

LoRaWAN with custom server

The free offer for a LoRaWAN server usually consists of only one network with up to 10 devices per registration. The purpose of this tutorial is to show, that the Network server of the LoRaWAN network can be quite simple and running on a local PC.

This tutorial follows a selected open-source server:

<https://gotthardp.github.io/lorawan-server>

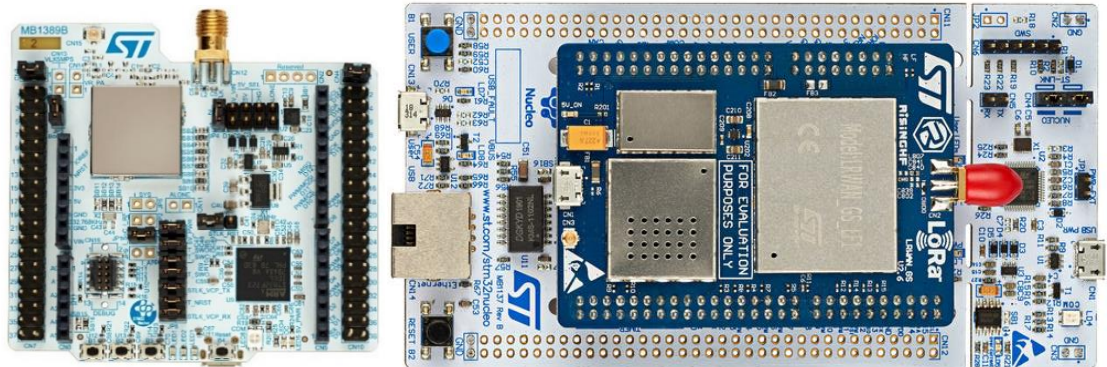
It acts as a Network server with an Application layer to handle e.g. join procedure, but the user Application layer – to reply to coming data – must be handled elsewhere. The server implementation offers a plugin to Semtech mote frames or handlers and connectors to external application. In comparison to Lorient or TTN, the main benefit is that this solution has no limit in terms of number of devices – the only limit is the hardware running the server.

The typical Application server is the Cayenne myDevices (both Lorient and TTN can connect to Cayenne), but to show a real simple custom application, it is possible to communicate with the LoRaWAN server using MQTT connection to a Node Red instance acting as application server user layer.

Requirements

Hardware

- A PC or laptop, in the tutorial Windows installation is shown but many Linux distributions are supported as well as Mac OS.
- A packet LoRa forwarder, in this case the STM32F7 Nucleo from the P-NUCLEO-LRWAN2 package is used, but any other packet forwarder should work.
- An end node, in this case STM32WL, but any LoRaWAN node is possible to use, e.g. from the P-NUCLEO-LRWAN2 package.



Software

The following versions of the software are proven to work together, it is possible to use newer versions, but some problems may arise.

- lorawan-server v0.6.7 - <https://github.com/gotthardp/lorawan-server/releases>
- Erlang OTP v23.2 - <http://erlang.org/download/>
- Node.js v12.17.0 - <https://nodejs.org/en/download/releases/>
- Node Red v1.0.6 - downloaded using npm
- Wget 1.11.4 - <http://gnuwin32.sourceforge.net/packages/wget.htm>

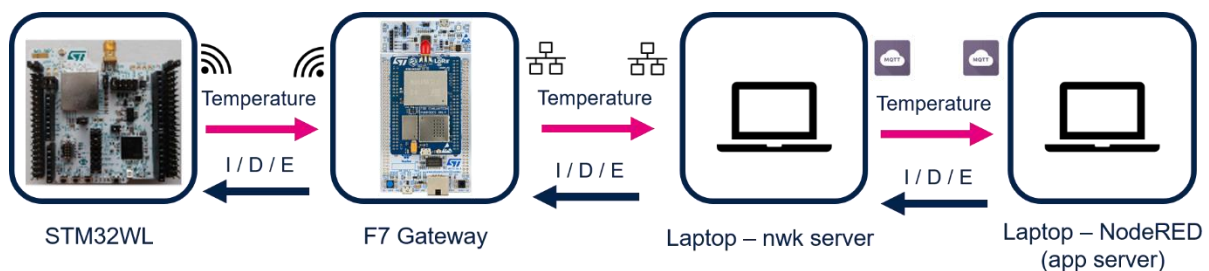
Model application

For this tutorial, to show the main benefit of an open source solution, it is possible to design a fully custom application.

The Application layer in this tutorial consists of a simple temperature controller, the node sends temperature in °C and the server responds with (I)ncrease, (D)ecrease, (E)qualized temperature against a certain threshold.

There are two ways to address the implementation of the (custom) application layer:

- **(optional)** Modified Semtech mote – which is the default handler for frames in the server implementation (simplified from mote frames to recognize only the temperature data and reply with I,D,E).
- Connect to the Nwk server using MQTT and HTTP and use external Application server, running on Node Red. The first approach is simpler, but one needs to compile the Erlang solution to modify the application. The second one, with Node Red, is very simple in modifying the application. With the software versions mentioned in the beginning, it is proven to work on most machines but in case of problems (connection via MQTT, more configuration steps, etc.), optional path is given.



The temperature measured on the End node ([STM32WL Nucleo](#) – running a simplified End_Node example from [Cube firmware package](#)) is propagated through the LoRaWAN network. From the node over the radio to a Gateway (STM32F7 with Rising HF module from [LRWAN2 pack](#)), then through the Ethernet connection to a Network server and finally it is either handled by the modified Semtech mote plugin of the server, or as in the diagram above, using a dedicated Node-Red server using MQTT connection.

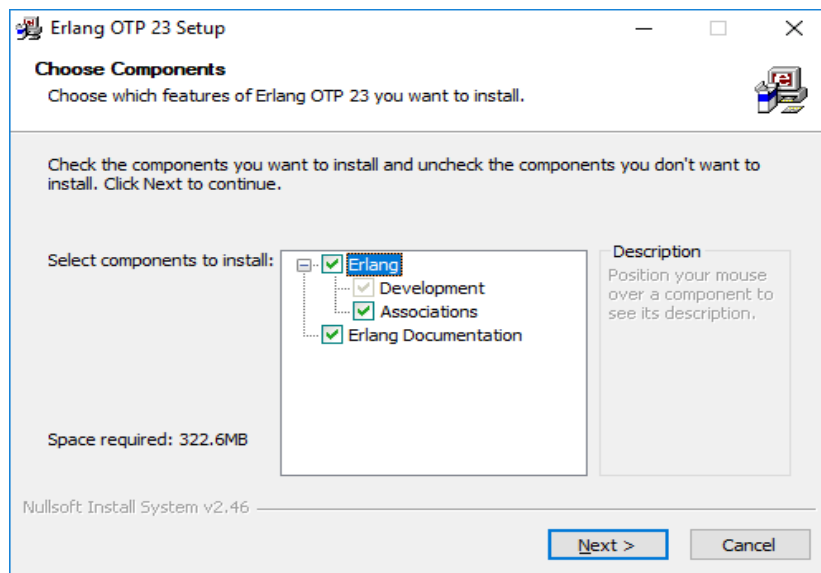
The network server installation

Installation steps

For the Nwk server only two things are needed. First, support of [Esl-Erlang OTP](#) (Windows x64 platform) is needed. Then the [LoraWan Server](#) release from Github.

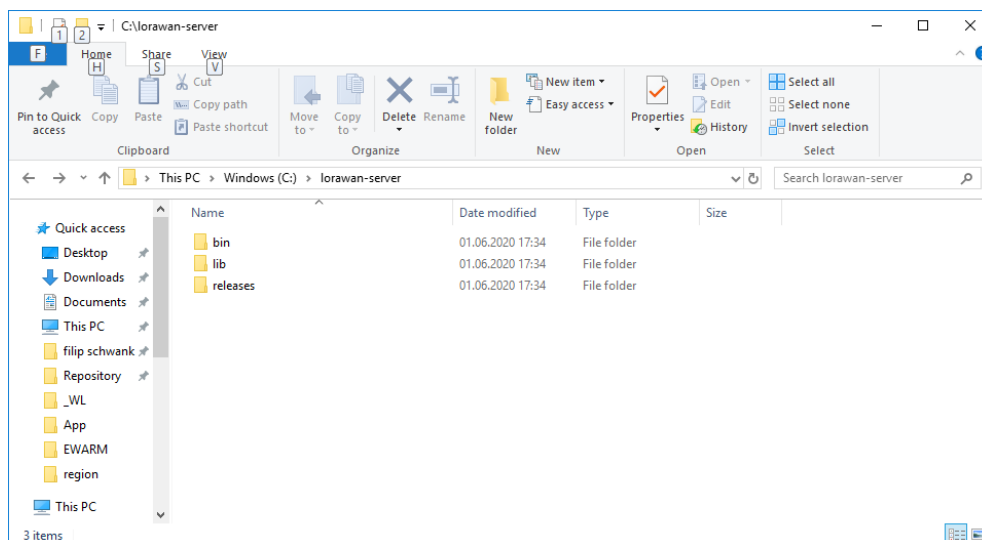
1. Esl-Erlang OTP

The installation of the Erlang is a simple “wizard” procedure. All settings can be left as default. On Windows 7 it may ask to install Microsoft Visual C++ Redistributable package.



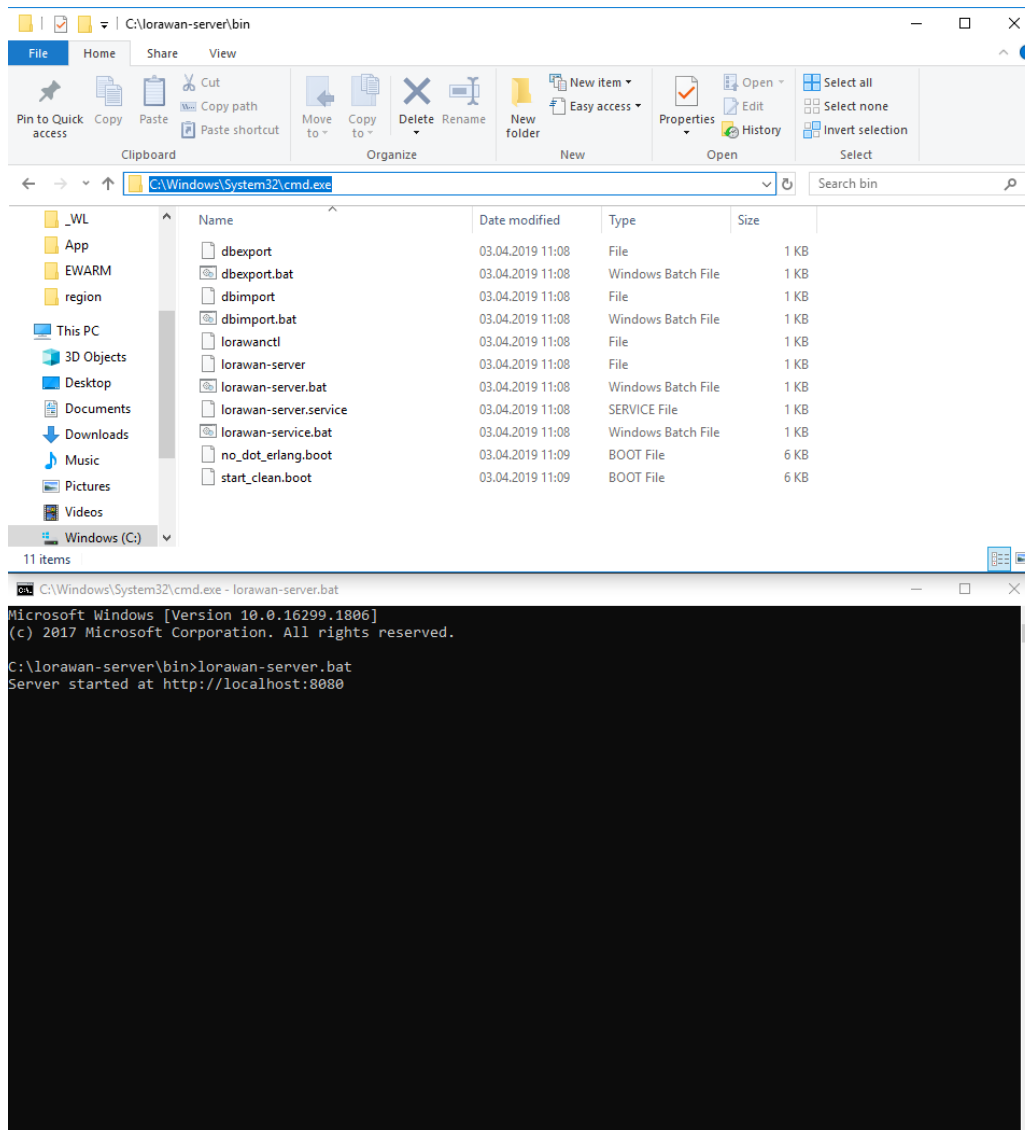
2. LoraWan Server

This is an archive file, unpack the zip to: **C:\lorawan-server**. After unpacking, the Nwk server folder structure should look like this:

