INDEX

1. Introduction .......................................................................................................................... 4
   1.1. Programming Cloud Service basis .................................................................................. 4
   1.2. Licenses .......................................................................................................................... 5
   1.3. Purchasing licenses ......................................................................................................... 6
   1.4. Accessing to the Programming Cloud Service ............................................................... 6

2. Programming Cloud Service configuration ............................................................................ 8
   2.1. Templates ...................................................................................................................... 8
   2.2. Serial ID ...................................................................................................................... 8
   2.3. Node name .................................................................................................................. 8
   2.4. Select model ................................................................................................................. 9
   2.5. Select sensor by socket ............................................................................................... 9
       2.5.1. Smart Water calibration points .............................................................................. 10
       2.5.2. Smart Water Xtreme sensor calibration ............................................................... 10
       2.5.3. Smart Water Ions calibration points ..................................................................... 11
       2.5.4. Radiation Control measuring period ..................................................................... 11
       2.5.5. Smart Security events configuration ................................................................... 12
   2.6. Industrial protocol ....................................................................................................... 14
       2.6.1. Modbus over RS-232/485 .................................................................................. 14
       2.6.2. CAN Bus ............................................................................................................... 15
   2.7. GPS module ................................................................................................................ 16
   2.8. Sleep time .................................................................................................................... 17
   2.9. RTC Watchdog ............................................................................................................ 17
   2.10. Critical battery warning ............................................................................................ 18
   2.11. Select communication module .................................................................................. 19
       2.11.1. XBee-PRO 802.15.4 .......................................................................................... 19
       2.11.2. XBee-PRO 900HP ............................................................................................ 22
       2.11.3. XBee 868LP ...................................................................................................... 25
       2.11.4. XBee ZigBee 3 .................................................................................................... 27
       2.11.5. 4G ....................................................................................................................... 28
       2.11.6. WiFi ..................................................................................................................... 29
       2.11.7. Sigfox ................................................................................................................ 30
       2.11.8. LoRaWAN ......................................................................................................... 31
   2.12. Select protocol and destination ................................................................................... 33
       2.12.1. XBee-PRO 802.15.4 .......................................................................................... 34
       2.12.2. XBee-PRO 900HP ............................................................................................ 36
       2.12.3. XBee 868LP ...................................................................................................... 38
       2.12.4. 4G ....................................................................................................................... 40
       2.12.5. WiFi ..................................................................................................................... 44
2.12.6. Sigfox ...............................................................49
2.12.7. LoRaWAN .........................................................49
2.13. Check configuration ................................................50
2.14. Compile and Download Binary .................................50
2.15. Upload a program to Plug & Sense! .................51
  2.15.1. Smart Devices App installation .......................51
  2.15.2. Binary update ................................................51
2.16. Limitations ..........................................................54
  2.16.1. Sensor probe repetition ..................................54
  2.16.2. Memory issues ................................................54

3. How data frames are generated ......................................55
  3.1. Data frame types used by the PCS .......................55
  3.2. Sensor fields definition ........................................56
    3.2.1. Modbus sensor data ......................................56

4. How to get the CA certificate for HTTPS connections ........58
  4.1. SSL / TLS basis ....................................................58
  4.2. Certificate management .........................................59

5. Meshlium GW configuration and settings .....................63
  5.1. Configure Meshlium to send data to your Cloud system .63
  5.2. How to download the Meshlium certificate for HTTPS connections ..............................64
  5.3. How to enable the HTTP service for HTTP non-secure connections ......................67

