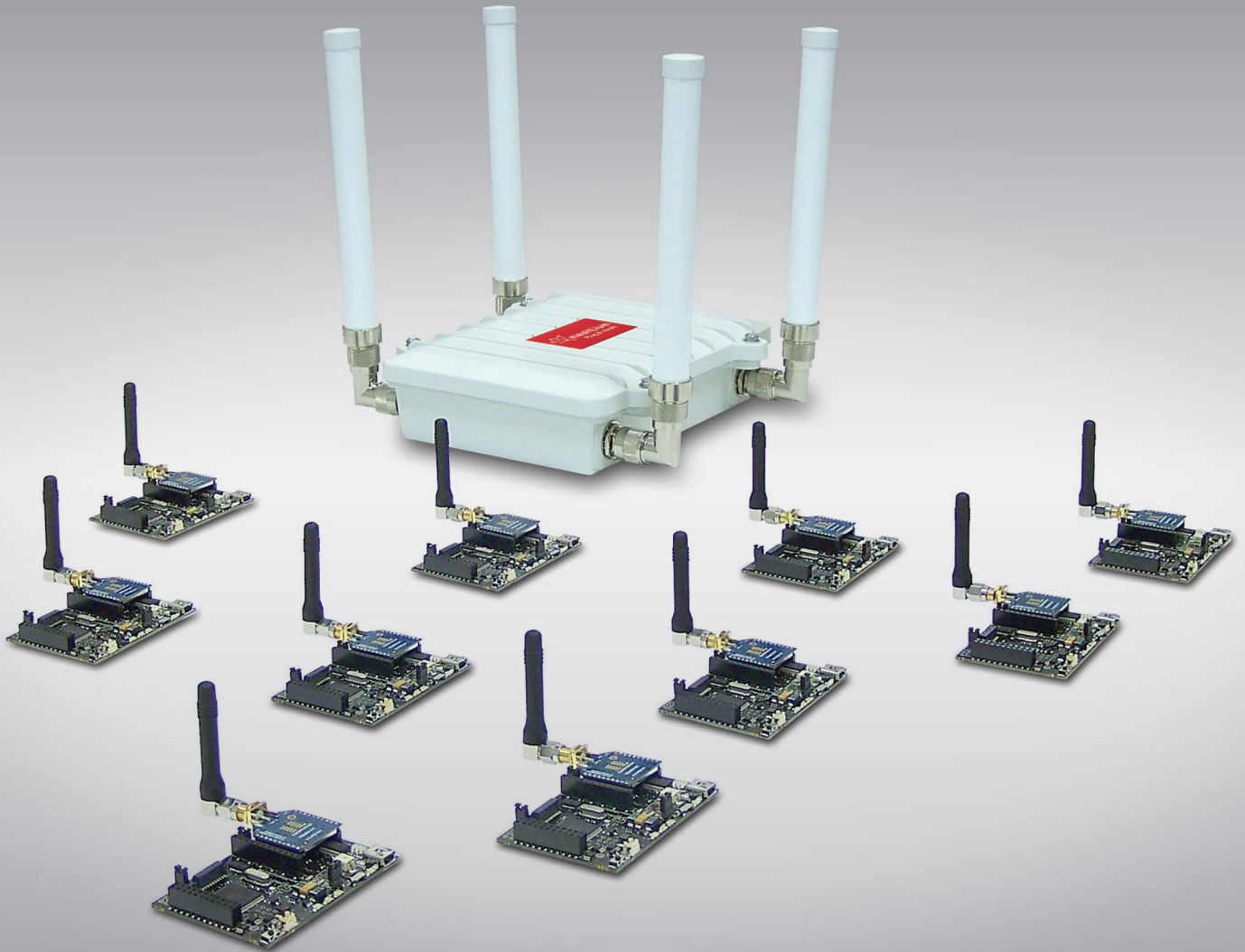


# Wireless Sensor Networks with Wasp mote and Meshlium



## Wireless Sensor Networks with Waspote and Meshlium

### • Three Libelium technologies:

**Waspote** is a sensor device specially oriented to developers. It works with different protocols (ZigBee, Bluetooth, GPRS) and frequencies (2.4GHz, 868MHz, 900MHz) being capable of getting links up to 12km. It counts with an hibernate mode of 0.6uA which allows to save battery when it is not transmitting. More than 50 sensors already available and a complete open source IDE (API libraries + compiler) made really easy to start working with the platform.

More info: <http://www.libelium.com/waspote>

**The new Waspote Plug & Sense!** line allows developers to forget about electronics and focus on services and applications. Now you can deploy wireless sensor networks in a easy and scalable way ensuring minimum maintenance costs. The new platform consists of a robust waterproof enclosure with specific external sockets to connect the sensors, the solar panel, the antenna and even the USB cable in order to reprogram the node. It has been specially designed to be scalable, easy to deploy and maintain.

More info: [http://www.libelium.com/plug\\_&\\_sense](http://www.libelium.com/plug_&_sense)

**Meshlium** is a Linux router which works as the Gateway of the Waspote Sensor Networks. It can contain 5 different radio interfaces: WiFi 2.4GHz, WiFi 5GHz, 3G/GPRS, Bluetooth and **ZigBee**. As well as this, Meshlium can also integrate a GPS module for mobile and vehicular applications and be solar and battery powered. These features a long with an aluminium IP-65 enclosure allows Meshlium to be placed anywhere outdoor. Meshlium comes with the Manager System, a web application which allows to control quickly and easily the WiFi, ZigBee, Bluetooth and 3G/GPRS configurations a long with the storage options of the sensor data received.

The new Meshlium Xtreme allows to detect iPhone and Android devices and in general any device which works with WiFi or Bluetooth interfaces. The idea is to be able to measure the amount of people and cars which are present in a certain point at a specific time, allowing the study of the evolution of the traffic congestion of pedestrians and vehicles.

