Select communication module *

XBee-PRO 900HP

Region	XBee-PRO 900HP US
MAC	0013A20040E35DD0
Sending attempts	3
PANID	1717
Channel mask	FFFFFFFFFFF7FFFF
Preamble	0
Link Encryption	Disabled
Payload Encryption (AES 256)	Disabled

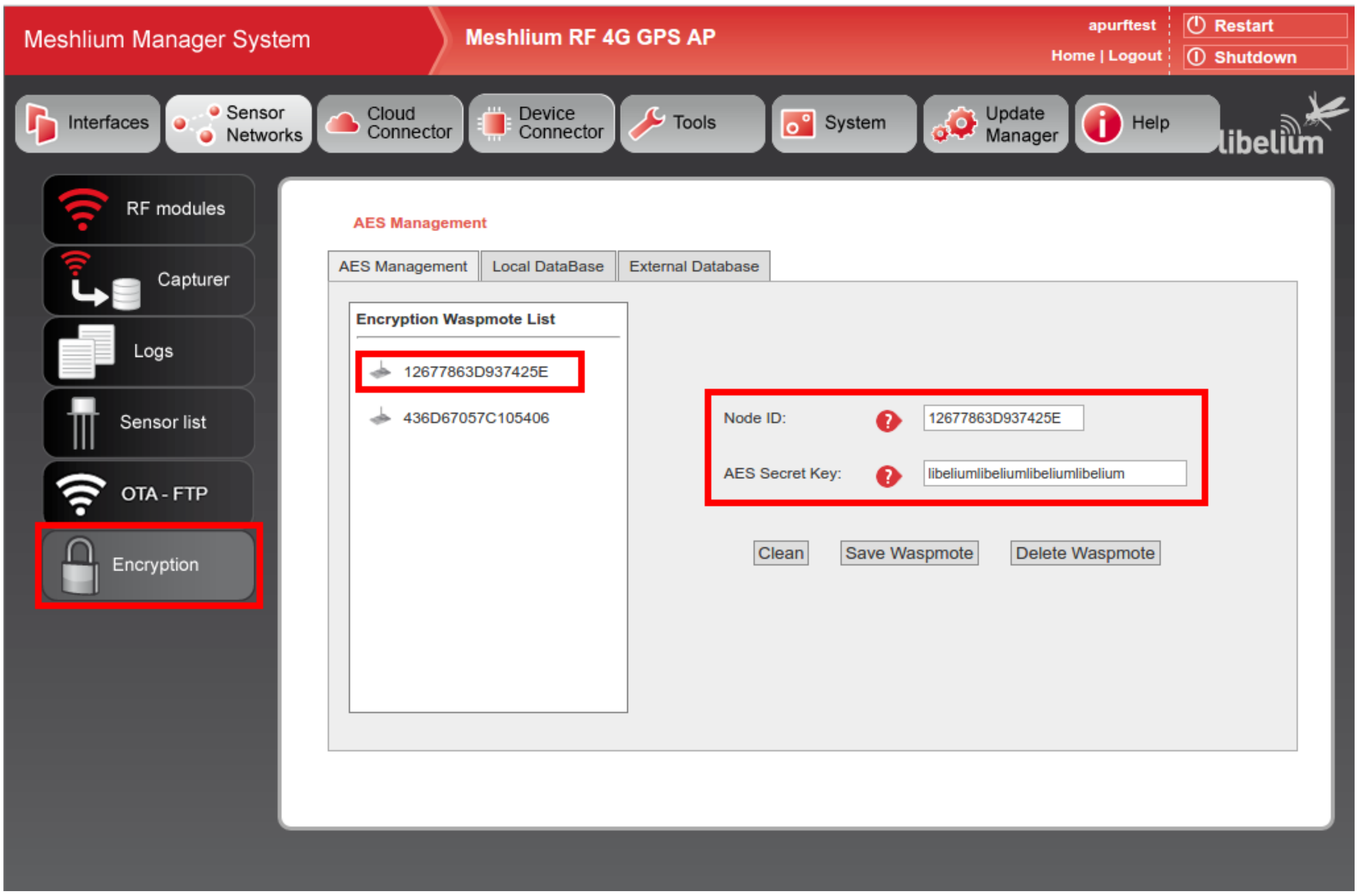
Figure: PCS form (XBee-PRO 900HP)

No encryption was configured in the previous example. If the user desires to enable the “Link encryption” or the “Payload encryption” layers, these features are also configured in both PCS form and Meshlium Manager System. Please refer to the “Programming Cloud Service configuration” section to see how you should configure the PCS form. On the other hand, the Meshlium Manager System provides the interfaces to enable both encryption layers:



PAN ID:	3332
Channel:	0x0C
Network address (4 hex digits):	1111
Node ID:	meshlium
Power Level:	4
Encrypted mode:	Off
Encryption key (must be 16 characters):	
MAC high:	13A200
MAC low:	415A5173

Figure: Meshlium Manager System “link encryption”