6. Sending to the Libelium Cloud Bridge .........................68

7. Documentation changelog ............................................69
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:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Basic</th>
<th>PRO</th>
<th>Elite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes to program</td>
<td>6</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>Plug &amp; Sense! models supported</td>
<td>All</td>
<td>All</td>
<td>All</td>
</tr>
<tr>
<td>Sensors supported</td>
<td>All</td>
<td>All</td>
<td>All</td>
</tr>
<tr>
<td>Actuators control (Relay on/off in Plug &amp; Sense! Smart Security)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sigfox/LoRaWAN settings setup</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>802.15.4 / RF 868/900 MHz settings setup</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>4G/WiFi settings setup</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>GPS settings setup</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HTTP (4G/WiFi)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HTTPS (4G/WiFi)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>RF link encryption (802.15.4 / RF 868/900)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>AES-256 payload encryption</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Industrial Protocols support (Modbus over RS232/RS485, CAN Bus)</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Low battery warning</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Templates manager</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Batch programming (generate up to 100 binaries in one click)</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Price (per year)</td>
<td>200€</td>
<td>300€</td>
<td>1499€</td>
</tr>
</tbody>
</table>

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.

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:

![PCS form](image)

*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.

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.

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“.
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.

![Select model](image)

*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.

![Select sensor by socket](image)

*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](image)

Figure: Smart Water calibration points

2.5.2. Smart Water Xtreme sensor calibration

Compared with other smart water sensors, the Smart Water Xtreme sensors can be easily calibrated following the instructions printed on the serial monitor. There is no need of selecting any calibration point in the configuration step.

Once the compiled binary has been uploaded into a node, a message will be prompted on the serial monitor asking to type 'C' and press 'Enter' to enter the calibration menu and therefore begin the calibration process. If no key is pressed, then the code will continue after a few seconds. Bear in mind that only sensor probes placed on socket E can be calibrated. A complete description of the calibration process of each sensor can be found on the Smart Water Xtreme technical guide.

Note: Remember that the calibration process can be only be performed on socket E, because it is the only socket with RS-485 signals (needed for calibration). The calibration process stores the calibration values on the sensor's memory, so afterwards it can be connected to any socket. That means that if your node has, for example, 2 sensors probes, you will create 2 binaries with the PCS for calibration (each sensor, always in socket E), and then a final 3rd binary with the final sensor distribution (each calibrated sensor can be connected to any available socket, for example, A and C).
2.5.3. 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.

![Smart Water Ions calibration points](image)

**Figure:** Smart Water Ions calibration points

2.5.4. 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).

![Radiation Control measuring time](image)

**Figure:** Radiation Control measuring time
2.5.5. 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.

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.
Programming Cloud Service configuration

Figure: Smart Security sensor events configuration (without 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.

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)

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)

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.

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 PCS 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.

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:](image)

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.

2.11.1. XBee-PRO 802.15.4

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**: 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.

When working with this protocol, the user needs to define the same parameters on all nodes in the network (that includes the Plug & Sense! device(s) and the Meshlium device). Besides, the Meshlium's XBee radio MAC address must be inserted on the PCS form.

The Manager System permits to configure the Meshlium's XBee radio as follows:
The PCS form should be filled with the same configuration that the receiving Meshlium has:

If the user wants to enable the “link encryption” security layer, this feature must be also configured in both PCS form and Meshlium Manager System. The Meshlium Manager System provides the interface to enable link encryption:
For further information, refer to the [802.15.4 Networking Guide](#).
2.11.2. XBee-PRO 900HP

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 PCS 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 00000000001FFFFFF to FFFFFFFFFFFFFFF. 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**: 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.

When working with this protocol, the user needs to define the same parameters on all nodes in the network (that includes the Plug & Sense! device(s) and the Meshlium device). Besides, the Meshlium's XBee radio MAC address must be inserted on the PCS form.

The Manager System permits to configure the Meshlium’s XBee radio as follows:

![Meshlium Manager System (XBee-PRO 900HP)](image_url)
The PCS form should be filled with the same configuration that the receiving Meshlium has:

![Figure: PCS XBee-PRO 900HP configuration](image)

If the user wants to enable the “link encryption” security layer, this feature must be also configured in both PCS form and Meshlium Manager System. The Meshlium Manager System provides the interface to enable link encryption:
Figure: Meshlium Manager System link encryption configuration

For further information, refer to the 900 Networking Guide.
2.11.3. XBee 868LP

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 PCS 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**: 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](#).