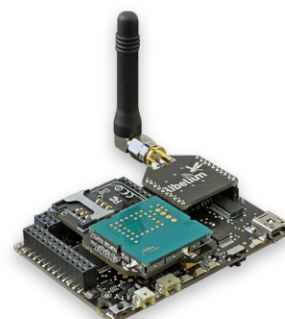
More info: <http://www.libelium.com/meshlium>

### How do they work together?

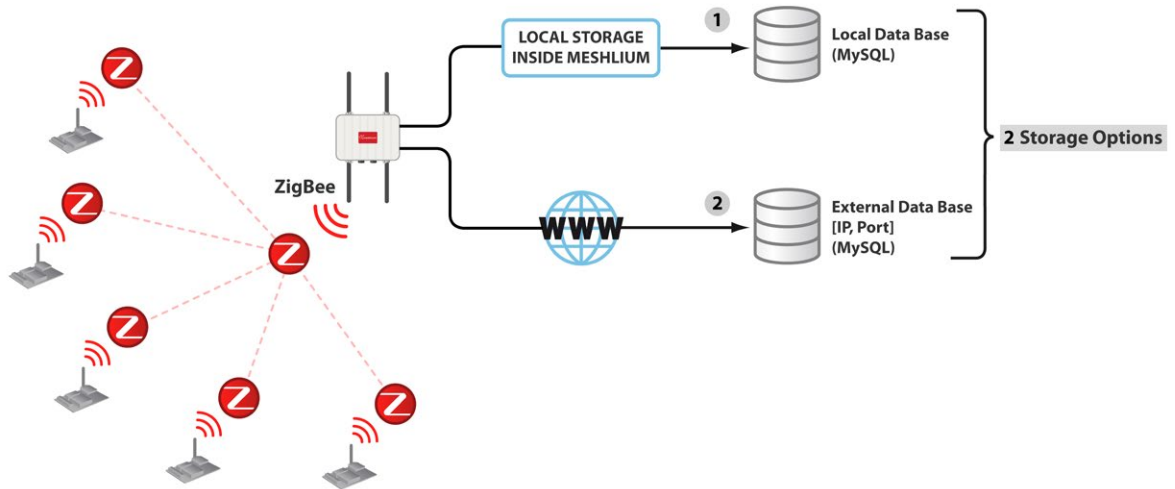
Meshlium receives the sensor data sent by Waspote using the ZigBee radio.

Then 4 possible actions can be performed:

1. Store the sensor data in the Meshlium Local Data Base (MySQL)
2. Store the ZigBee sensor data in an External Data Base (MySQL)
3. Send the information to the Internet using the Ethernet or WiFi connection
4. Send the information to the Internet using the 3G/GPRS connection

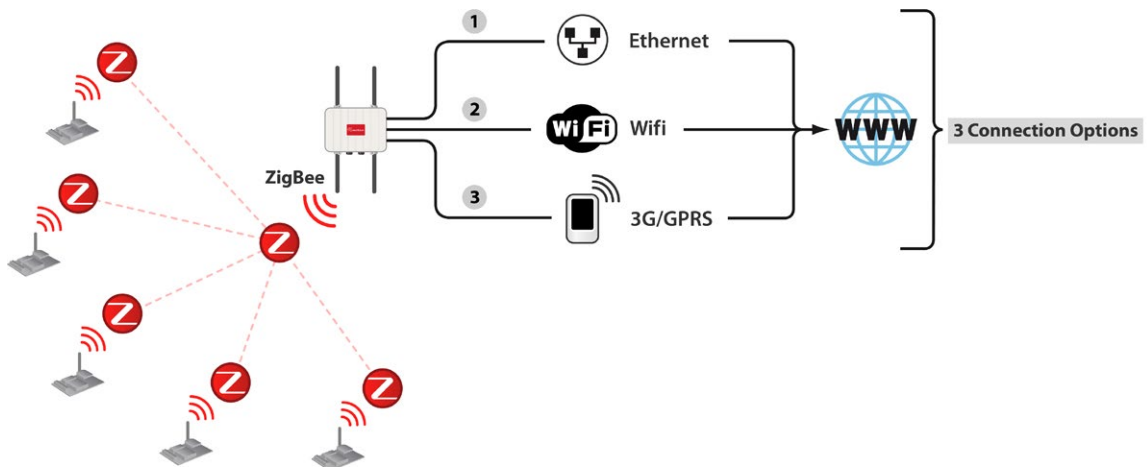


## • Meshlium Storage Options



- Local Data Base
- External Data Base

## • Meshlium Connection Options



- ZigBee → Ethernet
- ZigBee → WiFi
- ZigBee → 3G/GPRS

## • Capturing and storing sensor data in Meshlium from a Waspote sensor network

When you buy a kit containing Meshlium and Waspote, your Waspotes come configured to send frames to the Gateway. Later, once the user has developed the code for transmitting to the Gateway, he can switch to transmit to Meshlium.

Meshlium will receive the sensor data sent by Waspote using the ZigBee radio and it will store the frames in the Local Data Base. That can be done in an automatic way now thanks to the new **Sensor Parser**.

The **Sensor Parser** is a new feature for Meshlium (version 3.0.5 or older). It is a new software system which is able to do the following tasks in an easy and transparent way:

- receive frames from XBee (with the Data Frame format)
- parse these frames
- store the data in a local Database
- synchronize the local Database with an external Database

Besides, the user can add his own sensors.

The initial ZigBee frames sent by Waspote contain the next sequence:

```
~\0x00I\0x90\0x00}3\0xa2\0x00@z\0xcb\0x92\0xd8\0xd3\0x02<=>\0x80\0x03#35689722##7#ACC:80;10;987#IN_TEMP:22.50#BAT:93#\0xb4
```

Initially there are some hexadecimal characters, which belong to the XBee API frame, followed by the message. In the above example the message is:

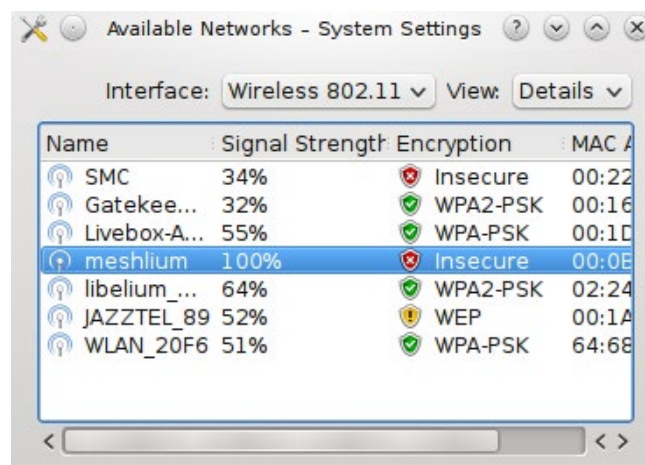
```
<=>\0x80\0x03#35689722##7#ACC:80;10;987#IN_TEMP:22.50#BAT:93#
```

They are formed by the accelerometer values, RTC internal temperature value, and battery level. The MAC address is added and other helpful information.