12677863D937425E
436D67057C105406

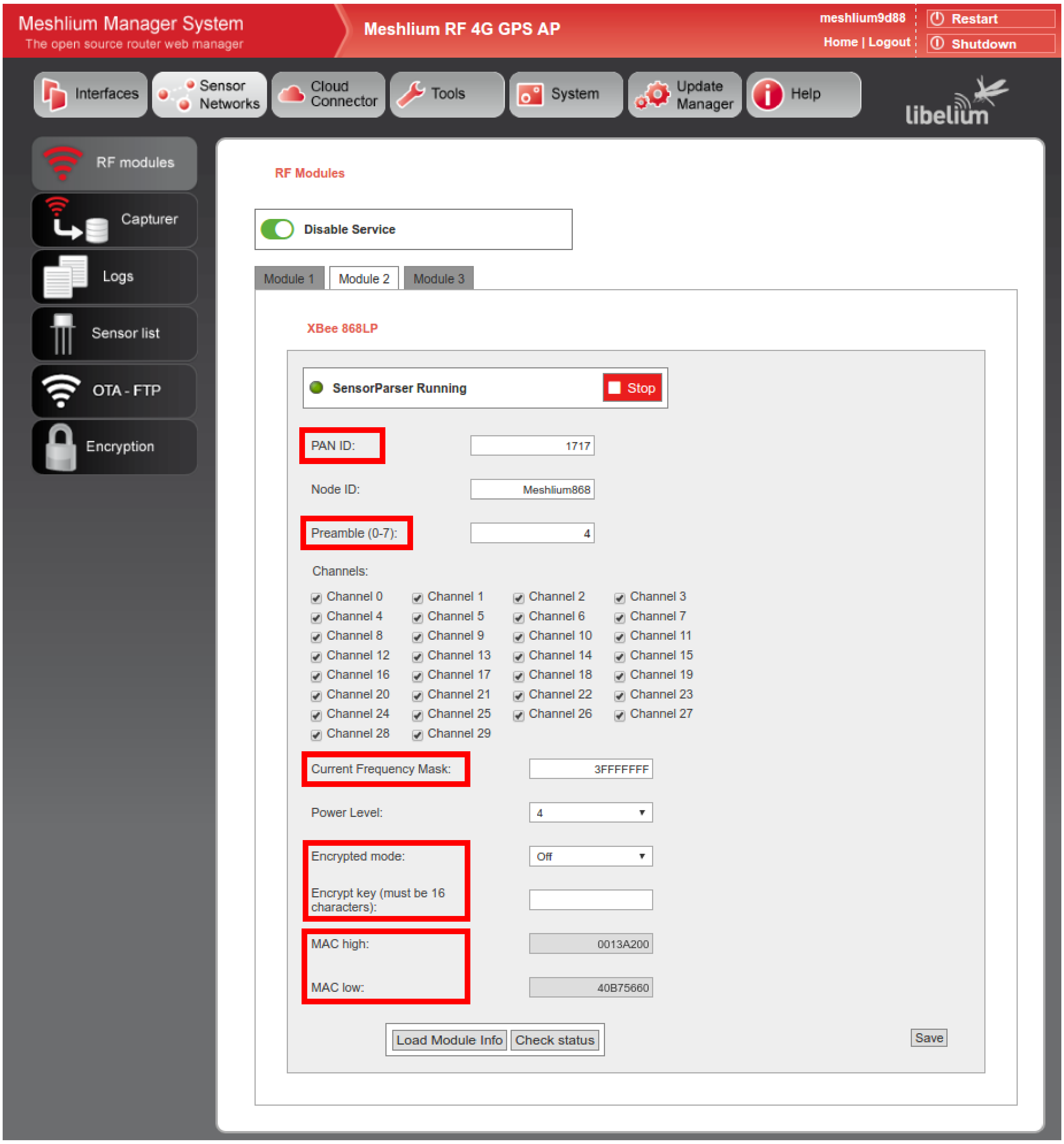
Node ID: 12677863D937425E
AES Secret Key: libeliumlibeliumlibeliumlibelium

Figure: Meshlium Manager System "payload encryption"

5.1.3. XBee 868LP

This protocol needs to define the same parameters on all nodes in the network (this includes the Plug & Sense! device and the Meshlium device). Besides, the Meshlium's XBee radio MAC address must be provided in order to send data properly.

The Meshlium Manager System interface permits to configure the receiving XBee radio as follows:



PAN ID:	1717
Node ID:	Meshlium868
Preamble (0-7):	4

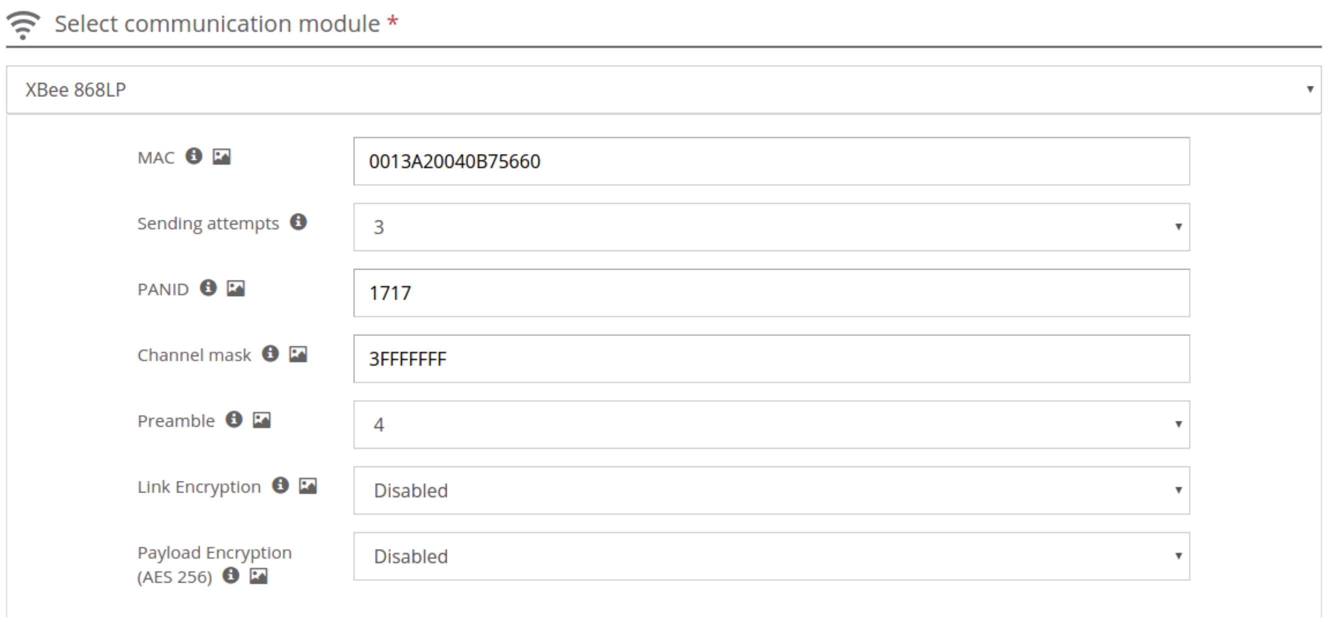
Channels:

✓ Channel 0	✓ Channel 1	✓ Channel 2	✓ Channel 3
✓ Channel 4	✓ Channel 5	✓ Channel 6	✓ Channel 7
✓ Channel 8	✓ Channel 9	✓ Channel 10	✓ Channel 11
✓ Channel 12	✓ Channel 13	✓ Channel 14	✓ Channel 15
✓ Channel 16	✓ Channel 17	✓ Channel 18	✓ Channel 19
✓ Channel 20	✓ Channel 21	✓ Channel 22	✓ Channel 23
✓ Channel 24	✓ Channel 25	✓ Channel 26	✓ Channel 27
✓ Channel 28	✓ Channel 29		