![Meshlium Manager System (XBee 868LP)](image-url)
The PCS form should be filled with the same configuration that the receiving Meshlium has:

![Select communication module](image)

**Figure: PCS XBee 868LP configuration**

If the user wants to enable the “link encryption” security layer, this feature must be also configured in both PCS form and Meshlium Manager System. The Meshlium Manager System provides the interface to enable link encryption:

![Meshlium Manager System link encryption configuration](image)

**Figure: Meshlium Manager System link encryption configuration**

For further information, refer to the [868 Networking Guide](#).
2.11.4. XBee ZigBee 3

The XBee ZigBee 3 can send frames to another XBee ZigBee radio. A ZigBee receiver device should be used in order to receive the frames from the Plug & Sense! device (Meshlium or Bridge do not support ZigBee). The PCS can configure several parameters:

- **MAC**: Defines the destination MAC address. Where the packet is sent to. If you are using a ZigBee receiver device to collect the data, it should allow to read and display its XBee radio MAC address (or maybe it's shown in an external sticker). 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**: Extended Personal Area Network identifier. This is a 8-byte field. It is necessary to define it as 16 hexadecimal digits (from 0000000000000000 to FFFFFFFFFFFFFF). It must be the same as the receiver's.

- **Channel mask**: The channels configuration where the ZigBee will scan the network. It must be the same as the receiver's configuration. This is a 2-byte field.

Link encryption: 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. This encryption layer is related to “Device to Device” encryption level described in the IoT Security Infographic.

When working with this protocol, the user needs to define the same parameters on all nodes in the network (that includes the Plug & Sense! device(s) and the ZigBee receiver device). Besides, the XBee ZigBee receiver radio MAC address must be inserted on the PCS form.

The PCS form should be filled with the same configuration that the receiving ZigBee device has:

For further information, refer to the ZigBee Networking Guide.
2.11.5. 4G

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

The PCS 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.

For further information, refer to the [4G Networking Guide](#).

[Select communication module *]

![4G configuration](#)

*Figure: 4G configuration*
2.11.6. WiFi

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 PCS 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 or DHCP methods.
  - Static: IP address, DNS, gateway and netmask addresses must be defined
  - DHCP: no parameters required.

The PCS form allows the user to select the “Send to Meshlium” protocol option. This will configure the Plug & Sense! device to send data via HTTPS request to the Meshlium unit. Also, the user must make sure that the AP settings are defined correctly. The Meshlium Manager System permits to configure the AP settings:

![Meshlium Manager System (WiFi AP settings)](image)

On the other hand, the PCS form must be configured with the same settings: SSID, security and password.
For further information, refer to the WiFi Networking Guide.

2.11.7. 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.
2.11.8. 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 among LoRaWAN EU, LoRaWAN US, LoRaWAN AU, LoRaWAN IN, LoRaWAN ASIA-PAC / LATAM and LoRaWAN JP / KR.
- **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.
### Programming Cloud Service configuration

#### Select communication module *

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LoRaWAN</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Region</strong></td>
<td>Select</td>
</tr>
<tr>
<td><strong>ADR</strong></td>
<td>Select</td>
</tr>
<tr>
<td><strong>Sending mode</strong></td>
<td>Select</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>Protocol</strong></td>
<td>Select</td>
</tr>
</tbody>
</table>

💡 Current European RF legislation allows to use the transmission mode the 1% of duty cycle. For Sigfox this means a maximum of 140 messages per day (128 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 $\rightarrow$ 128+8B), you can check that in the summary below. Take this into account when setting the transmission options.