Meshlium comes with all the radios ready to be used. Just “plug & mesh!”. All the Meshlium nodes come with the WiFi AP ready so that users can connect using their WiFi devices. Connect the ethernet cable to your network hub, restart Meshlium and it will automatically get an IP from your network using DHCP \*.

(\*) For the Meshlium Mesh AP and for the Meshlium ZigBee Mesh AP the Internet connection depends on the GW of the network.

Then access Meshlium through the WiFi connection. First of all search the available access points and connect to “Meshlium”.



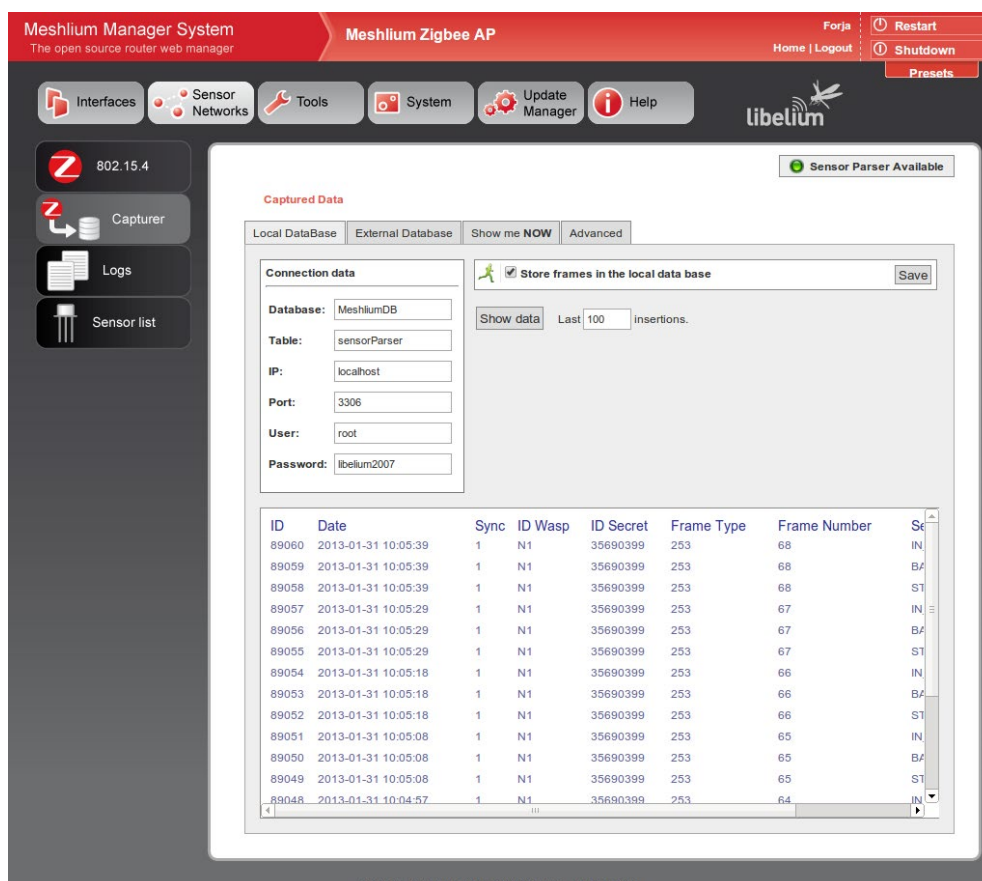
No password is needed as the network is public (you can change it later in the WiFi AP Interface options). When you select it, Meshlium will give an IP from the range 10.10.10.10 - 10.10.10.250.

Now you can open your browser and access to the Meshlium Manager System:

- **URL:** **http://10.10.10.1/ManagerSystem**
- **user:** root
- **password:** libelium



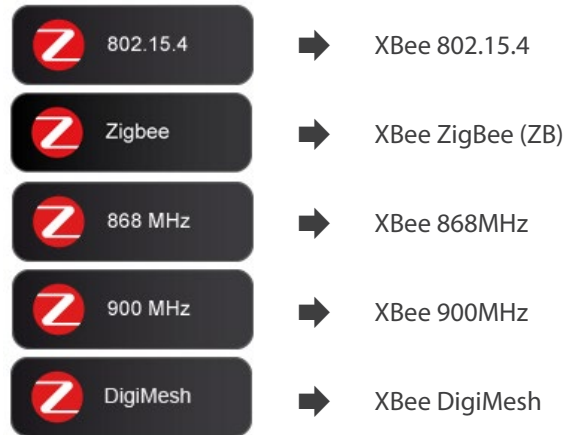
Now we go to the "Sensor Networks" tab.



The image shows the Meshlium Manager System interface, specifically the 'Sensor Networks' tab. The top navigation bar includes 'Interfaces', 'Sensor Networks', 'Tools', 'System', 'Update Manager', and 'Help'. The 'Sensor Networks' tab is active, showing a 'Captured Data' section. This section has tabs for 'Local DataBase', 'External Database', 'Show me NOW', and 'Advanced'. The 'Local DataBase' tab is selected, displaying a table of captured data. The table has columns for ID, Date, Sync, ID Wasp, ID Secret, Frame Type, Frame Number, and a status column. The data is sorted by ID in descending order.

ID	Date	Sync	ID Wasp	ID Secret	Frame Type	Frame Number	Status
89060	2013-01-31 10:05:39	1	N1	35690399	253	68	IN
89059	2013-01-31 10:05:39	1	N1	35690399	253	68	BA
89058	2013-01-31 10:05:39	1	N1	35690399	253	68	ST
89057	2013-01-31 10:05:29	1	N1	35690399	253	67	IN
89056	2013-01-31 10:05:29	1	N1	35690399	253	67	BA
89055	2013-01-31 10:05:29	1	N1	35690399	253	67	ST
89054	2013-01-31 10:05:18	1	N1	35690399	253	66	IN
89053	2013-01-31 10:05:18	1	N1	35690399	253	66	BA
89052	2013-01-31 10:05:18	1	N1	35690399	253	66	ST
89051	2013-01-31 10:05:08	1	N1	35690399	253	65	IN
89050	2013-01-31 10:05:08	1	N1	35690399	253	65	BA
89049	2013-01-31 10:05:08	1	N1	35690399	253	65	ST
89048	2013-01-31 10:04:57	1	N1	35690399	253	64	IN

There are 5 different XBee models which can be configured:



Depending on the kind of XBee model the parameters to be configured may vary.