Current Frequency Mask:	3FFFFFFF
Power Level:	4
Encrypted mode:	Off
Encrypt key (must be 16 characters):	
MAC high:	0013A200
MAC low:	40B75660

Figure: Meshlium Manager System (XBee 868LP)

Given the Meshlium configuration, the PCS form should be filled as follows:



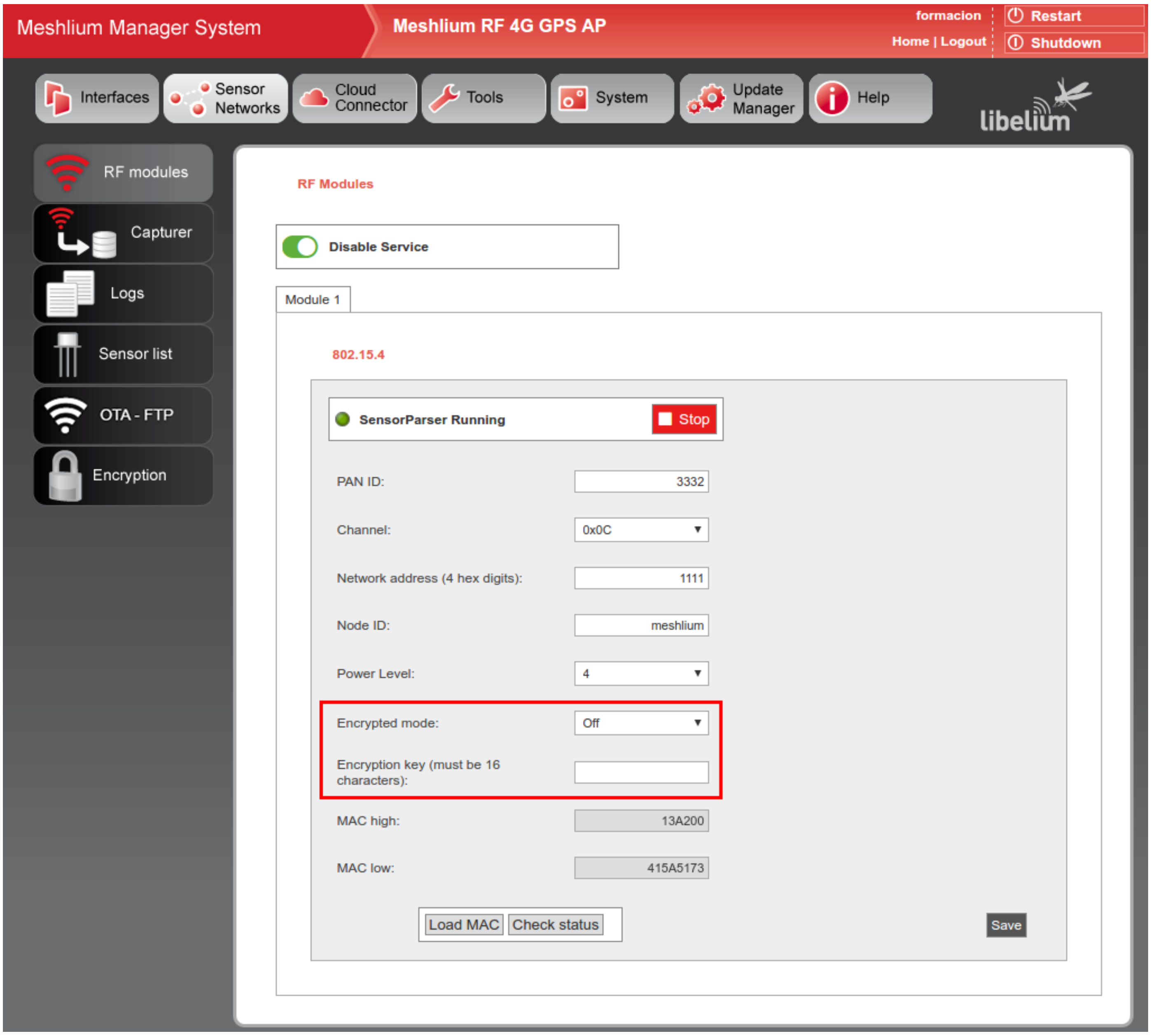
Select communication module *

XBee 868LP

MAC	0013A20040B75660
Sending attempts	3
PANID	1717
Channel mask	3FFFFFFF
Preamble	4
Link Encryption	Disabled
Payload Encryption (AES 256)	Disabled