*Figure: LoRaWAN configuration*

For further information, refer to the [LoRaWAN Networking Guide](#).
2.12. Select protocol and destination

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

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

LoRaWAN and Sigfox protocols cannot send data directly to a final cloud service or Meshlium as they send frames to the customer’s LoRaWAN cloud (the Network Server) and the Sigfox Cloud, respectively.

The XBee radios send data directly to Meshlium as there is another XBee radio inside Meshlium that allows reception. The XBee ZigBee 3 radio module is an exception since Meshlium does not support the ZigBee protocol.

Regarding the WiFi radio, it can connect to the Meshlium’s WiFi Access Point (AP) and proceed with the HTTP request. However, it is also possible to send data to Meshlium via Ethernet or 4G interface, or directly to a final cloud service if a WiFi 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 radio, it always uses the mobile network operator infrastructure, so they may use the Meshlium’s Ethernet or 4G interfaces in order to send sensor data to Meshlium.

Figure: Libelium Cloud Bridge service data flow

Note: The “Use Encryption Key” and “HTTPS” features are only supported by PRO and Elite license versions.
Programming Cloud Service configuration

The “Use Encryption Key” 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.

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

2.12.1. XBee-PRO 802.15.4

In the previous section we learned how to send frames to the XBee module that Meshlium has. Besides, the PCS allows the user to select the protocol and destination of the frame. When using XBee modules, the protocol is fixed and the PCS allows the user to select between 2 destinations: “Send to Meshlium GW” and “Send to Libelium Cloud Bridge”.

- Send to Meshlium GW (Meshlium Gateway)

![Image of PCS destination form (XBee-PRO 802.15.4 to Meshlium GW)](image)

The Meshlium Manager System provides the interface to enable payload encryption:

![Image of Meshlium Manager System payload encryption configuration](image)

When Meshlium is selected as destination, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 96” frames.
• Send to Libelium Cloud Bridge service

Select protocol and destination *

![Image of PCS destination form](image)

**Figure: PCS destination form (XBee-PRO 802.15.4 to Bridge)**

If the user wants to enable the payload encryption security layer, this feature must be also configured in both PCS form and the Libelium Cloud Bridge. The Bridge provides the interface to enable payload encryption:

![Image of Libelium Cloud Bridge Device Manager payload encryption configuration](image)

**Figure: Libelium Cloud Bridge Device Manager payload encryption configuration**

When the Bridge is selected as destination, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

Please refer to the [Waspmote Data Frame Programming Guide](link) to see the different types of frames and their behaviour.
2.12.2. XBee-PRO 900HP

In the previous section we learned how to send frames to the XBee module that Meshlium has. Besides, the PCS allows the user to select the protocol and destination of the frame. When using XBee modules, the protocol is fixed and the PCS allows the user to select between 2 destinations: “Send to Meshlium GW” and “Send to Libelium Cloud Bridge”.

• Send to Meshlium GW (Meshlium Gateway)

![PCS destination form (XBee-PRO 900HP to Meshlium GW)](image)

The Meshlium Manager System provides the interface to enable payload encryption:

![Meshlium Manager System payload encryption configuration](image)

When Meshlium is selected as destination, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 96” frames.
• Send to Libelium Cloud Bridge service

If the user wants to enable the payload encryption security layer, this feature must be also configured in both PCS form and the Libelium Cloud Bridge. The Bridge provides the interface to enable payload encryption:

When the Bridge is selected as destination, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

Please refer to the "Wasp mote Data Frame Programming Guide" to see the different types of frames and their behaviour.
2.12.3. XBees 868LP

In the previous section we learned how to send frames to the XBees that Meshlium has. Besides, the PCS allows the user to select the protocol and destination of the frame. When using XBees modules, the protocol is fixed and the PCS allows the user to select between 2 destinations: “Send to Meshlium GW” and “Send to Libelium Cloud Bridge”.