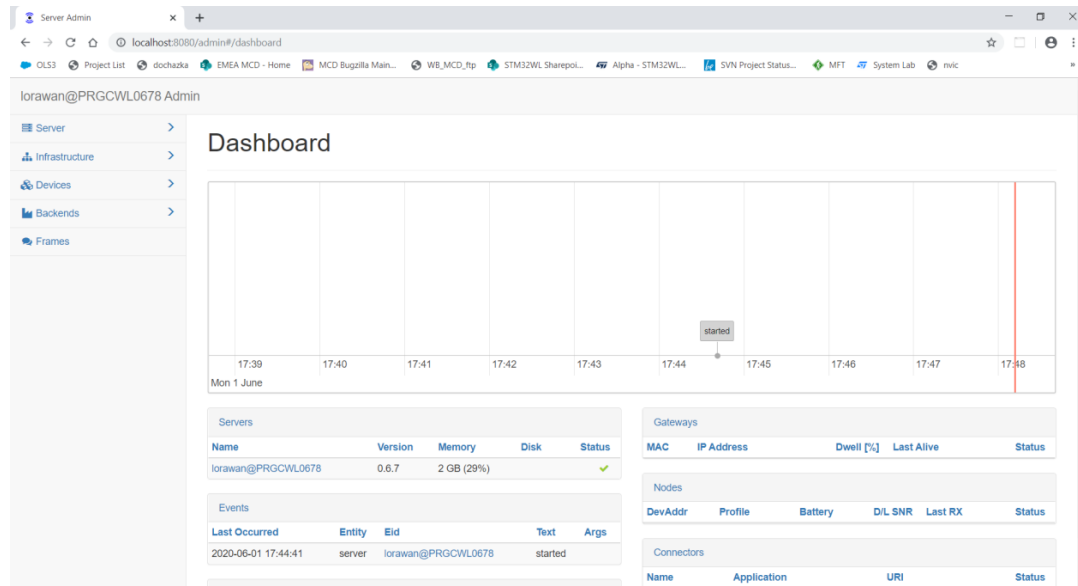


Next, it should be possible to run the server. Go to **C:\lorawan-server\bin** folder and write '**cmd**' into the folder address. Write in "**lorawan-server.bat**". The server is running now, **do not** close this window.

Avoid using a 32-bit(x86) application to launch the command-line and therefore the server. 32-bit application has different constants definition in the command-line and the .bat file will not work.

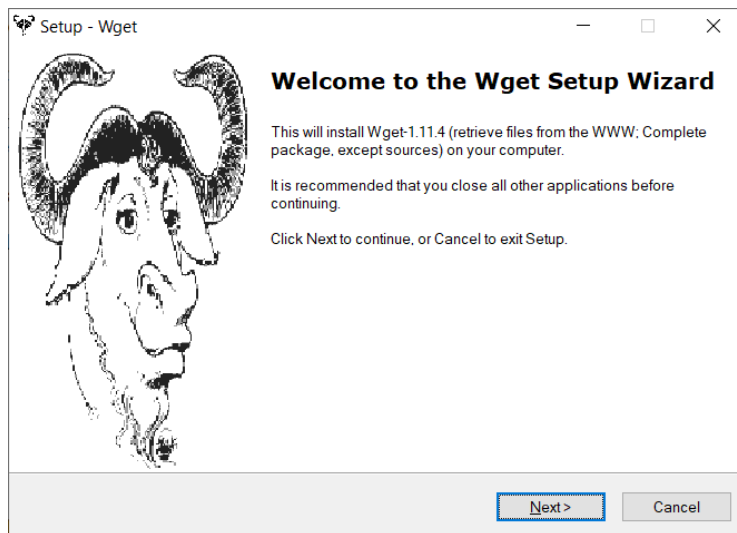


Now the http server administration has started alongside with the Nwk server and it is possible to login to **<http://localhost:8080/>** using username "**admin**" and psw "**admin**".

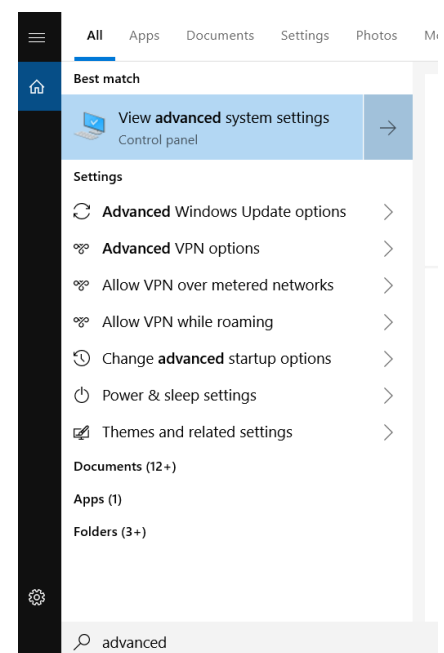
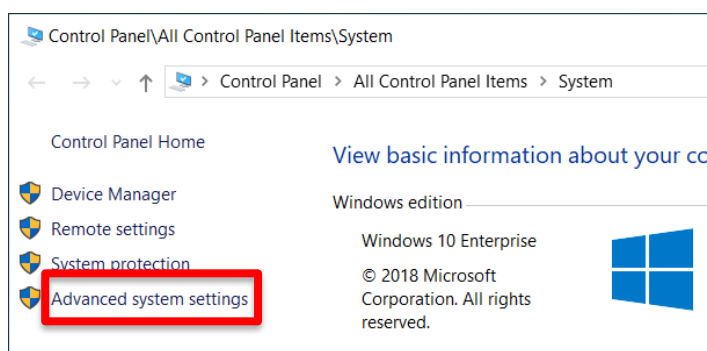


3. Wget for Windows

Wget in this tutorial is used to simplify the configuration process. The server installation contains a script to import the database from a json file. The script is using wget to access the HTTP API of the server and since Windows do not support wget natively, this package is used. Wget installation is again a simple procedure, only with addition that to make wget work, the environmental variable Path must be modified.

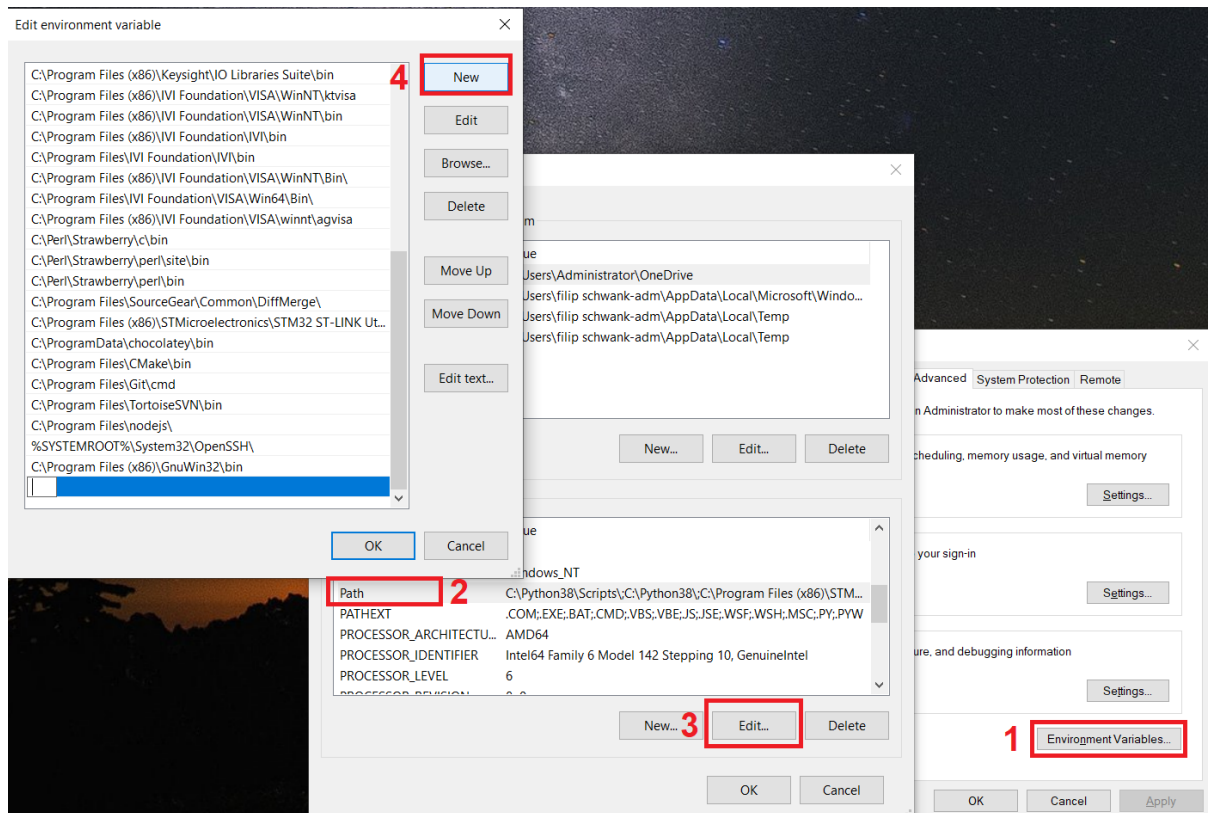


- Search for **Advanced system settings** or access **This PC** using the explorer, on **Windows 7** open **System information**, on **Windows 10** click **Computer** tab – **System properties** – **System info** on the top right. Then **Advanced system settings** on the left.



- Click on **Environment Variables(1)**, find **Path(2)** in **System variables**. Click **Edit(3)** and finally add a **New** entry with:

C:\Program Files (x86)\GnuWin32\bin



- Test wget by opening command line and entering **wget**

```
C:\WINDOWS\system32\cmd.exe

C:\>wget
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
wget: chybí URL
Použití: wget [PŘEPÍNAČ] [URL]

Příkaz "wget --help" vypíše další přepínače.

C:\>
```

- Missing URL is OK, only “ 'wget' is not recognized as an internal or external command, operable program or batch file.” would mean an error.

The network server configuration

With the server successfully installed, it is possible to add Lora devices and networks. The configuration of the server itself can be kept as is, in default values. For LoRaWAN functionality with the node a network must be created. Resp. several straightforward steps must be taken:

1. Importing configuration

The server supports database import, in the installation folder scripts can be found to do so. **wget** on Windows is required to run the **dbimport.bat** script from **C:\lorawan-server\bin**

This script is simplifying the process of manual configuration described below – using the provided folder with **.json** files **import_me_to_server**. Even after importing all the configuration, one must change his own Gateway ID (as described below in “**Configuration of the STM32F7 Gateway**”) and eventually a device (section “**Adding the actual devices**”). But this process largely simplifies the configuration.

It is also good to check that the configuration was set correctly by following the manual configuration below, but only by checking that the parameters were set as in the description.

- The import script contains a small mistake, as its origins are in Linux systems, so modify the **dbimport.bat** by **removing** the **echo** so only **wget** command will be left.

```
for %%F in (%DIR%\*.json) do (
    wget -nv %AUTH% --header=Content-Type:application/json --post-file=%%F http://%SERVER%/api/%%~nF
)
```


- With the server running on **localhost:8080**, run second command line from the same **C:\lorawan-server\bin** folder and run the command **dbimport.bat** “**path to json files**” such as:


dbimport.bat C:\LoRaWAN_custom_server\resources\import_me_to_server

- The import will take a short while, after that the output of the console should look like this:

```
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:00 URL:http://localhost:8080/api/gateways [0] -> "gateways" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:05 URL:http://localhost:8080/api/groups [0] -> "groups" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:10 URL:http://localhost:8080/api/handlers [0] -> "handlers" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:15 URL:http://localhost:8080/api/ignored_nodes [0] -> "ignored_nodes" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:20 URL:http://localhost:8080/api/multicast_channels [0] -> "multicast_channels" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:25 URL:http://localhost:8080/api/networks [0] -> "networks" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:30 URL:http://localhost:8080/api/nodes [0] -> "nodes" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:36 URL:http://localhost:8080/api/profiles [0] -> "profiles" [1]
SYSTEM_WGETRC = c:/progra~1/wget/etc/wgetrc
syswgetrc = C:\Program Files (x86)\GnuWin32/etc/wgetrc
2021-01-22 18:14:41 URL:http://localhost:8080/api/users [0] -> "users" [1]
```

- Back on the server, the configuration should be done and e.g. the template Gateway should appear.