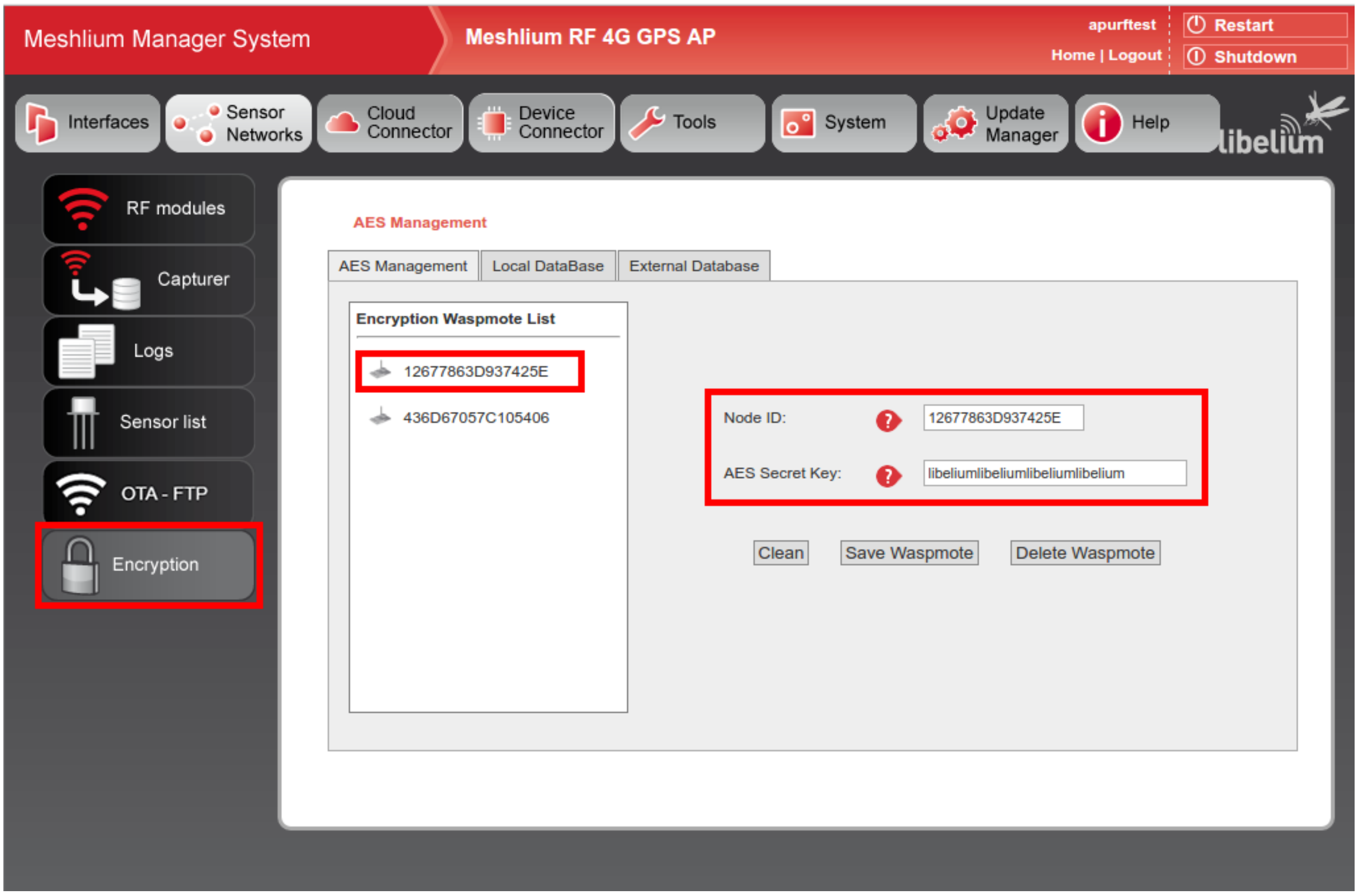
Figure: PCS form (XBee 868LP)

No encryption was configured in the previous example. If the user desires to enable the "Link encryption" or the "Payload encryption" layers, these features are also configured in both PCS form and Meshlium Manager System. Please refer to the "Programming Cloud Service configuration" section to see how you should configure the PCS form. On the other hand, the Meshlium Manager System provides the interfaces to enable both encryption layers:



PAN ID:	3332
Channel:	0x0C
Network address (4 hex digits):	1111
Node ID:	meshlium
Power Level:	4
Encrypted mode:	Off
Encryption key (must be 16 characters):	
MAC high:	13A200
MAC low:	415A5173

Figure: Meshlium Manager System "link encryption"



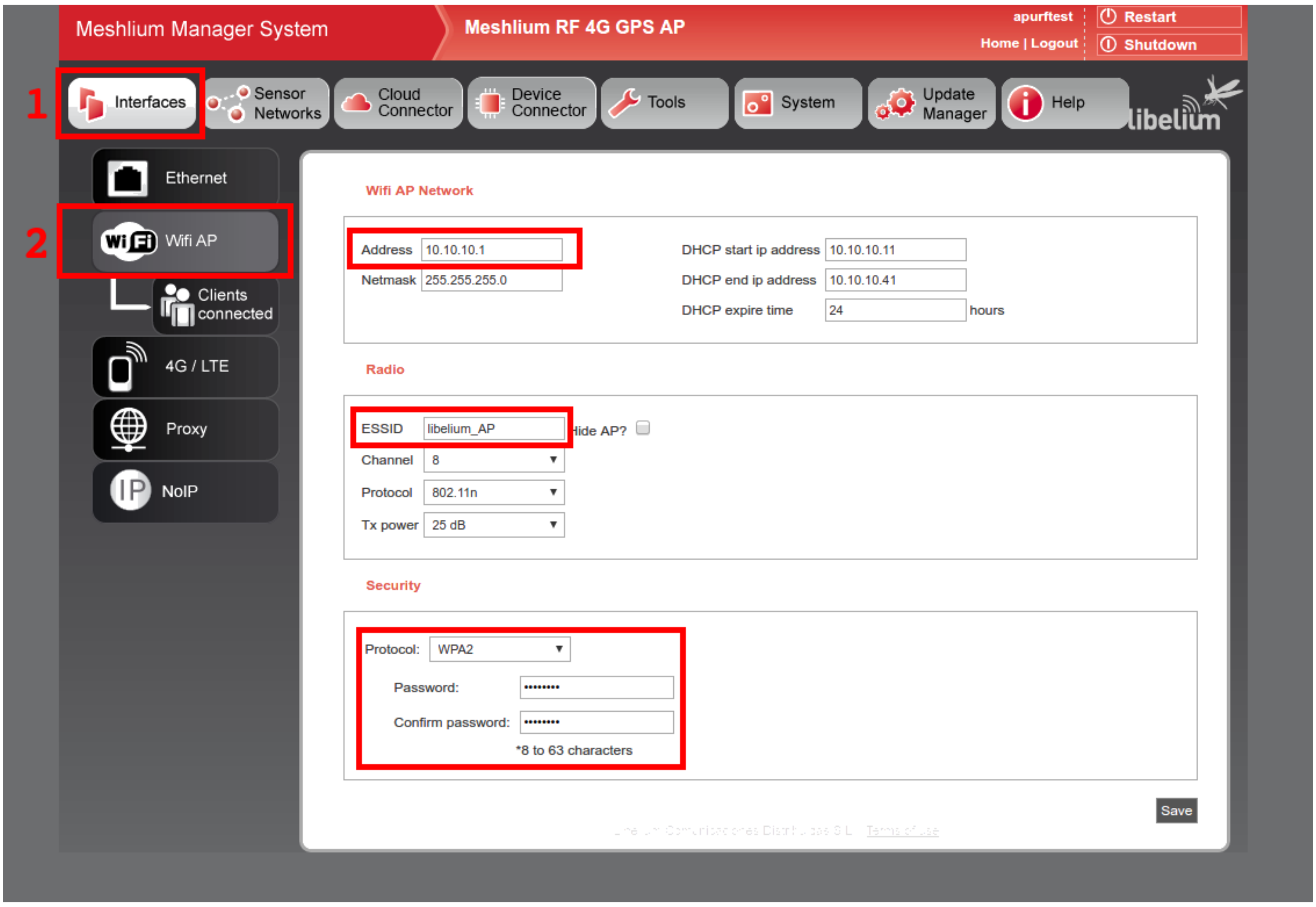
12677863D937425E
436D67057C105406

Node ID: 12677863D937425E
AES Secret Key: libeliumlibeliumlibeliumlibelium

Figure: Meshlium Manager System "payload encryption"