- Send to Meshlium GW (Meshlium Gateway)

![PCS destination form (XBee-PRO 868LP to Meshlium GW)](image)

The Meshlium Manager System provides the interface to enable payload encryption:

![Meshlium Manager System payload encryption configuration](image)

When Meshlium is selected as destination, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 96” frames.
• Send to Libelium Cloud Bridge service

![PCS destination form (XBee-PRO 868LP to Bridge)](image)

If the user wants to enable the payload encryption security layer, this feature must be also configured in both PCS form and the Libelium Cloud Bridge. The Bridge provides the interface to enable payload encryption:

![Libelium Cloud Bridge Device Manager payload encryption configuration](image)

When the Bridge is selected as destination, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

Please refer to the [Waspmote Data Frame Programming Guide](#) to see the different types of frames and their behaviour.
2.12.4. 4G

Besides the required configuration for the module to connect to the access point properly. The PCS allows the user to select the “protocol and destination” of the frame. When using the 4G communication module, the PCS allows the user among different options.

- Send to Libelium Cloud Bridge
  When this option is selected, the frames sent from P&SI will be type 103, AES128 encrypted.

  ![Select protocol and destination](image)

  *Figure: PCS destination form (4G to Bridge)*

  If the “Use encryption Key” box is enabled the P&SI will encrypt using AES256 the content of the frame before doing the previous AES128 encryption. This results in a frame type 100 inside of a frame type 103.

- Send to Meshlium GW
  When this option is selected, the P&SI will send type 6 frames, type 7 frames or type 8 frames, depending on the sensor board within the P&SI. If the “Use encryption Key” box is enabled the P&SI will send type 96 frames.

  When 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.

  Also, the Meshlium certificate must be specified in the PCS form. In order to know how to download the Meshlium certificate please refer to the section “How to download the Meshlium certificate for HTTPS connections”.
If the user wants to enable the payload encryption security layer, this feature must be also configured in both PCS form and the Libelium Cloud Bridge. The Bridge provides the interface to enable payload encryption:

- SMS

When this option is selected, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.
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.

When this option is selected as destination, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

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.

**Figure**: PCS destination form (4G HTTP)
When this option is selected, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

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>?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.

![Select protocol and destination form](https://test.libelium.com:443/get_post_frame_parser.php?frame=data)

**Figure: PCS destination form (4G HTTPS)**

When this option is selected, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

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.

Please refer to the "Waspmote Data Frame Programming Guide" to see the different types of frames and their behaviour.
2.12.5. WiFi

Besides the required configuration for the module to connect to the access point properly, the PCS allows the user to select the “protocol and destination” of the frame. When using the WiFi communication module, the PCS allows the user among different options.

- Send to Libelium Cloud Bridge
  When this option is selected, the frames sent from P&S! will be type 103, AES128 encrypted.

  ![Select protocol and destination](image)

  Figure: PCS destination form (WiFi to Bridge)

  If the “Use encryption Key” box is enabled the P&S! will encrypt using AES256 the content of the frame before doing the previous AES128 encryption. This results in a frame type 100 inside of a frame type 103.

- Send to Meshlium GW
  When this option is selected, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 96” frames.

  The “Select protocol and destination” form must be configured with the same settings defined in the Meshlium Manager System and the Meshlium certificate. In order to know how to download the Meshlium certificate please refer to the section “How to download the Meshlium certificate for HTTPS connections”.

  ![Select protocol and destination](image)

  Figure: PCS destination form (WiFi to Meshlium GW)

  If the user wants to enable the payload encryption security layer, this feature must be also configured in both PCS form and the Libelium Cloud Bridge. The Bridge provides the interface to enable payload encryption:
Programming Cloud Service configuration

Figure: Meshlium Manager System payload encryption configuration
• **HTTP**

When this option is selected, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

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.

![Select protocol and destination](image)

*Figure: PCS destination form (WiFi HTTP)*
When this option is selected, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense!. If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

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.