Gateways				
MAC	IP Address	Dwell [%]	Last Alive	Status
DEADBEEFDEADBEEF				

<input type="checkbox"/> MAC	Area	Description	IP Address	Dwell [%]	Last Alive	↑ Status
<input type="checkbox"/> DEADBEEFDEADBEEF	TestArea	Change my MAC				

- The server is now configured for the Node Red application server with every setting that is required and should be the same on most machines, except for the unique IDs of Gateway and optionally an End node device (which is registered also by the Node Red server, but may be added manually as well – for the Semtech mote option it is required).
- In any case, please follow the manual instructions as well and compare that everything is set correctly – especially the section with firewall is important, as it may cause the most trouble.

2. Create a new area and its administrator (Infrastructure -> Areas)

Create new area

List

Name *

TestArea

✓

Administrators

admin ✕

✓

Slack Channel

Log Ignored?

true ✕ ▾

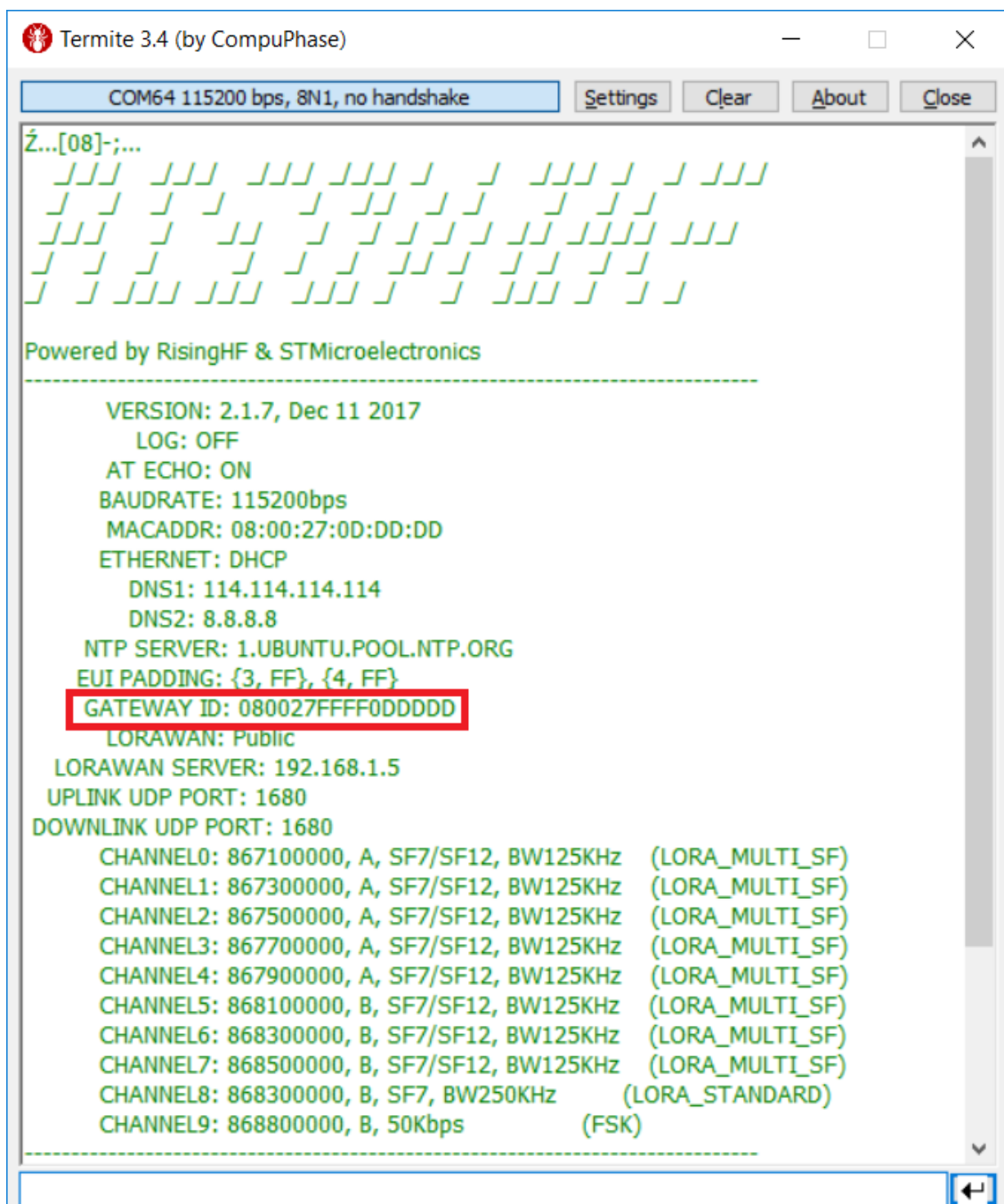
✓ Submit

3. Create a new gateway (Infrastructure -> Gateways)

To be able to configure the gateway, information about the MAC of the device is required.

To get it, simply plug in the micro USB cable to the blue extension board to power the GW and another micro USB cable to the ST-Link side. It should be detected as a **ST-Link virtual COM port**. The driver should be installed automatically – only on Windows 7 machines it requires a [driver](#). The connection parameters are **115200bps, 8bits, 1 stop bit, no parity**. It is possible to use any com port terminal, like Termit, or TeraTerm. For more details, please refer to [UM2587](#) for P-NUCEO-LRWAN2.

After powering the GW, or toggling the black Reset button, something like this should appear:



The **GATEWAY ID** is what must be entered in the **MAC** box.

Create new gateway

[List](#)

General

MAC *

080027FFFF0DDDDD ✓

Area

TestArea ✕ ✓

TX Chain *

0

Antenna Gain (dBi)

e.g. 6

Description

Location *

4. Ethernet connection

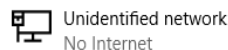
The Gateway can be anywhere in your local network, what is important is that the GW has an assigned IP address and that the LoraWAN server IP is pointing to your PC hosting the server. In this tutorial, direct wire connection to a laptop is used, so a **static network** must be defined in the Internet setting in Windows.

If the GW is connected to a router with **DHCP** IP assignment, you can go to the chapter "**Configuration of the STM32F7 Gateway**". But it is also recommended to use address reservation feature present on most routers. Address reservation will prevent that the PC running the server will always have the same IP and reconfiguration on the Gateway side is then not required. Reserving PC IP address is recommended, Gateway IP reservation is optional.

- In the system tray right click on your internet connection, Network & Internet settings, Change adapter options.

Ethernet

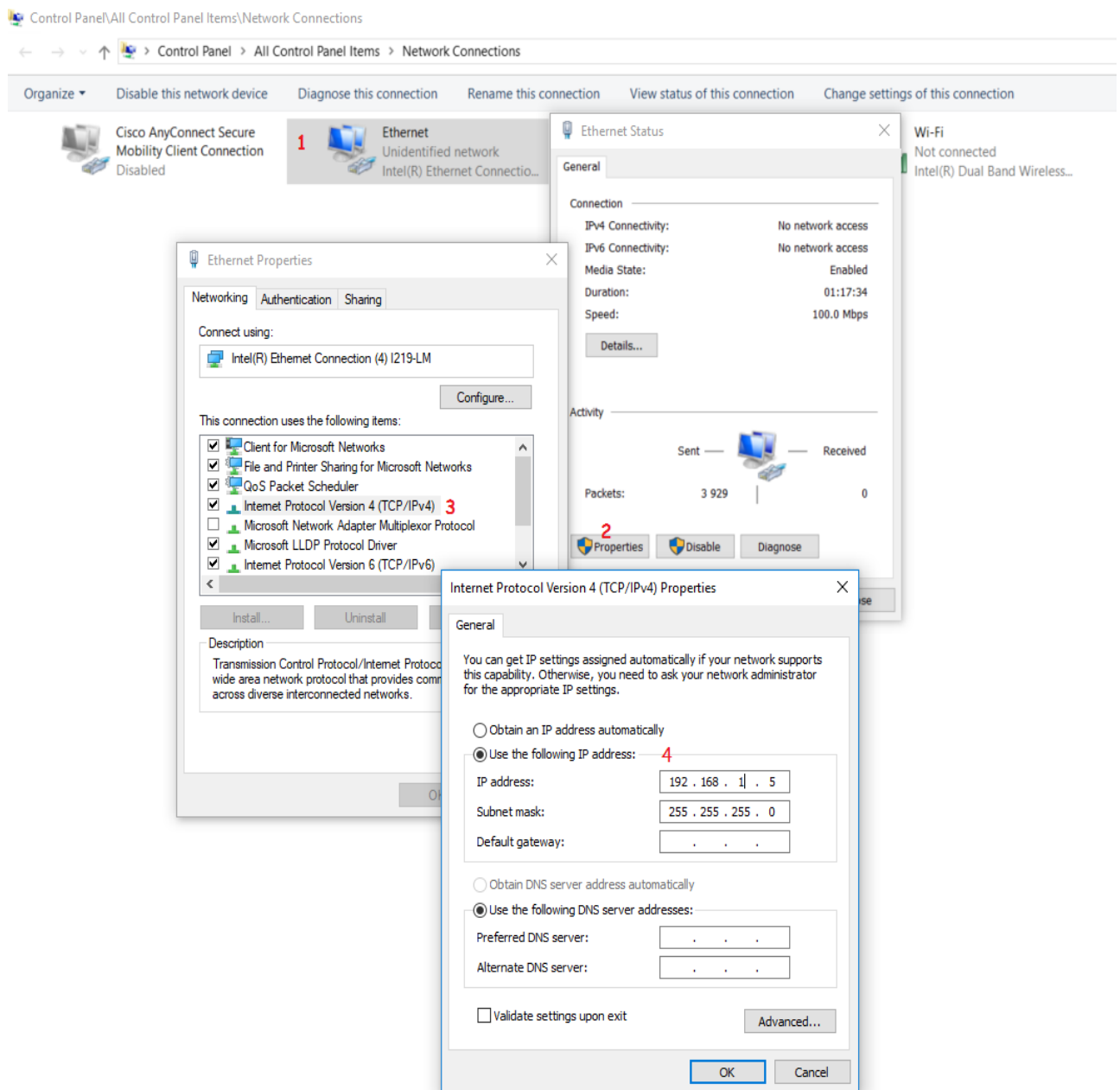
Ethernet



Related settings

[Change adapter options](#)[Change advanced sharing options](#)[Network and Sharing Center](#)[HomeGroup](#)[Windows Firewall](#)

- 1. Adapter options, 2. Open its properties, 3. Open IPV4 settings, 4. Set IP, e.g. **192.168.1.5** for the PC, with subnet mask **255.255.255.0**



- Check correct IP assignment – open command line (cmd.exe) and enter:
ipconfig -all