5.1.4. WiFi

The PCS form allows the user to select the “Send to Meshlium” protocol option. This permits the customer to configure the Plug & Sense! device to send data via HTTP request to the Meshlium device. Also, the user must keep in mind that the AP settings shall be defined correctly. The Meshlium Manager System permits to configure the AP settings:



Wifi AP Network

Address	10.10.10.1	DHCP start ip address	10.10.10.11
Netmask	255.255.255.0	DHCP end ip address	10.10.10.41
		DHCP expire time	24 hours

Radio

ESSID	libelium_AP
Hide AP?	☐
Channel	8
Protocol	802.11n
Tx power	25 dB

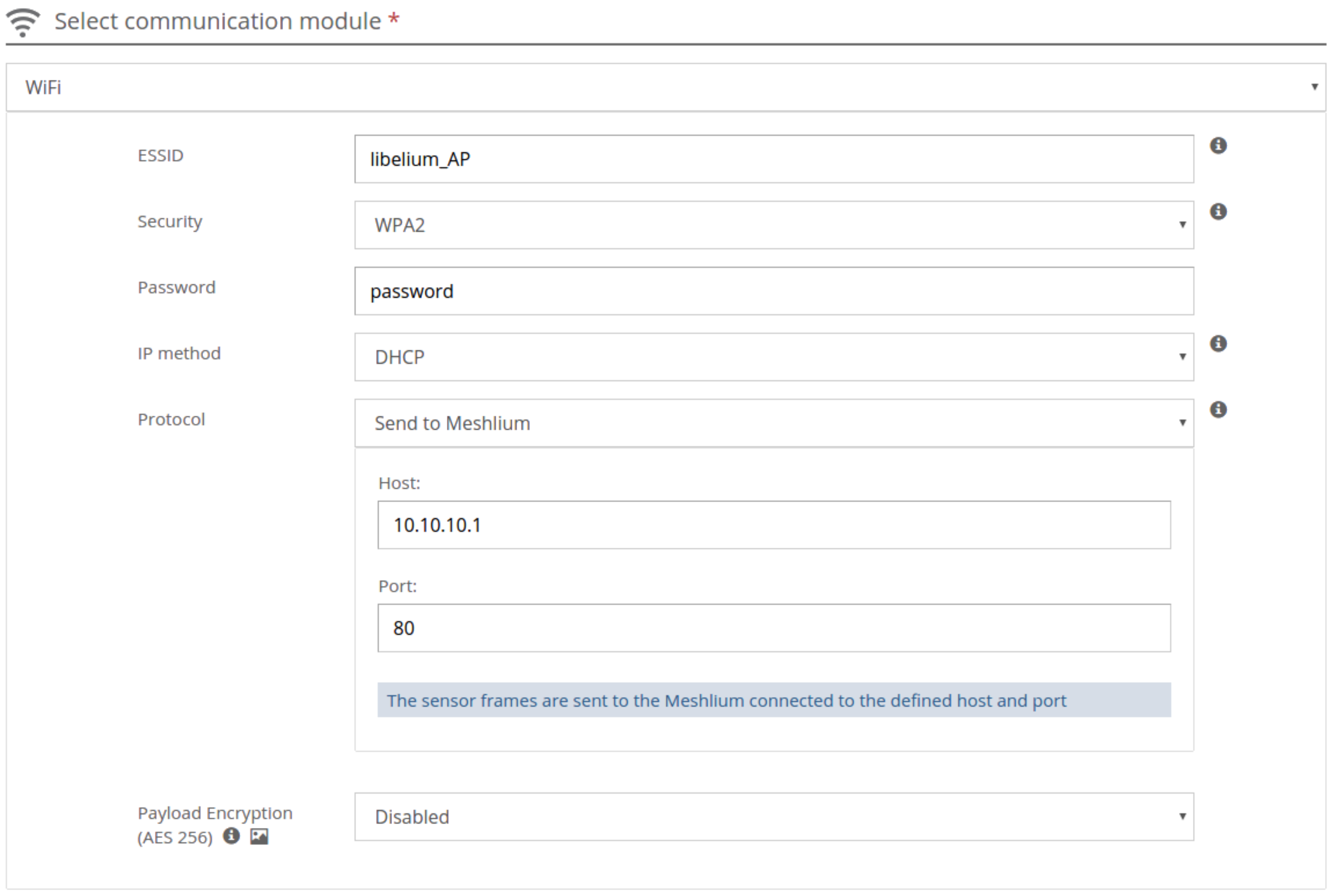
Security

Protocol:	WPA2
Password:	••••••••
Confirm password:	••••••••

*8 to 63 characters

Figure: Meshlium Manager System (WiFi AP settings)

On the other hand, the PCS form must be configured with the same settings:



Select communication module *

WiFi

ESSID	libelium_AP
Security	WPA2
Password	password
IP method	DHCP
Protocol	Send to Meshlium