Complete list:

- **Network ID:** Also known as PAN ID (Personal Area Network ID)
- **Channel:** frequency channel used
- **Network Address:** 16b address (hex field) - MY
- **Node ID:** maximum 20 characters (by default "Meshlium")
- **Power level:** [0..4] (by default 4)
- **Encrypted mode:** true/false (by default false)
- **Encryption Key:** 16 characters maximum
- **MAC:** 64b hardware address. It is a read only value divided in two parts:
  - MAC-high: 32b (hex field)
  - MAC-low: 32b (hex field)

These parameters must be also configured in the Wasmote sensor nodes. Access to all the information related to Wasmote at: <http://www.libelium.com/wasmote>

**DigiMesh**

Network ID:	<input type="text" value="3332"/>
Channel:	<input type="text" value="0x0E"/>
Node ID:	<input type="text" value="Meshlium"/>
Power Level:	<input type="text" value="2"/>
Encrypted mode:	<input type="text" value="Off"/>
Encryption key:	<input type="text"/>
MAC high:	<input type="text" value="13a200"/>
MAC low:	<input type="text" value="407791fc"/>

To discover the MAC address of the XBee module just press the “Load MAC” button.

The “Check status” option allows to see if the ZigBee radio is working properly and if the configuration stored on it matches the values set in the Manager System.



Both process ("Load MAC" and "Check status") require the ZigBee capturer daemon to be stopped. This means no frames will be received while executing this actions. Be patient this can take up to 1 minute to finish.

**DigiMesh**

Network ID:	<input type="text" value="3332"/>
Channel:	<input type="text" value="0x0E"/>
Node ID:	<input type="text" value="meshlium"/>
Power Level:	<input type="text" value="2"/>
Encrypted mode:	<input type="text" value="Off"/>
Encryption key:	<input type="text"/>
MAC high:	<input type="text" value="13a200"/>
MAC low:	<input type="text" value="407791fc"/>

Connecting to serial port ...  
**Connected.**

Network ID: **OK**  
Node ID: **OK**  
Power Level: **OK**  
Encrypted Mode: **OK**

**Note:** When you buy a WaspMote Developer kit with Meshlium and with the XBee ZB as ZigBee radio both the WaspMote GW and Meshlium come configured as Coordinator of the network. Take into account that only one of them can be working at the same time.

**Note:** If the encryption check fails but the rest of parameters are OK, it means the ZigBee radio has an old version of the firmware but it is working perfectly.



## • Capturing and storing sensor data

When you buy a kit containing Meshlium and Wasp mote, your Wasp motes come configured to send frames to the Gateway. Later, once the user has developed the code for transmitting to the Gateway, he can switch to transmit to Meshlium.

The initial ZigBee frames sent by Wasp mote contain the next sequence (XBee API frame characters are removed here):

```
<=>\0x80\0x03#35689722##7#ACC:80;10;987#IN_TEMP:22.50#BAT:93#
```

They are formed by the accelerometer values, RTC internal temperature value, and battery level. The MAC address is added and other helpful information.

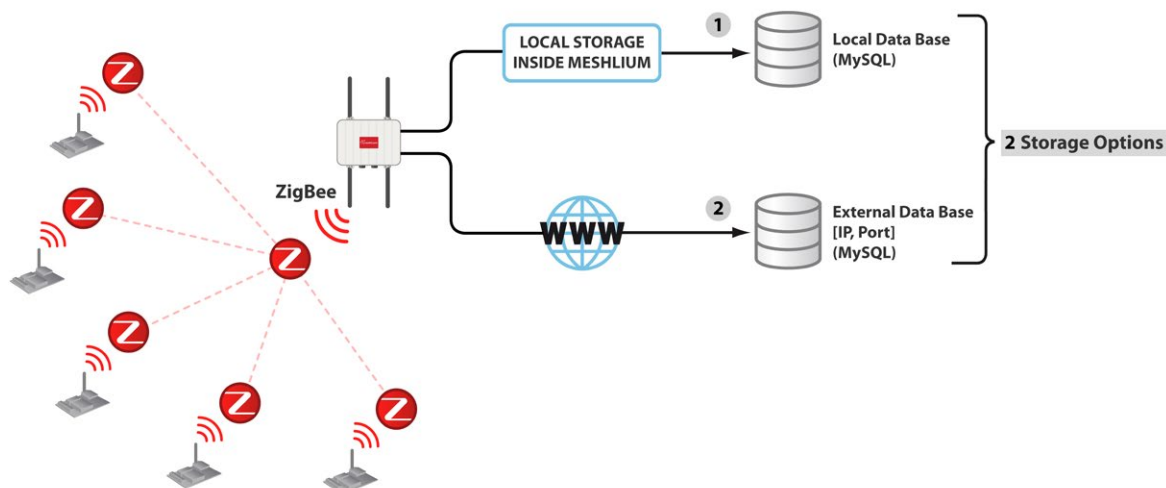
In order to add your own sensor frames properly go to the section "Sensors". However if you do not use the official Data Frame by Libelium, your data frames will be saved as a generic "Frame" in the database. See the picture below in order to see different frames types and how they are saved in the database.

ID	Date	Sync	ID Wasp	ID Secret	Frame Type	Frame Number	Se
87493	2013-01-31 08:33:38	0	N1	35690399	253	57	IN
87492	2013-01-31 08:33:38	0	N1	35690399	253	57	BA
87491	2013-01-31 08:33:38	0	N1	35690399	253	57	ST
87489	2013-01-31 08:33:27	0	<=>\0x80\0x03#35690399#N1#56#STR:XBee frame#BAT:90#IN_TE				
87488	2013-01-31 08:33:17	1	N1	35690399	253	55	IN
87487	2013-01-31 08:33:17	1	N1	35690399	253	55	BA
87486	2013-01-31 08:33:17	1	N1	35690399	253	55	ST
87485	2013-01-31 08:33:06	1	N1	35690399	253	54	IN
87484	2013-01-31 08:33:06	1	N1	35690399	253	54	BA
87483	2013-01-31 08:33:06	1	N1	35690399	253	54	ST
87482	2013-01-31 08:32:56	1	N1	35690399	253	53	IN
87481	2013-01-31 08:32:56	1	N1	35690399	253	53	BA
87480	2013-01-31 08:32:56	1	N1	35690399	253	53	ST