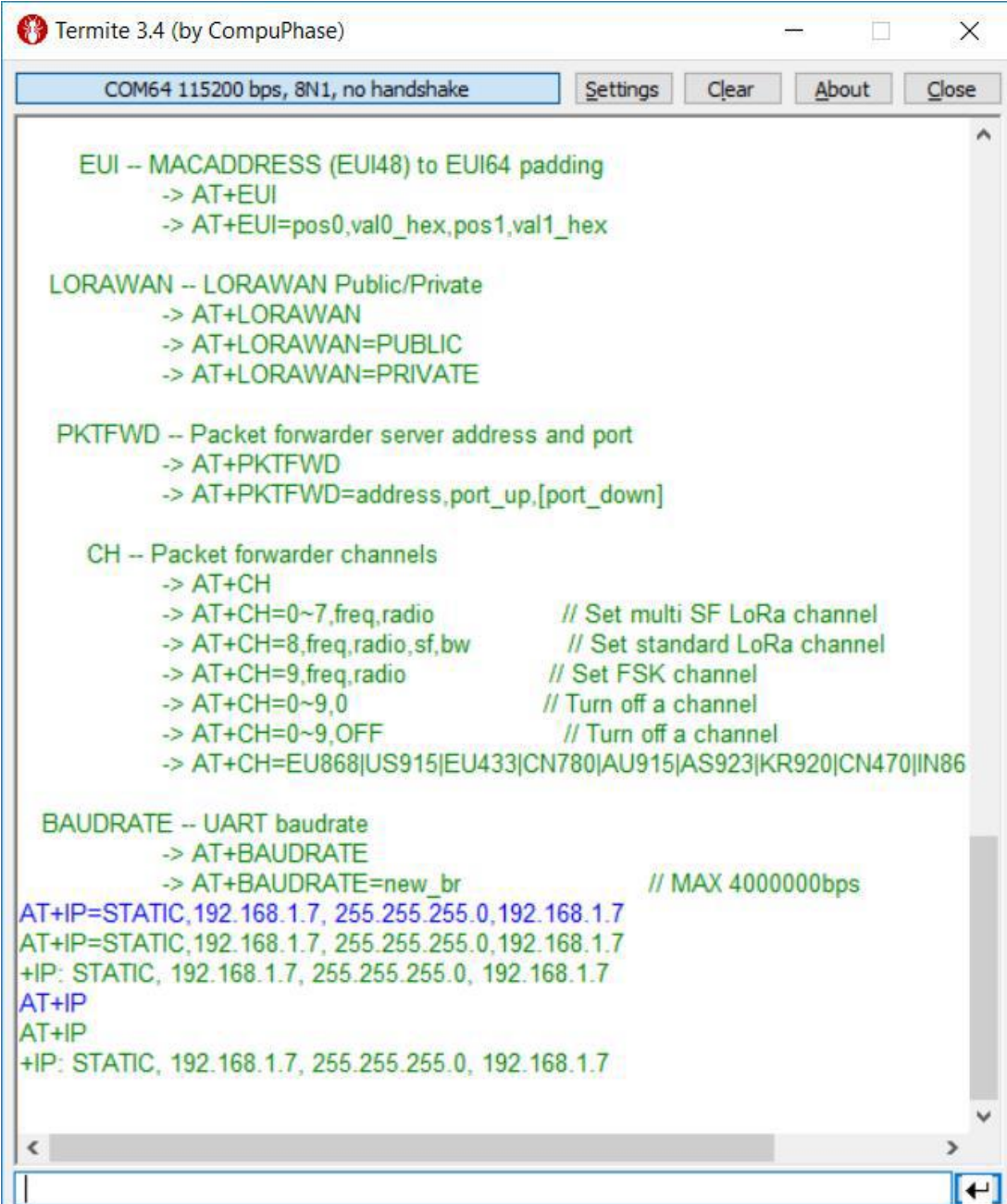
```
Ethernet adapter Ethernet:

Connection-specific DNS Suffix . : 
Description . . . . . : ASIX AX88179 USB 3.0 to Gigabit Ethernet Adapter
Physical Address. . . . . : 00-0E-C6-E2-14-B5
DHCP Enabled. . . . . : No
Autoconfiguration Enabled . . . . : Yes
Autoconfiguration IPv4 Address. . : 169.254.132.232(Preferred)
Subnet Mask . . . . . : 255.255.0.0
Default Gateway . . . . . : 192.168.1.7
NetBIOS over Tcpip. . . . . : Enabled
```

- If there is a different address from the one entered (such as **169.254.132.232** in the image above), it is probably in use by different adapter and cannot be assigned. One must choose a different address, e.g. by trying +1 increment 192.168.1.6, and repeat the process if necessary.

5. Configuration of the STM32F7 Gateway

Open again the Gateway terminal and send AT+HELP for the list of commands. One of the commands is **AT+IP=STATIC,ip,netmask,gateway**, which we can set in following way: **AT+IP=STATIC,192.168.1.7, 255.255.255.0,192.168.1.7**

The image shows a screenshot of the Termit 3.4 terminal window. The title bar reads "Termit 3.4 (by CompuPhase)". Below the title bar is a status bar showing "COM64 115200 bps, 8N1, no handshake" and buttons for "Settings", "Clear", "About", and "Close". The main text area contains a list of AT commands and their usage, color-coded in green. The commands listed are: EUI (MACADDRESS), LORAWAN (Public/Private), PKTFWD (Packet forwarder server address and port), CH (Packet forwarder channels), and BAUDRATE (UART baudrate). At the bottom of the list, the command **AT+IP=STATIC,192.168.1.7, 255.255.255.0,192.168.1.7** is shown in blue. Below this, the response **+IP: STATIC, 192.168.1.7, 255.255.255.0, 192.168.1.7** is shown in green. The terminal window has a scrollbar on the right and a command input line at the bottom with a cursor.

```
Termit 3.4 (by CompuPhase)
COM64 115200 bps, 8N1, no handshake
Settings Clear About Close

EUI -- MACADDRESS (EUI48) to EUI64 padding
-> AT+EUI
-> AT+EUI=pos0,val0_hex,pos1,val1_hex

LORAWAN -- LORAWAN Public/Private
-> AT+LORAWAN
-> AT+LORAWAN=PUBLIC
-> AT+LORAWAN=PRIVATE

PKTFWD -- Packet forwarder server address and port
-> AT+PKTFWD
-> AT+PKTFWD=address,port_up,[port_down]

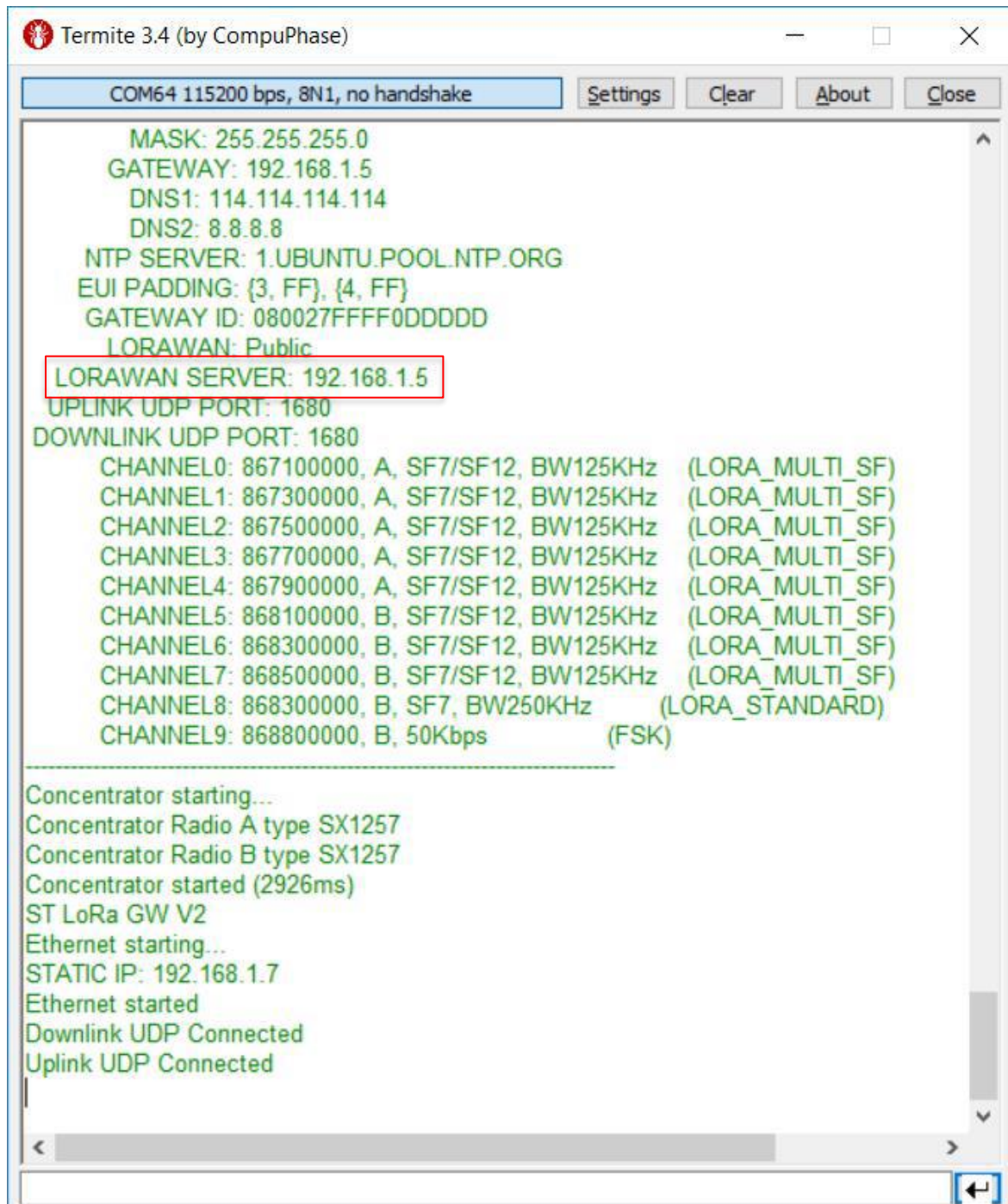
CH -- Packet forwarder channels
-> AT+CH
-> AT+CH=0~7,freq,radio // Set multi SF LoRa channel
-> AT+CH=8,freq,radio,sf,bw // Set standard LoRa channel
-> AT+CH=9,freq,radio // Set FSK channel
-> AT+CH=0~9,0 // Turn off a channel
-> AT+CH=0~9,OFF // Turn off a channel
-> AT+CH=EU868|US915|EU433|CN780|AU915|AS923|KR920|CN470|IN86

BAUDRATE -- UART baudrate
-> AT+BAUDRATE
-> AT+BAUDRATE=new_br // MAX 4000000bps
AT+IP=STATIC,192.168.1.7, 255.255.255.0,192.168.1.7
+IP: STATIC, 192.168.1.7, 255.255.255.0, 192.168.1.7
AT+IP
+IP: STATIC, 192.168.1.7, 255.255.255.0, 192.168.1.7
```

In the case that the GW is connected directly to a local router, following command can be used to let the router assign the IP:

AT+IP=DHCP

To point the GW to the server, use command **AT+PKTFWD=address,port_up,[port_down]**, in this case: **AT+PKTFWD=192.168.1.5,1680,1680**. And then reset the device. It should be able to establish a UDP connection with the server.



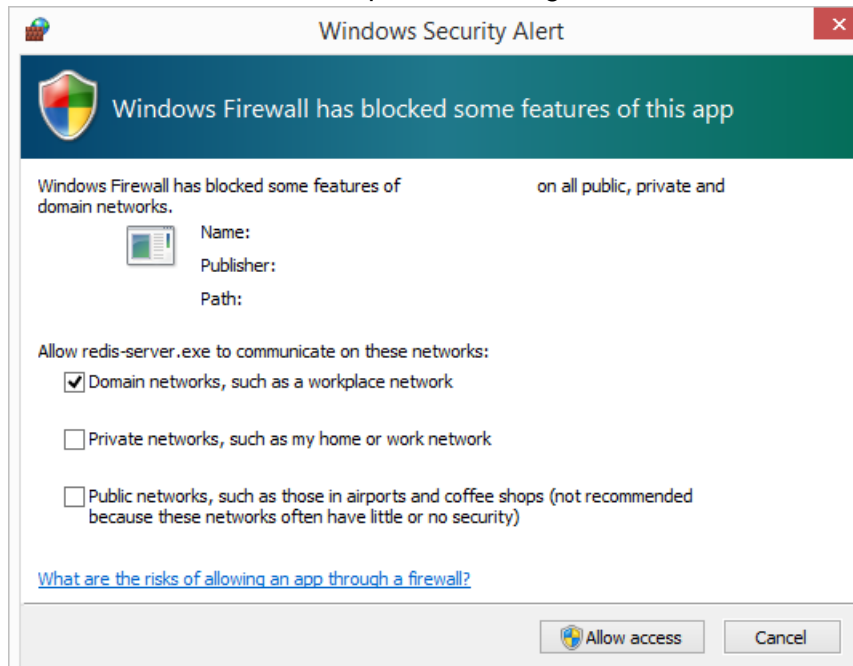
The screenshot shows a terminal window titled "Termite 3.4 (by CompuPhase)". The window has a menu bar with "Settings", "Clear", "About", and "Close". The status bar at the top indicates "COM64 115200 bps, 8N1, no handshake". The terminal displays the following configuration and startup logs:

```
MASK: 255.255.255.0
GATEWAY: 192.168.1.5
DNS1: 114.114.114.114
DNS2: 8.8.8.8
NTP SERVER: 1.UBUNTU.POOL.NTP.ORG
EUI PADDING: {3, FF}, {4, FF}
GATEWAY ID: 080027FFFF0DDDDD
LORAWAN: Public
LORAWAN SERVER: 192.168.1.5
UPLINK UDP PORT: 1680
DOWNLINK UDP PORT: 1680
CHANNEL0: 867100000, A, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL1: 867300000, A, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL2: 867500000, A, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL3: 867700000, A, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL4: 867900000, A, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL5: 868100000, B, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL6: 868300000, B, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL7: 868500000, B, SF7/SF12, BW125KHz (LORA_MULTI_SF)
CHANNEL8: 868300000, B, SF7, BW250KHz (LORA_STANDARD)
CHANNEL9: 868800000, B, 50Kbps (FSK)

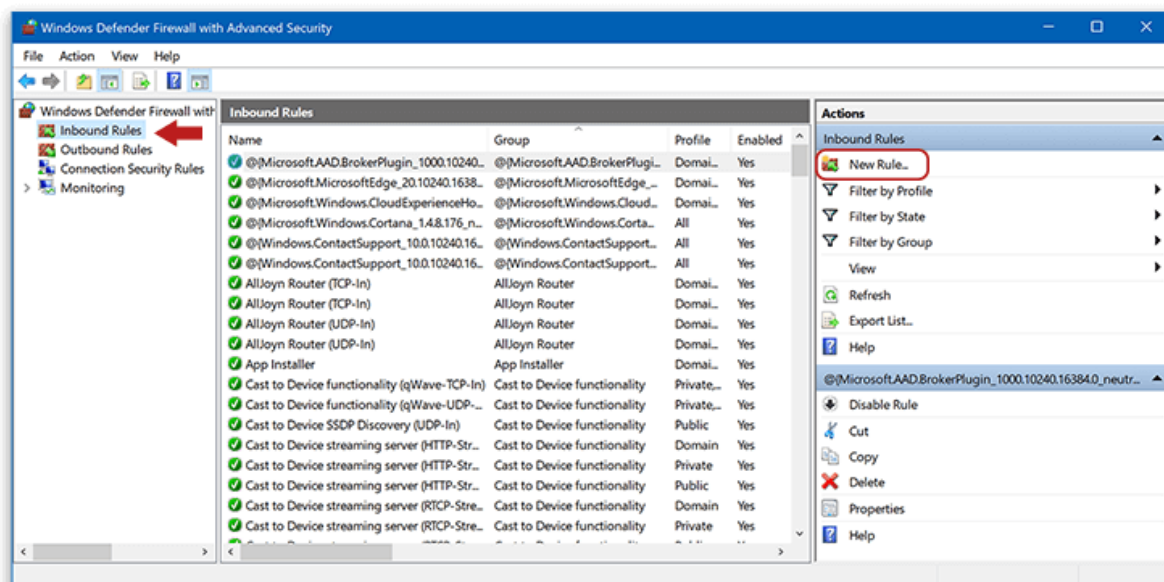
-----
Concentrator starting...
Concentrator Radio A type SX1257
Concentrator Radio B type SX1257
Concentrator started (2926ms)
ST LoRa GW V2
Ethernet starting...
STATIC IP: 192.168.1.7
Ethernet started
Downlink UDP Connected
Uplink UDP Connected
```