Host: 10.10.10.1

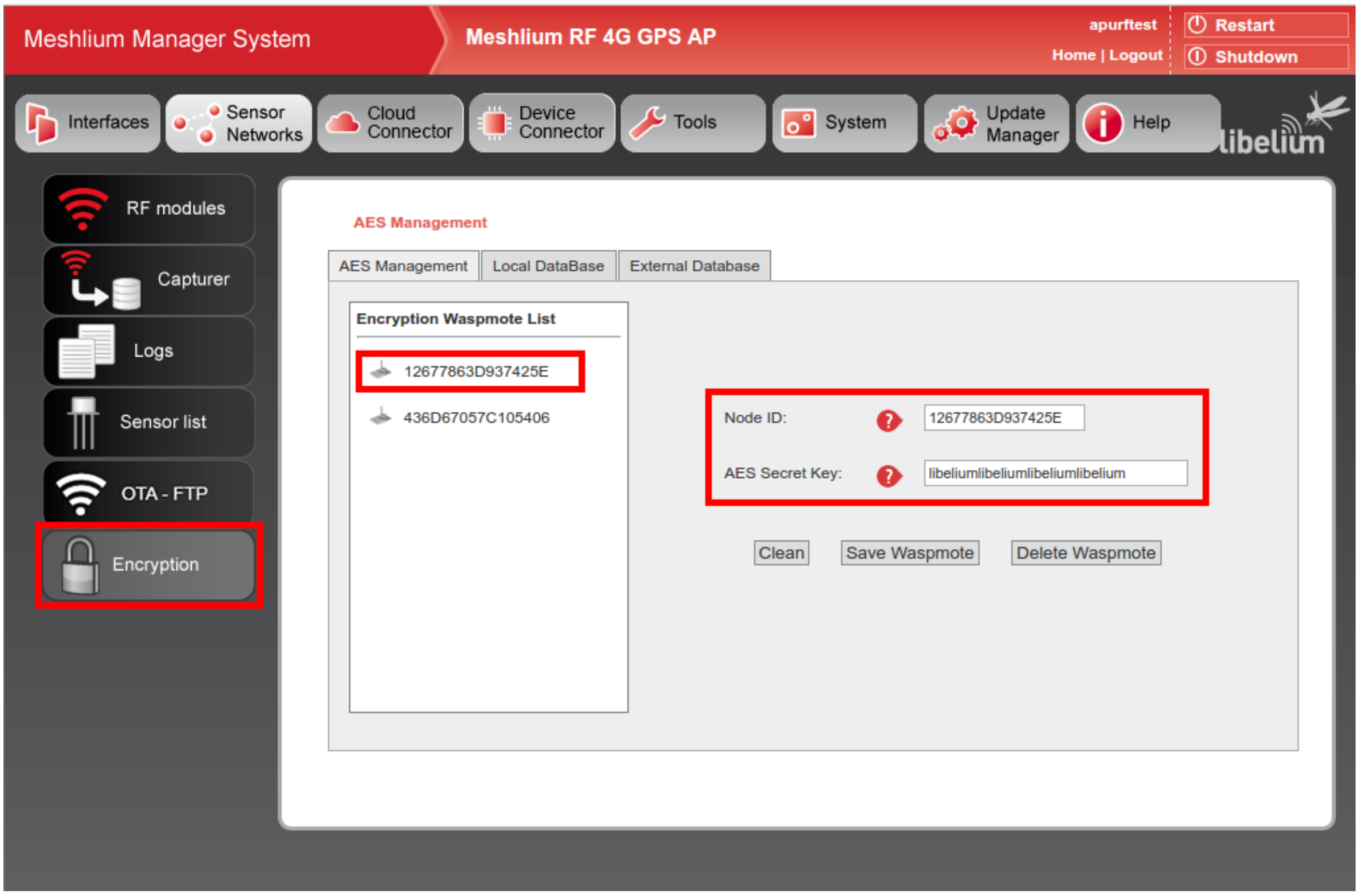
Port: 80

The sensor frames are sent to the Meshlium connected to the defined host and port

Payload Encryption (AES 256)	Disabled

Figure: PCS form (WiFi)

No encryption was configured in the previous example. If the user desires to enable the “Link encryption” or the “Payload encryption” layers, these features are also configured in both PCS form and Meshlium Manager System. Please refer to the “Programming Cloud Service configuration” section to see how you should configure the PCS form. On the other hand, the Meshlium Manager System provides the interfaces to enable both encryption layers:



Encryption Waspmote List

- 12677863D937425E
- 436D67057C105406

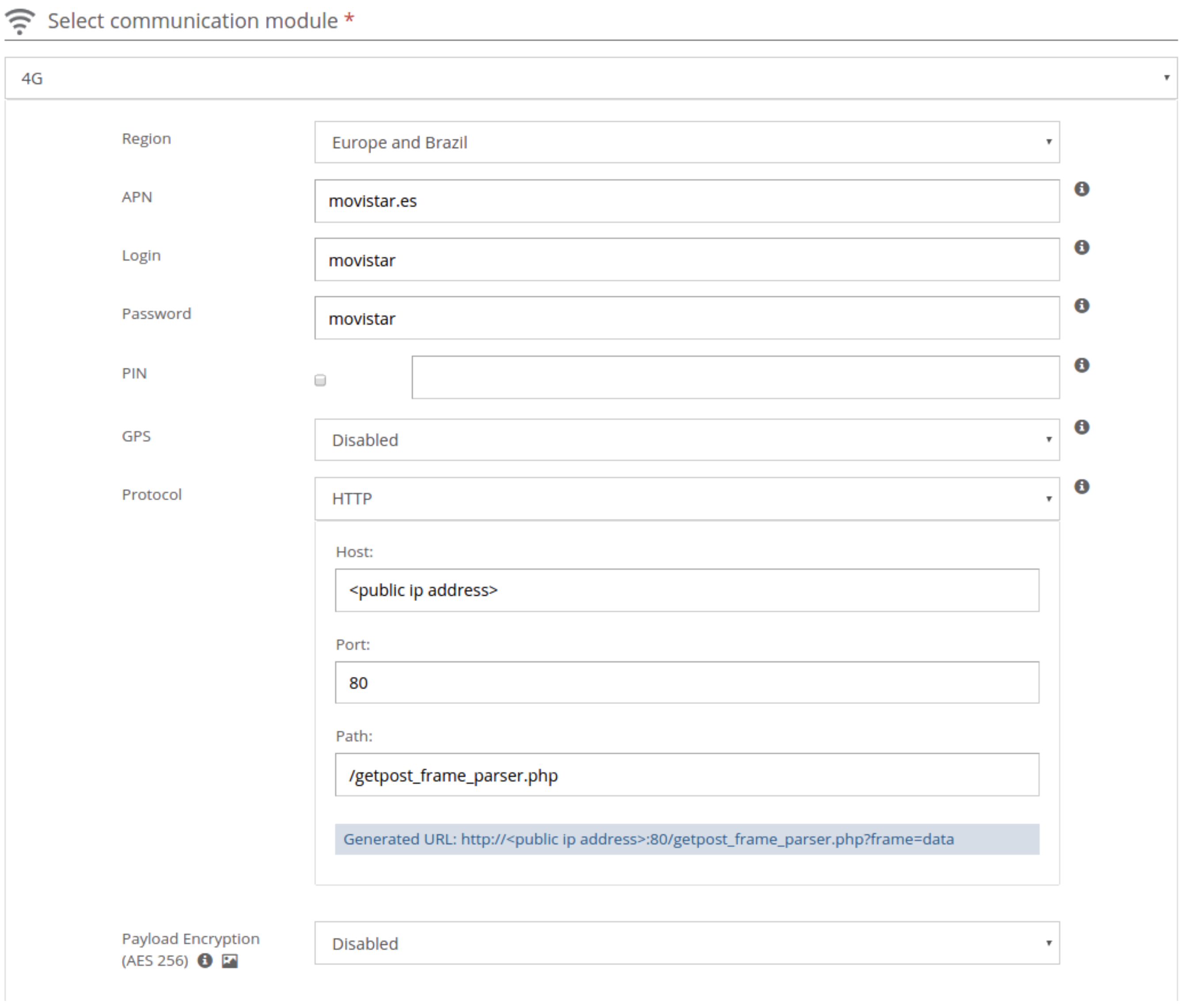
Node ID:	12677863D937425E
AES Secret Key:	libeliumlibeliumlibeliumlibelium