![Select protocol and destination](PCS destination form (WiFi HTTPS))
• TCP
When this option is selected, the Plug & Sense! unit will send “type 6” frames, “type 7” frames or “type 8” frames, depending on the sensor board inside Plug & Sense! If the “Use encryption Key” box is enabled, Plug & Sense! will send “type 100” frames.

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.

![Select protocol and destination *](image)

**Figure: PCS destination form (WiFi TCP)**

Please refer to the [“Waspmote Data Frame Programming Guide”](#) to see the different types of frames and their behaviour.
2.12.6. Sigfox

Sigfox protocol cannot send data directly to a Cloud or Meshlium as they send sensor frames to the Sigfox Cloud.

Regarding the PCS, the “Select protocol and destination” section will be disabled since the destination of the packets must be set in the Sigfox backend.

```
Select protocol and destination *

[ ] Current European RF legislation allows for the use of 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 -> 128+8B), you can check that in the summary below. Take this into account when setting the transmission options
```

Figure: PCS destination form (Sigfox)

2.12.7. LoRaWAN

LoRaWAN protocol cannot send data directly to a Cloud or Meshlium as they send sensor frames to the customer's LoRaWAN network server.

Regarding the PCS, the “Select protocol and destination” section will be disabled since the destination of the packets must be set in the LoRaWAN network server.

```
Select protocol and destination *

[ ] Current European RF legislation allows for the use of 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 -> 128+8B), you can check that in the summary below. Take this into account when setting the transmission options
```

Figure: PCS destination form (LoRaWAN)
2.13. 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.

2.14. 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
2.15. 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.

2.15.1. 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).

2.15.2. 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).

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

![Smart Devices App successful message](image)

*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.

![Smart Devices App error message](image)

*Figure: Smart Devices App error message*
2.16. Limitations

2.16.1. 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’.

2.16.2. 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 of 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.

<table>
<thead>
<tr>
<th>HEADER</th>
<th>PAYLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Type</td>
<td>Num of bytes</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>HEADER</th>
<th>PAYLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td>Length</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>HEADER</th>
<th>PAYLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Type</td>
<td>Num of bytes</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Radio</th>
<th>Frame type</th>
<th>Payload encryption (AES-256) feature?</th>
</tr>
</thead>
<tbody>
<tr>
<td>XBee-PRO 802.15.4</td>
<td>Binary</td>
<td>Yes</td>
</tr>
<tr>
<td>XBee-PRO 900HP</td>
<td>Binary</td>
<td>Yes</td>
</tr>
<tr>
<td>XBee 868LP</td>
<td>Binary</td>
<td>Yes</td>
</tr>
<tr>
<td>XBee ZigBee</td>
<td>Binary</td>
<td>No</td>
</tr>
<tr>
<td>4G</td>
<td>Binary</td>
<td>Yes</td>
</tr>
<tr>
<td>WiFi</td>
<td>Binary</td>
<td>Yes</td>
</tr>
<tr>
<td>Sigfox</td>
<td>Tiny</td>
<td>No</td>
</tr>
<tr>
<td>LoRaWAN</td>
<td>Tiny</td>
<td>No</td>
</tr>
</tbody>
</table>

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 structure](image)

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
How data frames are generated

In the form you should enter the next settings:

![Industrial protocol settings](image)

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:

<table>
<thead>
<tr>
<th>Sensor ID</th>
<th>Sensor field 1</th>
<th>Sensor field 2</th>
<th>Sensor field 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0</td>
<td>0xB7</td>
<td>0x11</td>
<td>0x00</td>
</tr>
<tr>
<td>Byte 1</td>
<td>0x11</td>
<td>0x92</td>
<td>0x11</td>
</tr>
<tr>
<td>Byte 2</td>
<td>0x00</td>
<td>0x15</td>
<td>0x01</td>
</tr>
<tr>
<td>Byte 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure: Sensor data structure
4. How to get the CA certificate for HTTPS connections

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.