6. Firewall

If the connection was not successful, check the firewall settings as it is most of the time causing issues with connection to the server. It depends on the configuration of the PC, on standard personal Windows installation it is as simple as allowing the connection when it pops up.



On more restricted PCs, it is necessary to make an exception for the `erl.exe` located in `C:\Program Files\erl-23.0\bin\erl.exe` (add inbound and outbound rule).



It is even possible, that the firewall is managed by other application (usually an anti-virus package with embedded firewall). In such case, please refer to your specific firewall documentation on how to make an exception. If you need to define ports, it is necessary to **open ports 1680, both TCP and UDP and 8080** in case you access the PC running server remotely (localhost is usually unrestricted).

Help protect your PC with Windows Defender Firewall

Windows Defender Firewall can help prevent hackers or malicious software from gaining access to your PC through the Internet or a network.

! These settings are being managed by vendor application Symantec Endpoint Protection

A successful connection is visible in the server dashboard (green tick). Yellow exclamation mark usually still means connection, only red cross means that the GW is not connected.

Gateways				
MAC	IP Address	Dwell [%]	Last Alive	Status
080027FFFF0DDDDD	192.168.1.7		2020-06-02 16:28:30	✓

7. Adding a network (Infrastructure -> Networks)

To add a network, Region depends on the country, in this case EU 863-870MHz, NetID can be 000000 for private network, Max Data Rate should be SF7 for EU. The rest can be left at default/suggested values. Just to note, not every parameter is required, but when one is missing, sometimes the popup window to inform the user does not appear. Also beware of configuration windows with **multiple tabs** such as in the image below – each tab must be filled correctly before submitting the form, otherwise it will not submit the information. This does not raise any popup messages either.

Gateway Power (dBm) *

e.g. 16

!

Please fill out this field.

✓

Submit

Create new network

List

General

ADR

Channels

Name *

TestNetwork

✓

NetID *

000000

✓

Region *

EU 863-870MHz

✓

Coding Rate *

4/5

RX1 Join Delay (s) *

5

RX2 Join Delay (s) *

6

RX1 Delay (s) *

1

RX2 Delay (s) *

2

Gateway Power (dBm) *

6

✓

✓

Submit

Create new network

List

General

ADR

Channels

Max EIRP
(dBm) *

14



Max
Power *

Max - 6 dB



Min
Power *

Max - 2 dB



Max Data
Rate *

SF7 125 kHz (5470 bit/s)



Initial Duty
Cycle

1 (100%)



Initial RX1
DR Offset *

0

Initial RX2
DR *

SF12 125 kHz (250 bit/s)



Initial RX2
Freq
(MHz) *

868



Create new network

General

ADR

Channels

Initial
Channels *

0-2



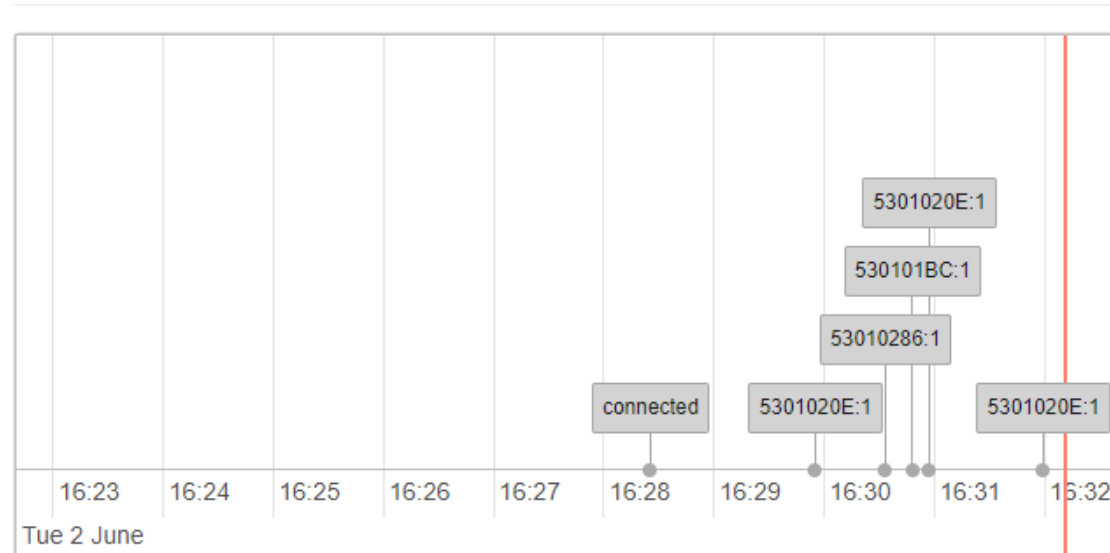
Channels

Add new channels

Submit

If one looks back at the running server, there should be a gateway connection in the dashboard, with some **notes** from the gateway (or at least a **connected** message).

Dashboard



Servers				
Name	Version	Memory	Disk	Status
lorawan@PRGCWL0678	0.6.7	3 GB (38%)		✓

Events				
Last Occurred	Entity	Eid	Text	Args
2020-06-02 16:28:25	gateway	080027FFFF0DDDDD	connected	{{192,168,1,7},50000}
2020-06-02 14:49:37	server	lorawan@PRGCWL0678	started	
2020-06-01 17:44:41	server	lorawan@PRGCWL0678	started	

Frames					
Dir	Time	Application	DevAddr	MAC	U/L SNR

Gateways				
MAC	IP Address	Dwell [%]	Last Alive	Status
080027FFFF0DDDDD	192.168.1.7		2020-06-02 16:28:30	✓

8. Creating a new group (Devices -> Groups)

Last step on the network server side is to define a group for devices which will be added. To do that, again a few steps is needed:

Create new group

[List](#)

Name *	<input type="text" value="TestGroup"/>	✓
Network *	<input type="text" value="TestNetwork"/>	✓
SubID	<input type="text" value="e.g. 0:3"/>	
Administrators	<input type="text" value="admin"/>	✓
Slack Channel	<input type="text"/>	
Can Join?	<input type="text" value="true"/>	✕ ▼
<input type="button" value="✓ Submit"/>		

9. Creating a new profile (Devices -> Profiles)

Join and FCnt check is modified to avoid possible errors when resetting devices too often. Application – by default it is a **semtech-mote** layer embedded in this LoRaWAN server, later in this tutorial it will be changed it to model Application server. ADR can be left as is.

Create new profile

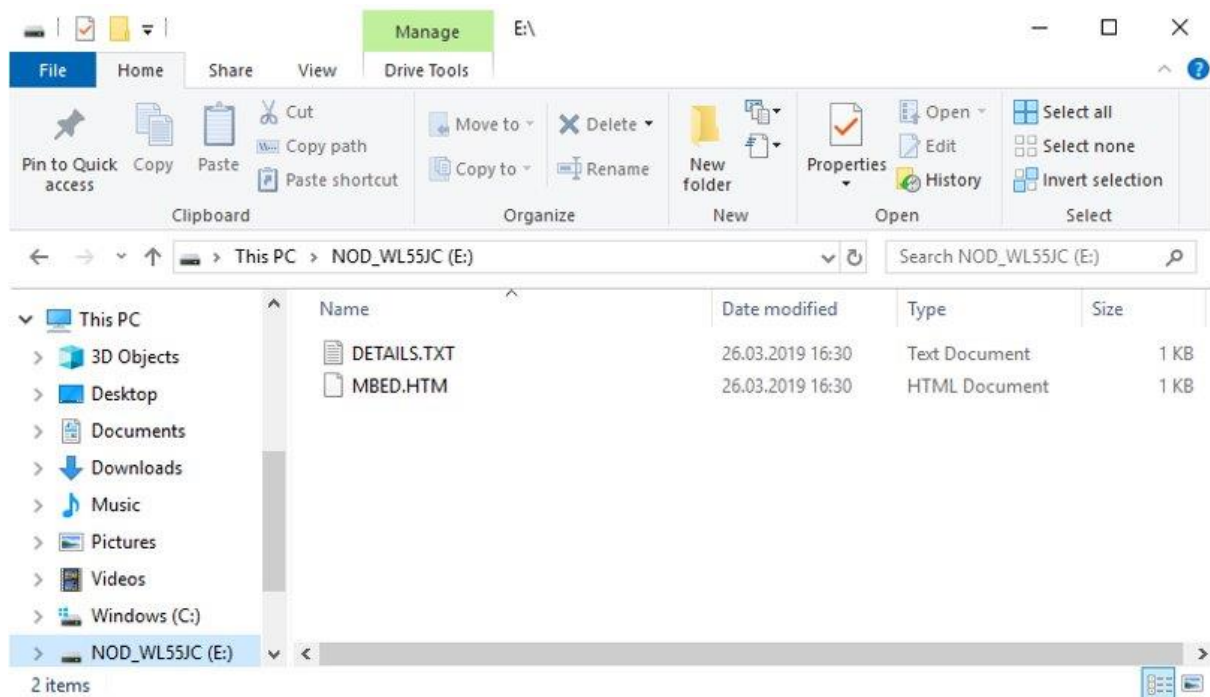
[List](#)

General	ADR	
Name *	<input type="text" value="TestProfile"/>	✓
Group *	<input type="text" value="TestGroup"/>	✓
Application *	<input type="text" value="semtech-mote"/>	✓
App Identifier	<input type="text" value="0"/>	✓
Join	<input type="text" value="Allowed with old Nonce"/>	✕ ✓
FCnt Check	<input type="text" value="Reset on zero"/>	✕ ✓
TX Window	<input type="text" value="Auto"/>	✕ ▼
<input type="button" value="✓ Submit"/>		

10. End node application – STM32WL

For the next steps, it would be useful to have a LoRaWAN application running also on the end node device. The provided binary can act as a demo application for the testing purposes of the server application.

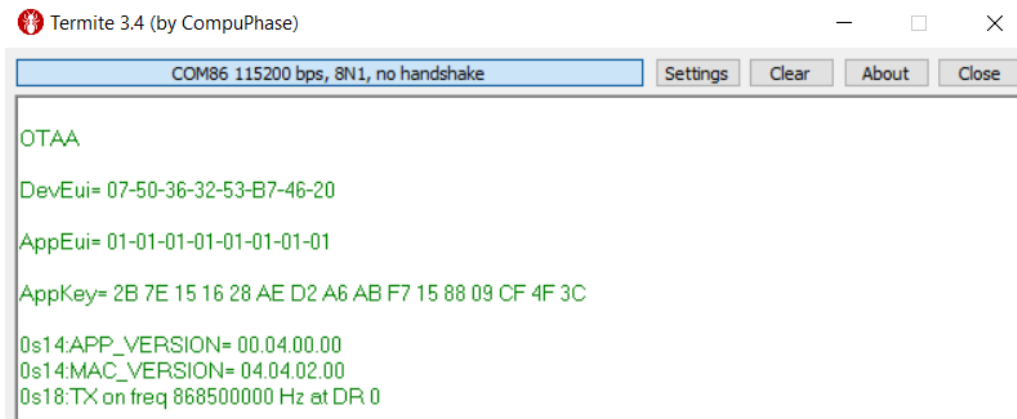
To flash the device, it is as easy as copying a file (LoRaWAN_End_Node.bin) to a USB stick. The STM32WL Nucleo will appear as mass storage device and can be opened in Windows explorer.