Figure: Meshlium Manager System “payload encryption“

5.1.5. 4G

In case of using the 4G radio, the user must keep in mind that the Meshlium's public IP address and port must be known in order to send data to Meshlium. The Meshlium device can be reached over Ethernet and 4G interfaces. The user is responsible for providing the correct IP address and port to the PCS form. These settings must be public so the 4G radio can reach them through an Internet connection.

Besides, the PCS form allows the user to select the "HTTP" protocol option in order to send data to Meshlium. The default "path" setting (/getpost_frame_parser.php) is needed as this is the file used by Meshlium in order to parse the data received from an HTTP request.


Select communication module *

4G

Region	Europe and Brazil
APN	movistar.es
Login	movistar
Password	movistar
PIN	
GPS	Disabled
Protocol	HTTP

Host: <public ip address>

Port: 80

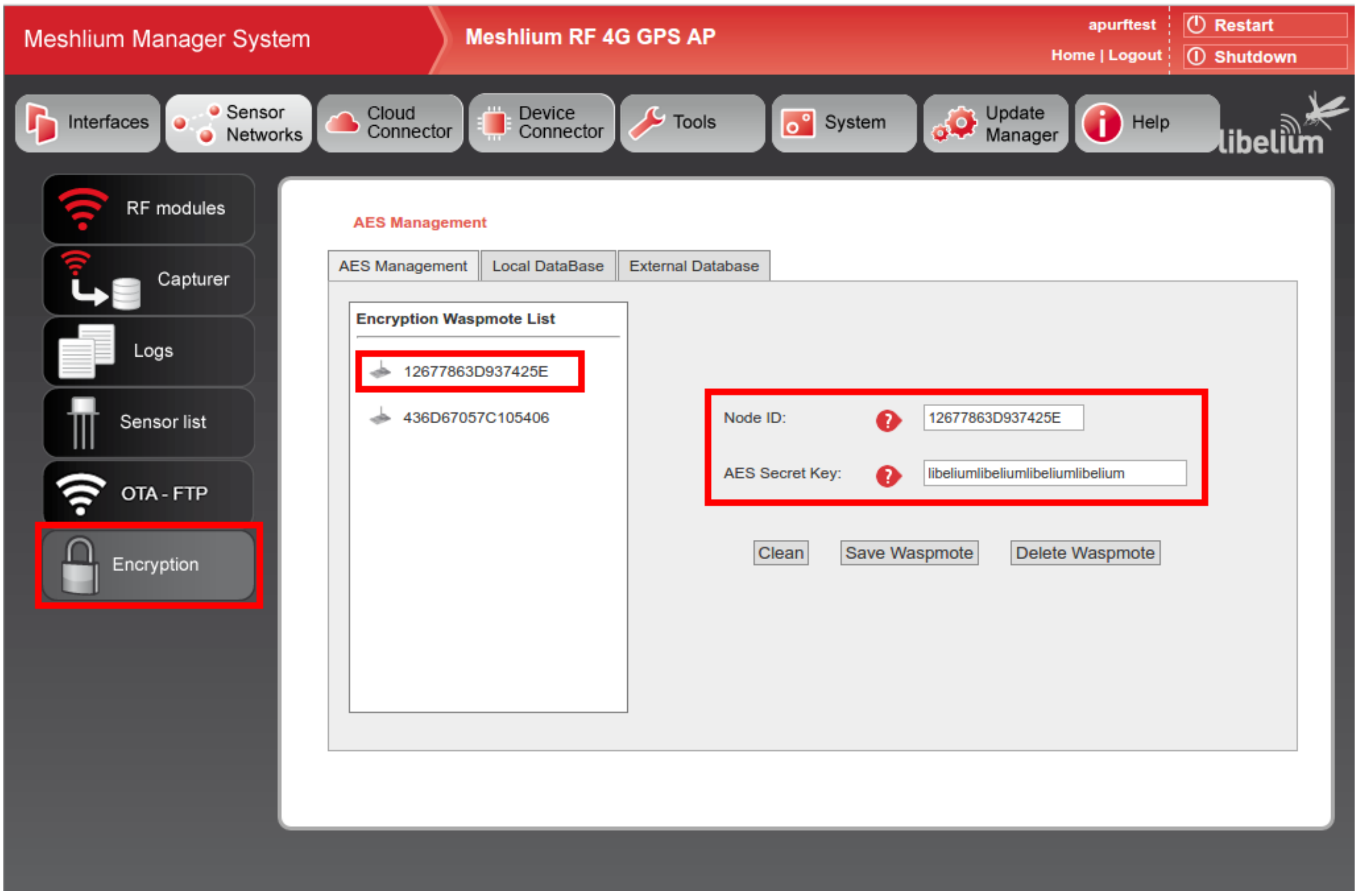
Path: /getpost_frame_parser.php

Generated URL: http://<public ip address>:80/getpost_frame_parser.php?frame=data

Payload Encryption (AES 256): Disabled

Figure: PCS form (4G)