![SSL/TLS negotiation process between Client and Server](image)

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.

![Access to the website](image1)

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

![Get the details of the Server's certificate](image2)

**Step 3:** View Server's certificate.

![View Server's certificate](image3)
How to get the CA certificate for HTTPS connections

**Step 4:** Extract information about the Certificate Authority.

![Certificate Viewer](image1)

**Figure:** Extract information about the Certificate Authority

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

![Browser Settings](image2)
How to get the CA certificate for HTTPS connections

**Figure: Web browser settings**

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

![Manage certificates in Chrome](image1)

**Figure: HTTPS/SSL settings**

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

![Export CA certificate](image2)
Step 8: Now you can open the certificate with a text editor application and copy it into the PCS form.

![Certificate Image](image.png)

Figure: Copy CA certificate contents
5. Meshlium GW configuration and settings

5.1. 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 to the “Cloud Connector” process dedicated to this task.

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.
5.2. How to download the Meshlium certificate for HTTPS connections

Since Meshlium Manager System v4.0.9, HTTPS is the default method for sending data. HTTPS is the recommended technology because it provides many cyber security services.

For HTTPS, the user must keep in mind that the Meshlium’s certificate has to be installed on the Waspmote or Plug & Sense! radio prior to opening secure connections. The next picture shows how the user can download the Meshlium’s certificate from Manager System —› System —› Users Manager —› Download Certificate:

Figure: Meshlium certificate export process
Then, the certificate can be opened and copied from a normal text editor in order to paste the contents of the certificate into the PCS form. So when the PCS user selects some protocol which sends data to Meshlium via HTTPS, it is mandatory to insert the corresponding Meshlium's certificate as it is shown in the next image.

Figure: Copy the certificate from text editor
Figure: Paste the certificate into the PCS form
5.3. How to enable the HTTP service for HTTP non-secure connections

Since Meshlium Manager System v4.0.9, HTTPS is the default method in order to send data. HTTPS is the recommended technology because it provides many cyber security services. Therefore, the HTTPS service is always enabled on Meshlium.

However, Meshlium Manager System v4.1.0 and greater versions allow the user to enable the HTTP service in order to be able to receive HTTP non-secure requests. The user must go to Manager System → System → Security → HTTP Service:

![Image: Enable HTTP service in Meshlium Manager System](image)
6. Sending to the Libelium Cloud Bridge

The Programming Cloud Service has the option “Send to Libelium Cloud Bridge” in the “Protocol and destination” section to generate binaries that allow Plug & Sense! to send data to the Libelium Cloud Bridge service.

In all cases, the recommended option is “Send to Libelium Cloud Bridge”. Depending on the radio module, data (a) will need to go through the Meshlium gateway, (b) will be sent to the Sigfox or LoRaWAN server and forwarded from there, or (c) will be sent directly to the Libelium Cloud Bridge service.

The Libelium Cloud Bridge features all the security and encryption layers that can be configured in the PCS.

The information needed in the Programming Cloud Service to generate the specific binary 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).

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, etc) as shown in the next diagram.

![Diagram: Libelium Cloud Bridge service data flow](image)

*Figure: Libelium Cloud Bridge service data flow*
7. Documentation changelog

From v7.4 to v7.5
• Added references to new regional versions of the LoRaWAN radio

From v7.4 to v7.5
• Added references to the new XBee ZigBee 3 radio

From v7.3 to v7.4
• Added references to the new Libelium Cloud Bridge service

From v7.2 to v7.3
• Modified the PCS Basic License conditions (added 3 security features, 6 nodes instead of 5)

From v7.1 to v7.2
• Added references to Smart Water Xtreme.

From v7.0 to v7.1
• Added indications to send frames to Meshlium via HTTP or HTTPS for WiFi/4G communication modules. By default, Meshlium Manager System only permits secure connections (HTTPS) from version equal or greater than v4.0.9.