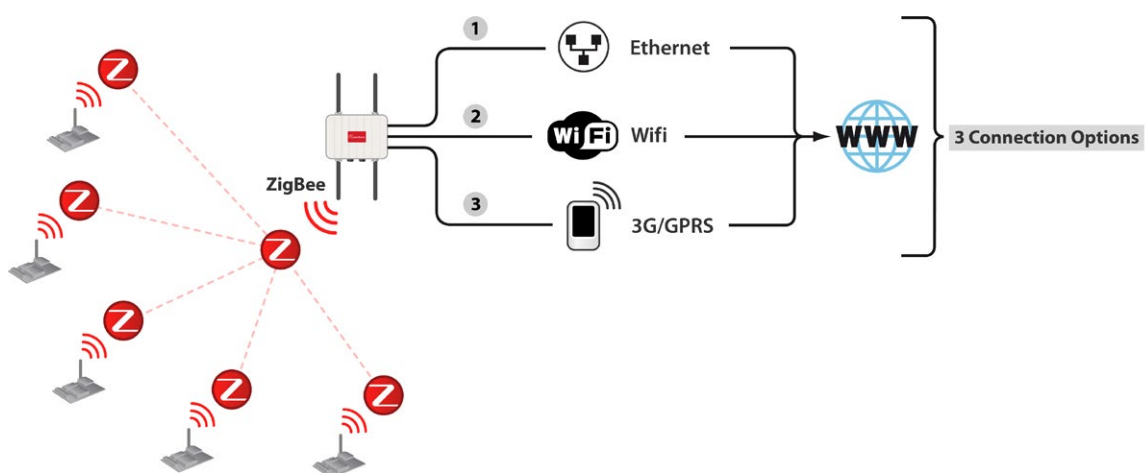
If you change any of the parameters in Wasp mote or Meshlium you will have to do it in both platforms so that they still can communicate.

We can perform two different storage options with the ZigBee frames captured:

- Local Data Base
- External Data Base



You can also send the information received to the Internet using the Ethernet, WiFi and 3G/GPRS interfaces.



## • Local Data Base

Meshlium has a MySQL data base up and running which is used to store locally the information captured. In the “Local Data Base” tab you can see the connection parameters.

- **Database:** MeshliumDB
- **Table:** sensorParser
- **IP:** localhost / 10.10.10.1 \*
- **Port:** 3306
- **User:** root
- **Password:** libelium2007

You can change the password, see the Users Manager section.

(\*) Depending on the parameters set in the Interfaces section.

**Captured Data**

Local DataBase External Database Show me NOW Advanced

**Connection data**

Database: MeshliumDB

Table: sensorParser

IP: localhost

Port: 3306

User: root

Password: libelium2007

☒ Store frames in the local data base

Last 100 insertions.

ID	Date	Sync	ID Wasp	ID Secret	Frame Type	Frame Number	Se
73650	2013-01-30 18:57:18	0	N1	35690399	253	29	IN
73649	2013-01-30 18:57:18	0	N1	35690399	253	29	BA
73648	2013-01-30 18:57:18	0	N1	35690399	253	29	ST
73647	2013-01-30 18:57:07	0	N1	35690399	253	28	IN
73646	2013-01-30 18:57:07	0	N1	35690399	253	28	BA
73645	2013-01-30 18:57:07	0	N1	35690399	253	28	ST
73644	2013-01-30 18:56:57	0	N1	35690399	253	27	IN
73643	2013-01-30 18:56:57	0	N1	35690399	253	27	BA
73642	2013-01-30 18:56:57	0	N1	35690399	253	27	ST
73641	2013-01-30 18:56:46	0	N1	35690399	253	26	IN
73640	2013-01-30 18:56:46	0	N1	35690399	253	26	BA
73639	2013-01-30 18:56:46	0	N1	35690399	253	26	ST
73638	2013-01-30 18:56:36	0	N1	35690399	253	25	IN

### Steps:

1. Set the check box “Store frames in the local data base” and press the “Save” button.

From this time Meshlium will automatically perform Scans and will store the results in the Local Data Base. This process will also continue after restarting Meshlium.

At any time you can see the last “x” records stored. Just set how many insertions you want to see and press the “Show data” button.

## • External Data Base

Meshlium can also store the information captured in an External Data Base.

### Steps:

1. Pressing the "Show sql script" you will get the code needed to create the data base along with the table and the right privileges.