11. Adding the actual devices

Now it is possible to add the actual devices, to register an OTAA device, use **Commissioned**, for ABP use Activated and fill out the form. In this tutorial the registration is handled by the Application server, so this form is skipped. But if you register a device manually, you will see in the dashboard some unhandled frames coming from the node or 'not a **Semtech mote**' event, because the default Application plugin does not recognize data coming from the node.

(Optional) If you want to add a device manually, you need to go to **Devices -> Commissioned** and fill out the form. To find the data, the easiest is to open a terminal with the device and in the case of an End_Node application all the necessary information will be printed on the screen.



Edit device #0750363253B74620

 List  Delete

DevEUI *	<input type="text" value="0750363253B74620"/>								
Profile *	<input type="text" value="TestProfile"/>								
App Arguments	<input type="text"/>								
AppEUI	<input type="text" value="0101010101010101"/>								
AppKey *	<input type="text" value="2B7E151628AED2A6ABF7158809CF4F3C"/>								
Description	<input type="text"/>								
Last Joins	<table><thead><tr><th>Time</th><th>Dev Nonce</th></tr></thead><tbody><tr><td>2020-09-09 17:42:50</td><td>6D95</td></tr><tr><td>2020-09-09 17:39:34</td><td>B6CD</td></tr><tr><td>2020-06-02 17:28:53</td><td>5870</td></tr></tbody></table>	Time	Dev Nonce	2020-09-09 17:42:50	6D95	2020-09-09 17:39:34	B6CD	2020-06-02 17:28:53	5870
Time	Dev Nonce								
2020-09-09 17:42:50	6D95								
2020-09-09 17:39:34	B6CD								
2020-06-02 17:28:53	5870								
Node	<input type="text" value="00E092B4"/>								

This is a joined device, but the information necessary to be filled is visible as well. Only the Node field is not necessary to fill out, as it is generated automatically by the network.

Application server installation and configuration

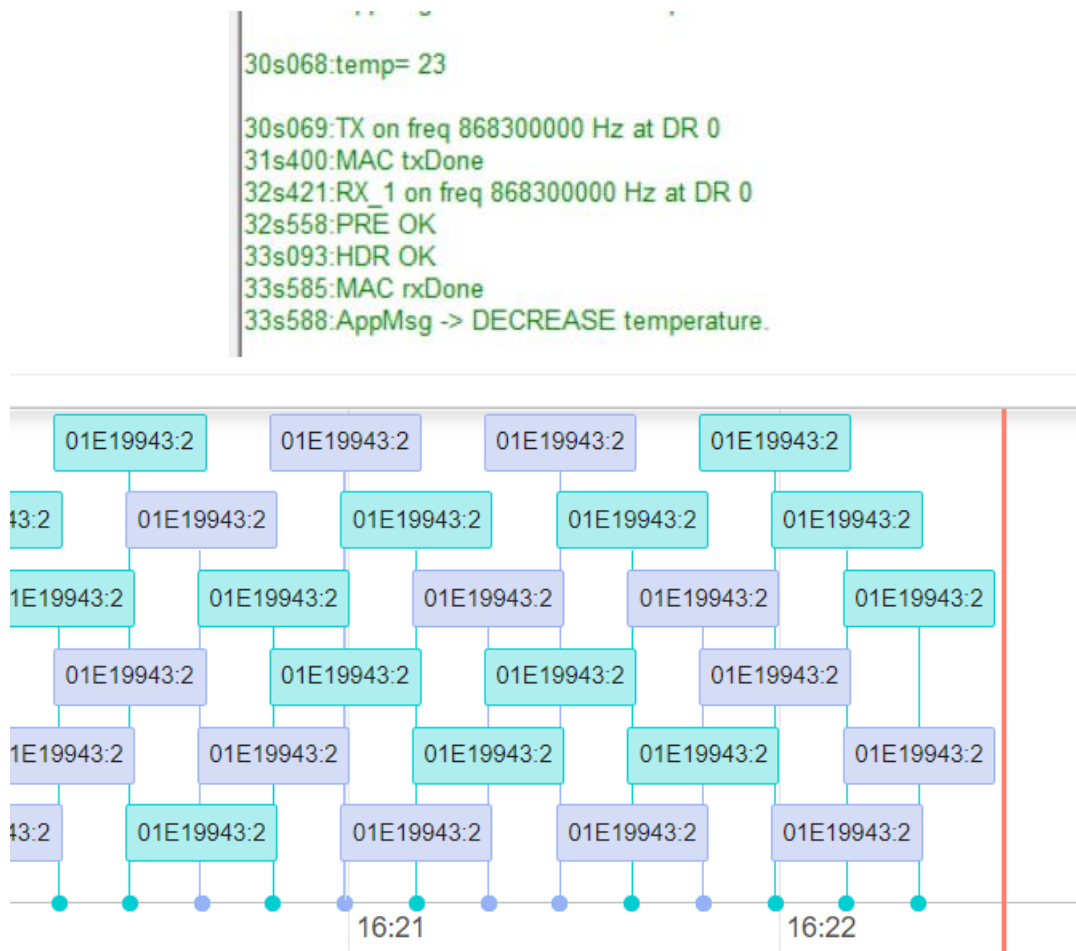
Now that the network server is up and running, steps to create an Application server follow. As mentioned in the introduction, there are two ways of implementing the temperature controller – or any other custom application.

1. (Optional) Modifying Semtech mote application

The installation of the LoRaWAN server is modular and open-source, so there is the option to take the sources from [Github](#) and modify them. The default application in the Devices->Profiles section is the Semtech mote. If one would take our temperature controller end node device and **commission** it, the server would send the frames to the Semtech mote plugin, which would reply with “not a semtech mote” – because the format is incorrect. So, the easiest way is to modify this plugin, because it replies to frames and is already configured within the server.

Compilation from sources is described on the Github – section [manual installation](#). Since the Erlang language is not very straightforward when transitioning from e.g. C language, with this tutorial there is provided the **replacement plugin file**.

In the end, with the provided file, the temperature controller application is as easy as overwriting the default plugin in `lorawan-server\lib\lorawan_server-0.6.7\ebin\lorawan_application_semtech_mote.beam`. And then keeping the Devices->Profiles to semtech-mote. Then with a connected and **commissioned** node, the result should look like this:



For a more interested reader, here is also the modification in the [source code](#):

```
init(_App) ->
    ok.

handle_join({_Network, _Profile, _Device}, {_MAC, _RxQ}, _DevAddr) ->
    % accept any device
    ok.

handle_uplink({_Network, _Profile, _Node}, _RxQ, {missed, _Receipt}, _Frame) ->
    retransmit;

handle_uplink(_Context, _RxQ, _LastMissed, _Frame) ->
    % accept and wait for deduplication
    {ok, []}.

handle_rxq({_Network, _Profile, #node{devaddr=DevAddr}}, _Gateways, _WillReply,
    #frame{port=2, data= <<>>}, []) ->
    lager:debug("PUSH_DATA ~s ignore empty frame", [lorawan_utils:binary_to_hex(DevAddr)]),
    ok;

handle_rxq(_Context, _Gateways, _WillReply, #frame{port=Port, data=Data}, []) ->
    Received = list_to_integer(binary_to_list(Data)),
    ToSend = if Received > 22 -> "D";
               Received < 22 -> "I";
               Received ==22 -> "E"
    end,
    {send, #txdata{port=Port, data= list_to_binary(ToSend)}}.

handle_delivery({_Network, _Profile, _Node}, _Result, _Receipt) ->
    ok.
```

2. Node-Red installation

This is the main path of this tutorial and with the software versions mentioned in the beginning, it is proven to work on most machines, but one cannot predict every configuration, therefore there is the optional path taken into account.

To run the temperature controller application, **Node-Red** server was selected to run the application server on localhost. It receives frames from the End node and based on the value sends the answer back to the **Network server**.

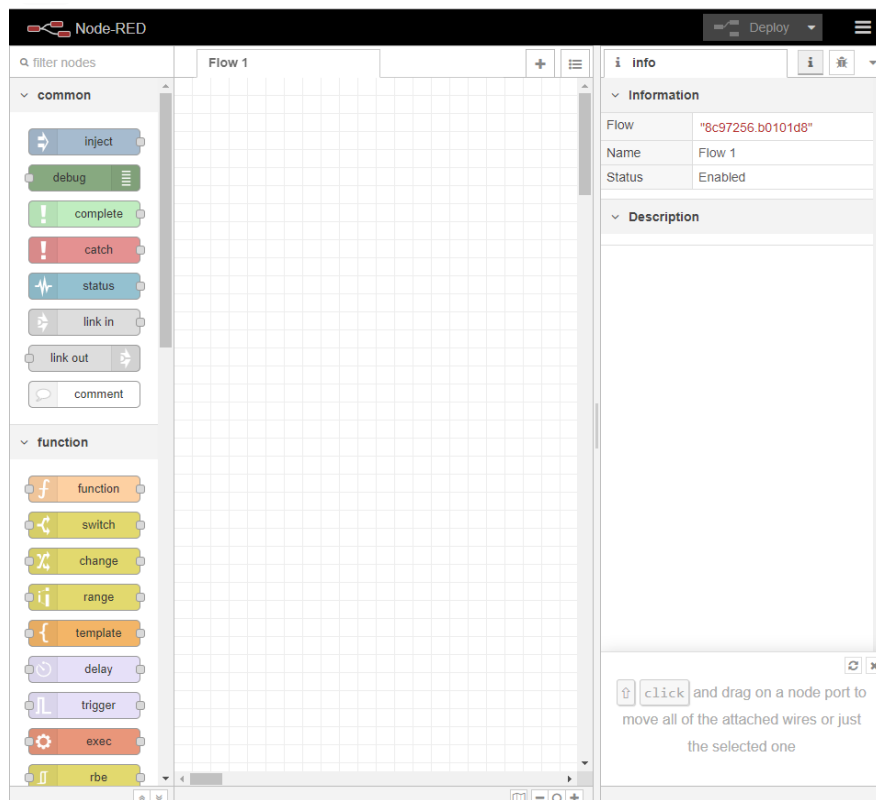
- The installation of Node-Red is again quite an easy task. Follow link: <https://nodered.org/docs/getting-started/local>
- Download **Node.js 12.17** for your platform (**msi** installer is the easiest option on Windows):
<https://nodejs.org/en/download/>
- Now open Windows command line (cmd.exe) and install NodeRed **v1.0.6**. Just type:

```
npm install -g --unsafe-perm node-red@1.0.6
```

- In case there are problems related to proxy settings, please follow this [tutorial](#)
- After installation run Node-Red server by typing in command line:

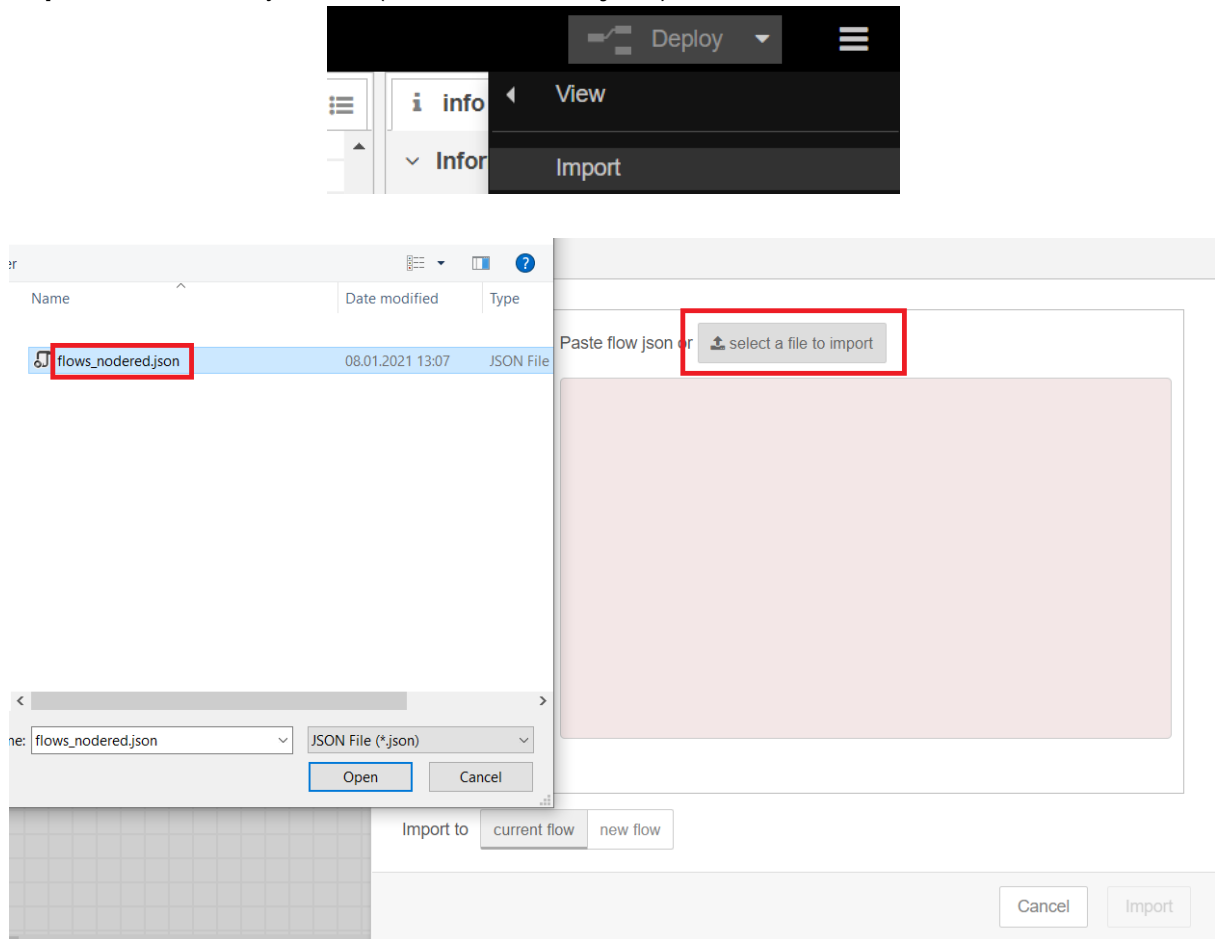
node-red

- The server should be available at: <http://localhost:1880/>

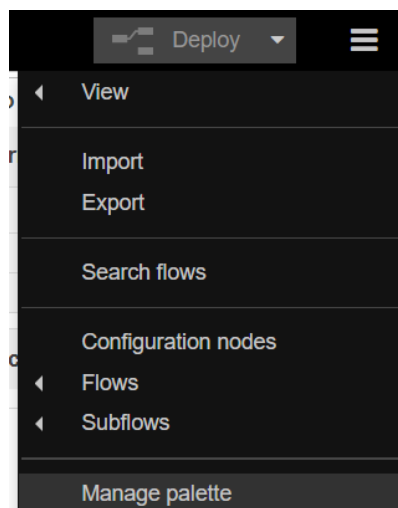


3. Adding a Flow

The **Node Red** server works with flows, so that it is easy to use, because it is almost like an application flow chart. The included flow is ready to handle our application, just click top right, **Import** and link the json file (**flows_nodered.json**).



You will get a pop up with a non-recognized package, which is the MQTT broker used in the application. Install it from the **Manage palette** in the top right corner menu, **Install** tab, find **node-red-contrib-aedes** and click **install button**.



Nodes

Install

sort:

a-z

recent

Q node-red-contrib-aedes

1 / 3039

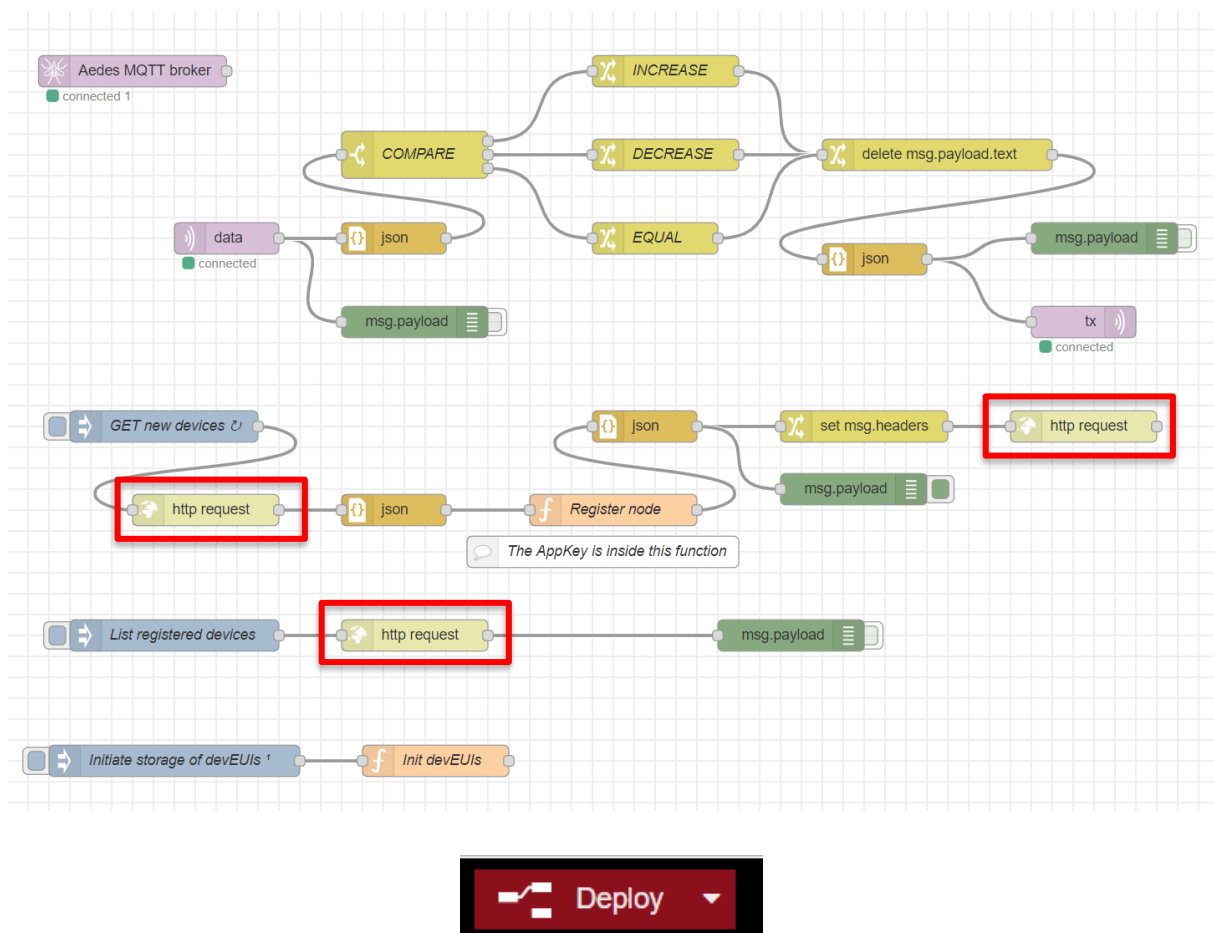
node-red-contrib-aedes

Node Red MQTT broker node based on aedes.js

0.4.0 2 months ago

installed

After install, it is needed to fill out the authentication for each **http request** (3 nodes) with **username admin, psw admin**, otherwise it will return Unauthorized status. Lastly just click the red button Deploy to run the server.



4. Connecting the App server with Nwk server

Now, the Application and Network servers are running, but they do not know about each other. In the App server there is also a MQTT handler which provides connection to the App server, so that it can read the coming data – subscribe and post new data coming back to the Nwk server. So now a MQTT connection must be defined from the Nwk server to the MQTT broker.

- Add a handler in the Nwk server (Backends -> Handlers)

Edit handler #NodeRed

[List](#)[Delete](#)

Application *

NodeRed

Uplink
Fields

devaddr ✕

Payload

ASCII Text



Parse
Uplink

Event
Fields

Filter values

Parse
Event

Build
Downlink

D/L
Expires *

Never



Test

send

to no connection

- Create a connector (Backends -> Connectors)

Create new connector

[List](#)[General](#)[Authentication](#)

Connector Name *	<input type="text" value="NodeRedConnector"/>	✓
Application	<input type="text" value="NodeRed"/>	✕ ✓
Format *	<input type="text" value="JSON"/>	✓
URI *	<input type="text" value="mqtt://localhost:1883"/>	✓
Publish QoS	<input type="text" value="Filter values"/>	▼
Publish Uplinks	<input type="text" value="data"/>	✓
Publish Events	<input type="text"/>	
Subscribe QoS	<input type="text" value="Filter values"/>	▼
Subscribe	<input type="text" value="tx/#"/>	✓
Received Topic	<input type="text" value="tx/#"/>	✓
Enabled *	<input checked="" type="checkbox"/>	✓
Failed	<input type="text" value="Filter values"/>	

- Edit the **TestProfile** (Devices -> Profiles -> TestProfile)

If everything is successful, connection status should be in green and we can move to the last step and that is changing the application in the device profile.