No encryption was configured in the previous example. If the user desires to enable the “Link encryption” or the “Payload encryption” layers, these features are also configured in both PCS form and Meshlium Manager System. Please refer to the “Programming Cloud Service configuration” section to see how you should configure the PCS form. On the other hand, the Meshlium Manager System provides the interfaces to enable both encryption layers:



12677863D937425E
436D67057C105406

Node ID: 12677863D937425E
AES Secret Key: libeliumlibeliumlibeliumlibelium

Figure: Meshlium Manager System “payload encryption“

5.2. Configure Meshlium to send data to your Cloud system

The Meshlium Manager System provides several plugins in order to transmit the received data to an external Cloud system. The data received in Meshlium from the nodes of the network is stored in the internal database. Thus, the data is synchronized to an external server thanks the "Cloud Connector" process dedicated to this task.

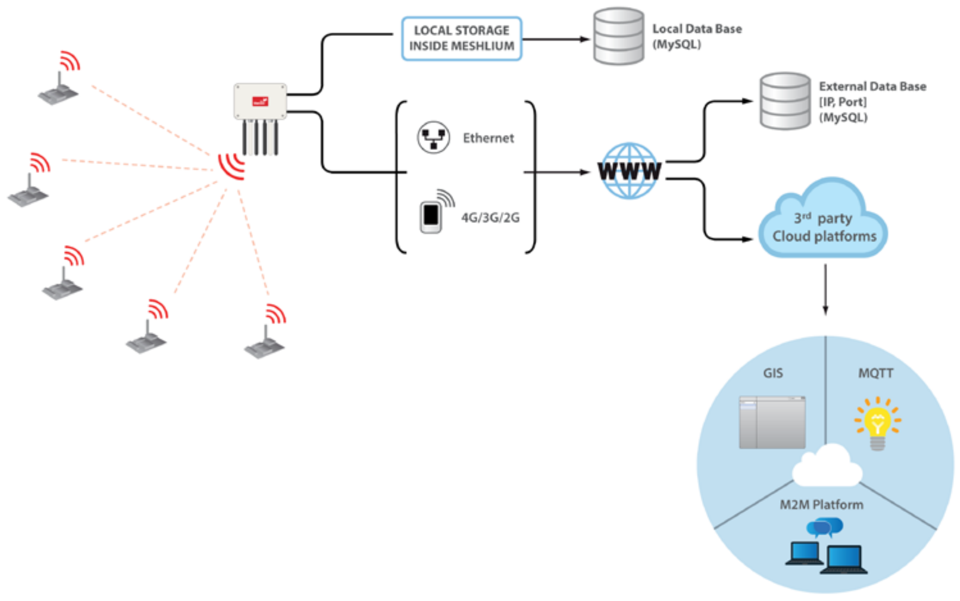

LOCAL STORAGE INSIDE MESHLIUM
Local Data Base (MySQL)
External Data Base [IP, Port] (MySQL)
Ethernet
4G/3G/2G
WWW
3rd party Cloud platforms
GIS
MQTT
M2M Platform

Figure: Meshlium database synchronization diagram

Meshlium runs a set of scripts for implementing the data synchronization from its internal database "to the cloud". In other words, those scripts send data to web servers where the cloud service providers host their clouds. Those scripts are called Cloud Connector.

All Cloud Connectors can be found in the "Cloud Connector" tab in the Meshlium Manager System. The user must select their own Cloud Connector and enter the credentials for initializing the process that will take care of the database synchronization. For more information, please refer to the "Cloud Connectors" section in the [Meshlium]() technical guide.

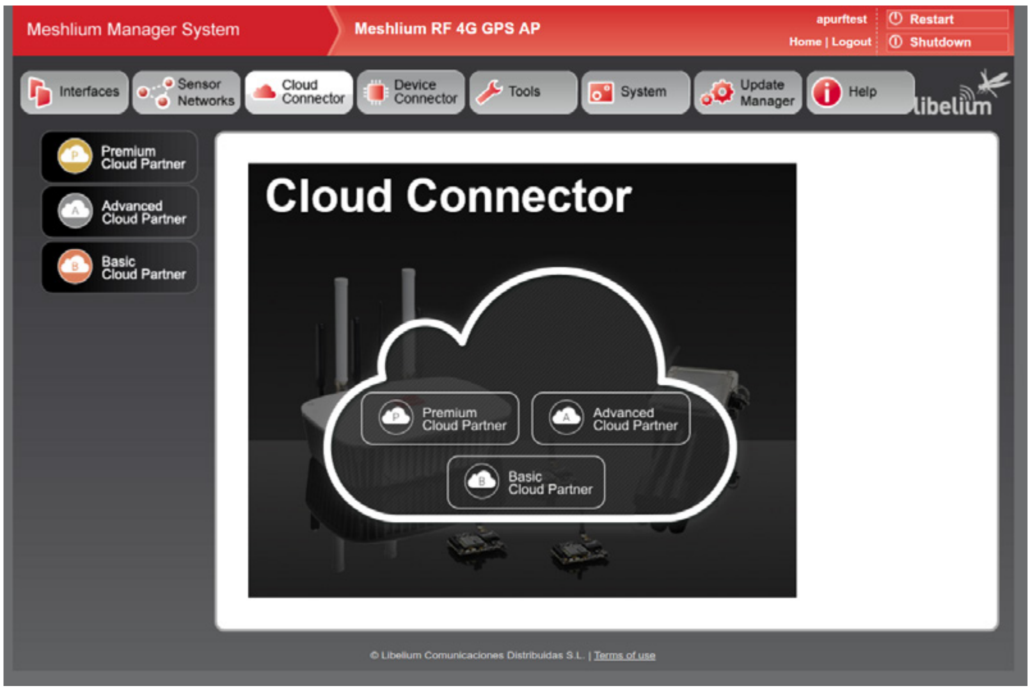


Figure: Cloud connector in Meshlium