**Captured Data**

Local DataBase External Database Show me **NOW** Advanced

**Connection data**

**Database:** ParserExternal


**Table:** zigbeeParser

**IP:** 192.168.1.6

**Port:** 3306

**User:** root

**Password:** root

 ☒ **Store frames in the external data base**

Synchronize each  seconds **Save**

**Show data** Last  insertions. **Show sql script** (to create database and table)

**Save** **Check Connection** **Synchronize Now**

**Just copy paste:**

```
CREATE database MeshliumDB;
```

**Just copy paste:**

```
CREATE TABLE IF NOT EXISTS `sensorParser` (
  `id` int(11) NOT NULL auto_increment,
  `id_wasp` text character set utf8 collate utf8_unicode_ci,
  `id_secret` text character set utf8 collate utf8_unicode_ci,
  `frame_type` int(11) default NULL,
  `frame_number` int(11) default NULL,
  `sensor` text character set utf8 collate utf8_unicode_ci,
  `value` text character set utf8 collate utf8_unicode_ci,
  `timestamp` timestamp NOT NULL default CURRENT_TIMESTAMP,
  `raw` text character set utf8 collate utf8_unicode_ci,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

2. Insert this code in your MySQL management application.

3. Fill the Connection Data fields with the information about where the data base is located (IP, Port) and with the authentication options (Database, Table, User, Password).

This data are stored in /mnt/lib/cfg/sensorExternalDB file.

4. Now press the "Check Connection" button to see if the configuration is correct.

**Captured Data**

Local DataBase External Database Show me NOW Advanced

**Connection data**

**Database:** ParserExternal


**Table:** zigbeeParser

**IP:** 192.168.1.6

**Port:** 3306

**User:** root

**Password:** root

 ☒ **Store frames in the external data base**

Synchronize each  seconds **Save**

**Show data** Last  insertions. **Show sql script** (to create database and table)

**Save** **Check Connection** **Synchronize Now**

Connecting to the database server ...

Selecting database ...

**OK**

5. Set the check box "Store frames in external database", you can defined the interval how often to synchronize the local database with external database and press the "Save" button.

From this time Meshlium will automatically perform Scans and will store the results in the External Data Base each . This process will also continue after restarting Meshlium.

You can also choose to sync when you want. Just press the “Synchronize Now” button.

### Captured Data

Local DataBase
External Database
Show me NOW
Advanced

**Connection data**

**Database:**

**Table:**

**IP:**

**Port:**

**User:**

**Password:**

Synchronizing...

☒ Store frames in the external data base

Synchronize each  seconds

(to create database and table)

ID	Date	ID Wasp	ID Secret	Frame Type	Frame Number	Serial
73848	2013-01-30 19:03:06	N1	35690399	253	62	IN_
73847	2013-01-30 19:03:06	N1	35690399	253	62	BAT
73846	2013-01-30 19:03:06	N1	35690399	253	62	STR
73845	2013-01-30 19:02:56	N1	35690399	253	61	IN_
73844	2013-01-30 19:02:56	N1	35690399	253	61	BAT
73843	2013-01-30 19:02:56	N1	35690399	253	61	STR
73842	2013-01-30 19:02:45	N1	35690399	253	60	IN_
73841	2013-01-30 19:02:45	N1	35690399	253	60	BAT
73840	2013-01-30 19:02:45	N1	35690399	253	60	STR
73839	2013-01-30 19:02:35	N1	35690399	253	59	IN_
73838	2013-01-30 19:02:35	N1	35690399	253	59	BAT
73837	2013-01-30 19:02:35	N1	35690399	253	59	STR
73836	2013-01-30 19:02:24	N1	35690399	253	58	IN_


At any time you can see the last “x” records stored. Just set how many insertions you want to see and press the “Show data” button.

**Captured Data**

Local DataBase
External Database
Show me **NOW**
Advanced

**Connection data**

Database: ParserExternal
Table: zigbeeParser
IP: 192.168.1.6
Port: 3306
User: root
Password: root


☒ Store frames in the external data base

Synchronize each 30 seconds
Save

Show data
Last 100 insertions.
Show sql script (to create database and table)

Save
Check Connection
Synchronize Now

ID	Date	ID Wasp	ID Secret	Frame Type	Frame Number	Serial
73593	2013-01-30 18:48:08	N1	35690399	253	233	IN_
73592	2013-01-30 18:48:08	N1	35690399	253	233	BAT
73591	2013-01-30 18:48:08	N1	35690399	253	233	STR
73590	2013-01-30 18:47:57	N1	35690399	253	232	IN_
73589	2013-01-30 18:47:57	N1	35690399	253	232	BAT
73588	2013-01-30 18:47:57	N1	35690399	253	232	STR
73587	2013-01-30 18:47:47	N1	35690399	253	231	IN_
73586	2013-01-30 18:47:47	N1	35690399	253	231	BAT
73585	2013-01-30 18:47:47	N1	35690399	253	231	STR
73584	2013-01-30 18:47:36	N1	35690399	253	230	IN_
73583	2013-01-30 18:47:36	N1	35690399	253	230	BAT
73582	2013-01-30 18:47:36	N1	35690399	253	230	STR
73581	2013-01-30 18:47:26	N1	35690399	253	229	IN_



- **Show me now!**

In the “Show me now!” tab you can see in real time the Scans captured.

You can specify if you want the information to be updated periodically with the defined interval just checking the “Use the Defined Interval” button.

**Captured Data**

Local DataBase	External Database	Show me <b>NOW</b>	Advanced Database
----------------	-------------------	--------------------	-------------------

Start Scan

☒ Use the defined Scan interval

Seconds

Clean

**ASCII frame**  
**Internal ID:**35690399 **Waspote ID:**N1 **Frame Type:**253 **Frame Number:**64  
**STR:**XBee frame  
**BAT:**93  
**IN\_TEMP:**29.75

**ASCII frame**  
**Internal ID:**35690399 **Waspote ID:**N1 **Frame Type:**253 **Frame Number:**62  
**STR:**XBee frame  
**BAT:**93  
**IN\_TEMP:**29.75

**ASCII frame**  
**Internal ID:**35690399 **Waspote ID:**N1 **Frame Type:**253 **Frame Number:**60  
**STR:**XBee frame  
**BAT:**93  
**IN\_TEMP:**29.75

**ASCII frame**  
**Internal ID:**35690399 **Waspote ID:**N1 **Frame Type:**253 **Frame Number:**58  
**STR:**XBee frame  
**BAT:**93  
**IN\_TEMP:**29.75

## • Advanced Database

In the “Advanced” tab you can see information about the state in which they are databases.

It displays information about the Loca and External database, showing the following information:

- Local and External Database names
- Local and External Database sizes
- Local and External Tables
- Total Local and External Entries
- Synchronized Local Frames
- Unsynchronized Local Frames

**Captured Data**

Local DataBase External Database Show me **NOW** Advanced

**Local Database**

Database: **MeshliumDB**  
 Database Size: **12.35 Mb**  
 Table: **sensorParser**  
 Entries: **900**  
 Synchronized Frames: **0**  
 Unsynchronized Frames: **900**

**Remove synchronized Data** **Remove ALL Content**

**External Database**

Database: **ParserExternal**  
 Database Size: **5.05 Mb**  
 Table: **zigbeeParser**  
 Entries: **72852**

**Logs Sync**

2013-01-30 17:48:50.257 - Synchronization OK  
 2013-01-30 17:49:20.157 - Synchronization OK  
 2013-01-30 17:49:50.218 - Synchronization OK  
 2013-01-30 17:50:20.077 - Synchronization OK  
 2013-01-30 17:50:50.327 - Synchronization OK  
 2013-01-30 17:51:20.088 - Synchronization OK  
 2013-01-30 17:51:50.187 - Synchronization OK  
 2013-01-30 17:52:24.039 - Synchronization OK  
 2013-01-30 17:52:53.808 - Synchronization OK

From this tab, **you can delete all the information contained in the Local database or Remove synchronized data**. Before performing these actions, a confirmation message will be displayed.

**Note:** Before running these options, it is recommended to have a backup or having synchronized your local database with external database.

**Captured Data**

Local DataBase External Database Show me **NOW** Advanced

**Local Database**

Database:	<b>MeshliumDB</b>
Database Size:	<b>13.50 Mb</b>
Table:	<b>sensorParser</b>
Entries:	<b>15301</b>
Synchronized Frames:	<b>15295</b>
Unsynchronized Frames:	<b>6</b>

Remove synchronized Data Remove ALL Content

**External Database**

2013-01-31 08:33:49.315 - Synchronization OK  
2013-01-31 08:34:19.401 - Synchronization OK  
2013-01-31 08:34:49.138 - Synchronization OK

**Mensaje de la página 192.168.1.103:**

Synchronized data of sensorParser table will be deleted.  
Do you want to continue?

Cancelar Aceptar

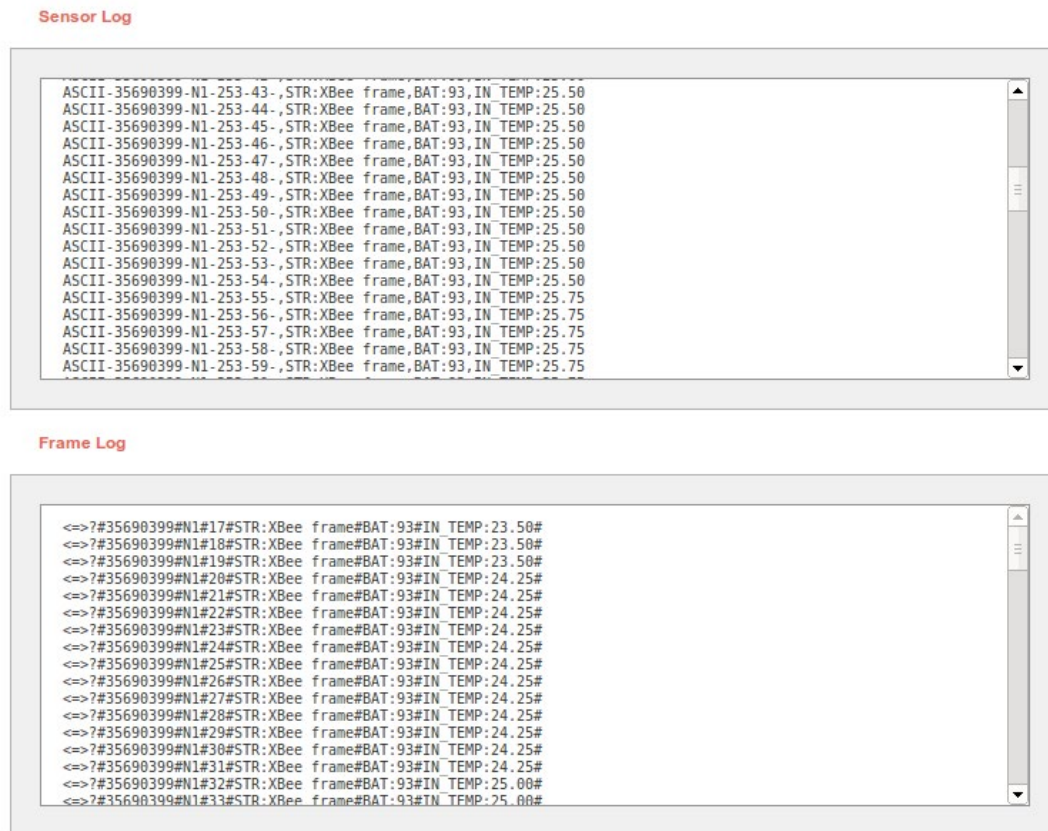
In addition can display a log of the date of the last synchronization between the local database and external database was successful.

## Logs Sync

2013-01-30 17:48:50.257 - Synchronization OK  
2013-01-30 17:49:20.157 - Synchronization OK  
2013-01-30 17:49:50.218 - Synchronization OK  
2013-01-30 17:50:20.077 - Synchronization OK  
2013-01-30 17:50:50.327 - Synchronization OK

## • Capturer logs

Inside “Sensor Networks” exist the section **Logs**, in this section you can see the last frames received on Meshlium.



First show the “sensor log”, in this logs shows the frames are stored after being processed.

```
ASCII-35690399-N1-253-198-,STR:XBee frame,BAT:93,IN_TEMP:31.50
```

Secondly shown “Frame Log”, in this logs shows the frames stored as the arrive to Meshlium.

```
<=>?#35690399#N1#198#STR:XBee frame#BAT:93#IN_TEMP:31.50#
```

## Sensors

In section "Sensor List", the user can **add new sensors or delete sensors**.

By default Meshlium recognize all Libelium official sensors frames. All sensors frames that Meshlium can capture and store must be specified in an XML file.

The file with official sensors of Libelium is located in `/mnt/lib/cfg/parser/sensors.xml`

The button "update sensors" update the Libelium official sensor. User sensors remaining unchanged.

Users can add and remove sensors in an easy and simple from ManagerSystem.

To add a new sensor the user must complete the fields:

- ASCII ID: sensor id for ASCII frame.
- Fields: This field specifies the number of sensor fields sent in the frame. This helps to calculate the frame length.
- Type: type of fields
  - uint8\_t
  - int
  - float
  - string
  - ulong
  - array(ulong)

Once all fields are filled in, click on the button "Add sensor"

Aviable Sensors

ASCII ID:

Fields:

Type:

Sensors Updated

Standard sensors

ID	ASCII ID	Fields	Type
0	CO	1	float
1	CO2	1	float
2	O2	1	float
3	CH4	1	float
4	LPG	1	float
5	NH3	1	float
6	AP1	1	float
7	AP2	1	float
8	SV	1	float
9	NO2	1	float
10	O3	1	float
11	VOC	1	float
12	TCA	1	float
13	TFA	1	float
14	HUMA	1	float
15	PA	1	float
16	PW	1	float
17	BEND	1	float
18	VBR	1	uint_8
19	HALL	1	uint_8
20	LP	1	uint_8
21	LL	1	uint_8
22	LUM	1	float
23	PIR	1	uint_8
24	ST	1	float
25	MCP	1	uint_8
26	CDG	1	uint_8

User sensors


	ID	ASCII ID	Fields	Type
	200	AGM	9	uint_8

The new user sensors will be added to the new XML file, the file with user sensors is located in `/mnt/lib/cfg/parser/user_sensors.xml`

**Note:** In "Waspote data frame guide" document is located more extensive information about how to build the frame.

To delete sensor the user must press the garbage can that appears to the left of the description of the sensor. To complete the action should accept a confirmation message.

## User sensors

	ID	ASCII ID	Fields	Type
	200	AGM	9	uint_8

## • Sending ZigBee frames from Meshlium to Wasp mote

Meshlium can also send ZigBee frames to the Wasp mote nodes. In order to use this feature you have to stop the “capturing and storing” daemon which is running in the system.

To do so access by SSH to Meshlium and stop the default ZigBee daemon::

```
$ /etc/init.d/ZigbeeScanD.sh stop
```

Now you can execute the ZigBeeSend command. There are several ways to send information to a node:

- Using its 802.15.4 MAC address (64b)
- Using its Network address (MY) (16b)
- Performing a broadcast transmission

### Sending to Wasp mote using its MAC address (64b):

```
$ ./ZigBeeSend -mac 0013a2004069165d "Hello Wasp mote!"
```

### Sending to Wasp mote using its Net address (MY - 16b):

```
$ ./ZigBeeSend -net 1234 "hello Wasp mote!"
```

### Send to all the Wasp mote devices at the same time - Broadcast mode:

```
$ ./ZigBeeSend -b "hello everybody!"
```

The source code “ZigbeeSend.c” and the reception program to be installed in Wasp mote can be downloaded from the Meshlium Development section: <http://www.libelium.com/development/meshlium>

You can download these files and change them in order to get new features and sending options.

### Compilation:

The compilation can be done in the same Meshlium. Just copy these files in a folder accessing by SSH and execute:

```
$ gcc -o ZigBeeSend ZigBeeSend.c -lpthread
```

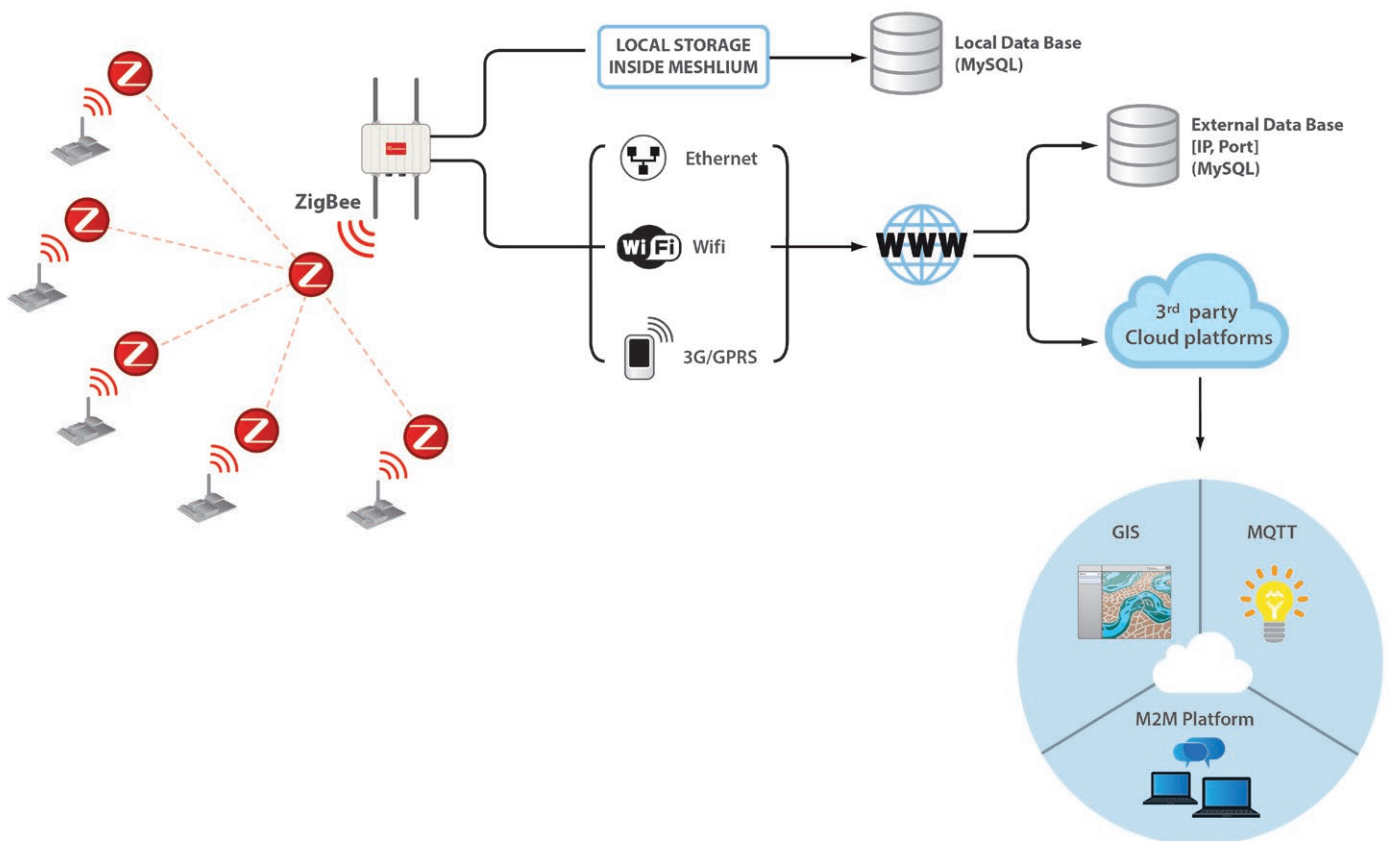
**Important:** If you want to create a “ZigBee sending” daemon that is executed each time Meshlium starts you have to deactivate the “ZigBee Capturer” daemon (/etc/init.d/ZigbeeScanD.sh) as the ZigBee radio has to be used by one process at a time.

You will find support in the Libelium Forum at: <http://www.libelium.com/forum>

## • Interacting with 3rd party Cloud platforms

Libelium has partnered with the best Cloud software solution providers to offer you all the necessary components to deploy Internet of Things (IoT), machine-to-machine (M2M) or Smart Cities projects with minimum time-to-market. Meshlium is ready to send sensor data to many Cloud software platforms. Just select the most suitable for you, get an account from the provider and configure your Meshlium. To get a list of the available Cloud platforms, see the section “Cloud Connector” of the Meshlium Technical Guide here:

<http://www.libelium.com/development/meshlium/documentation/meshlium-technical-guide/>



Cloud connector diagram