Edit profile #TestProfile

List Delete

General ADR

Name *

TestProfile

Group *

TestGroup

Application *

NodeRed

App Identifier

0

Join

Allowed with old Nonce

FCnt Check

Reset on zero

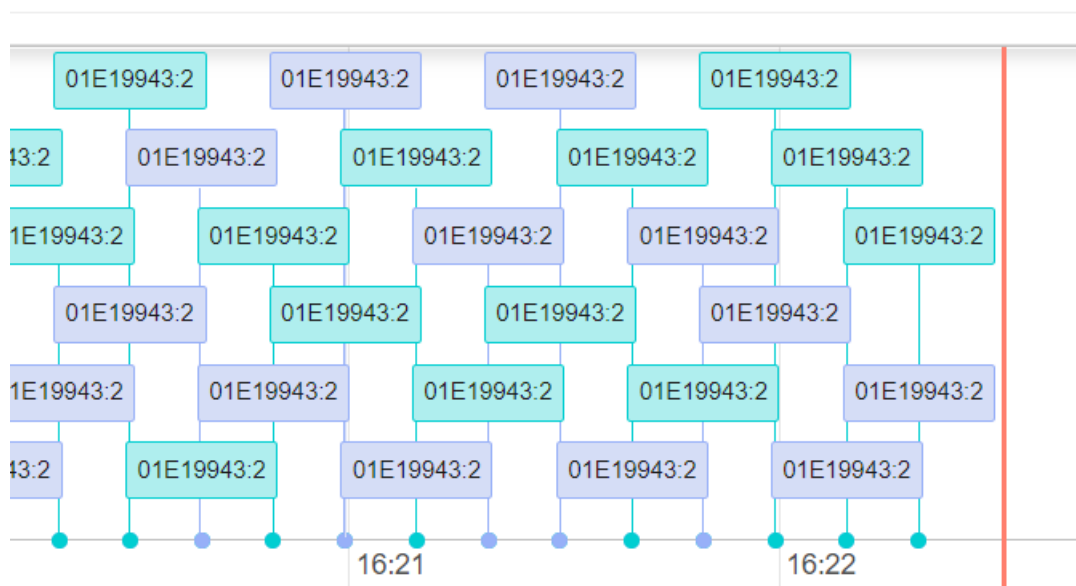
TX Window

Auto

Save changes

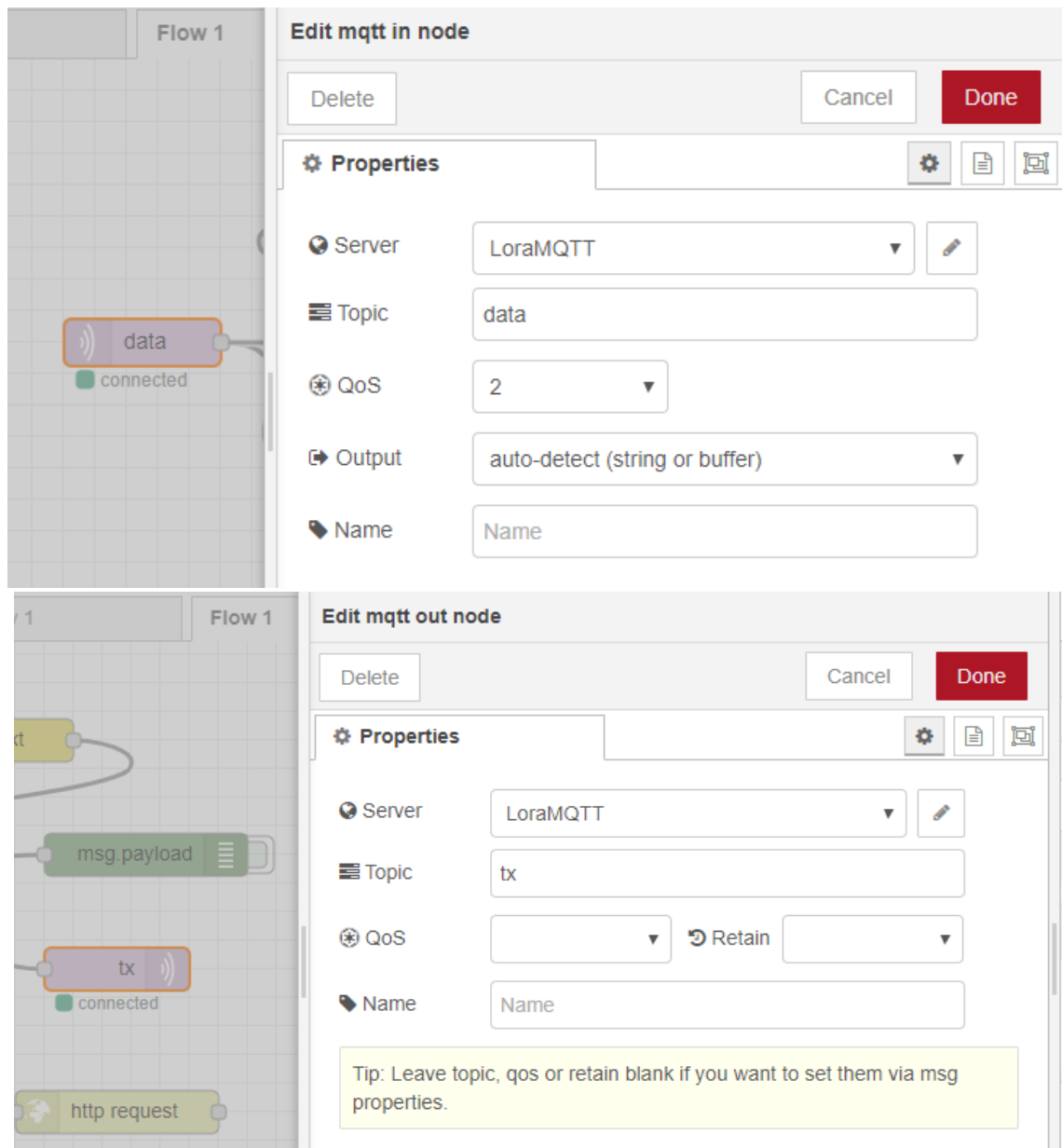
5. The application Flow

Everything is set and running and after connecting the WL board, there should be handled frames coming in.

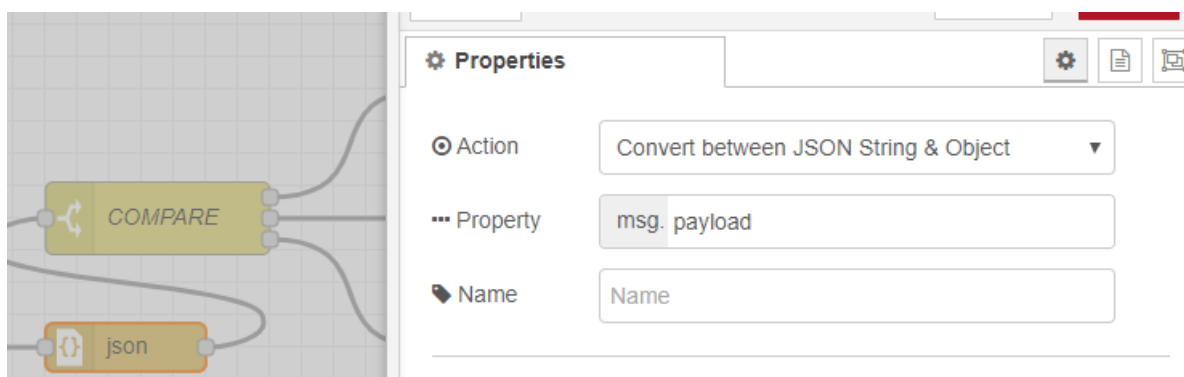


One can now modify the application as needed, e.g. changing the threshold for the temperature control. The flow used in this tutorial is quite simple, but there are hidden pieces of code to explain.

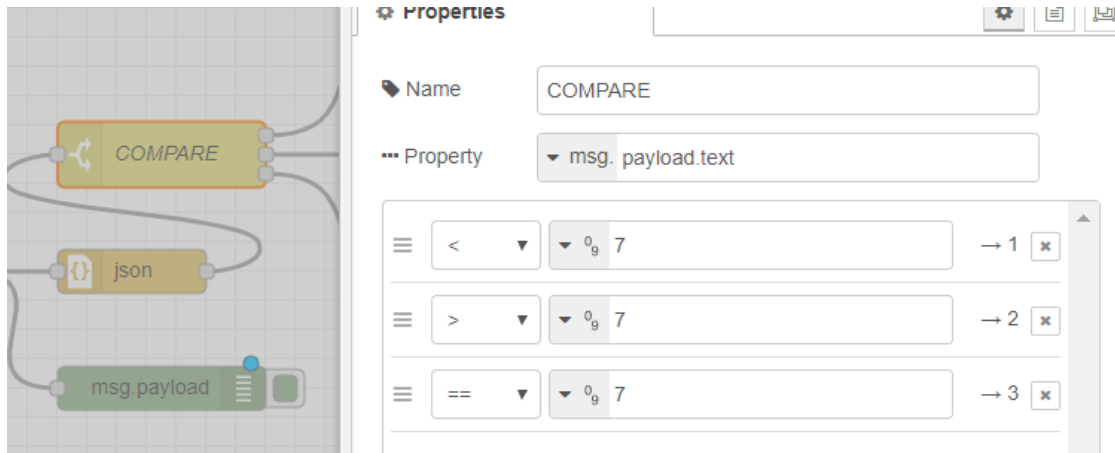
There are two mqtt brokers to receive and send data from and to the server.



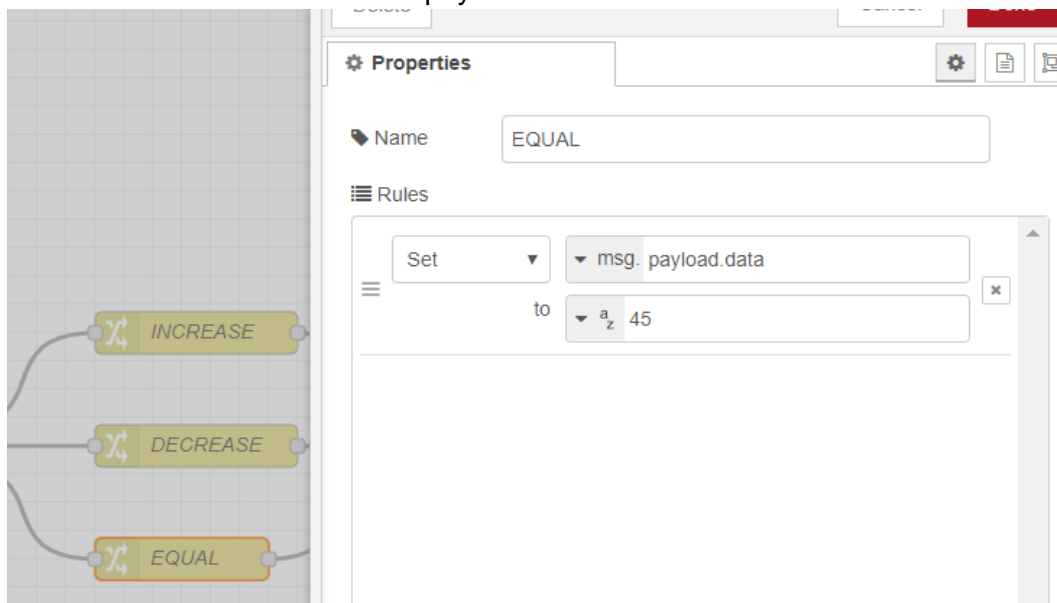
The received data is in JSON format, so a converter is used to get the actual data to an Javascript object.



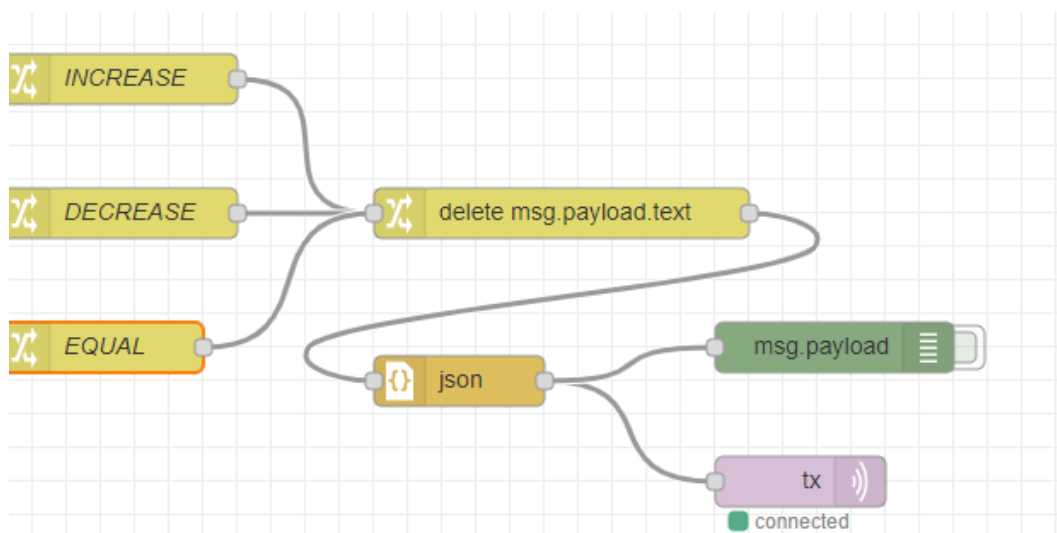
Then, in the payload.text attribute there is the data sent from our node, so it is compared as a number with a threshold.



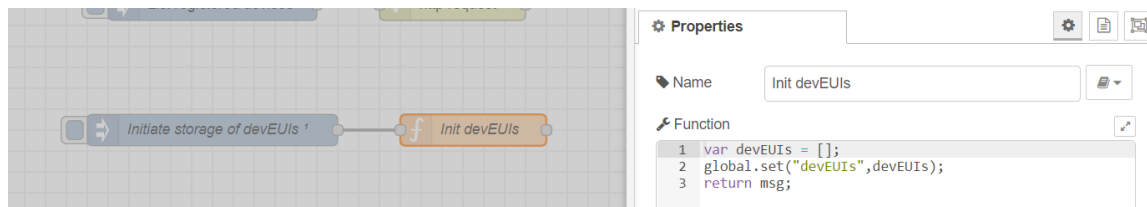
The comparison decides which path is taken (increase, decrease, equal) and each path inserts I,D,E character in hexadecimal to the payload.



The original text is deleted, the object converted again to JSON and sent over the mqtt broker.

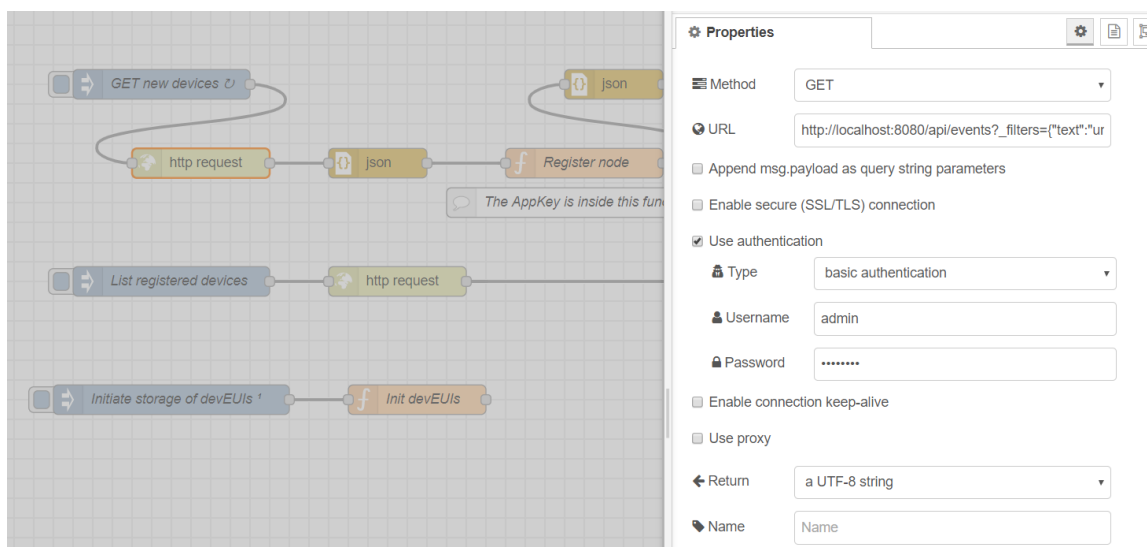


This was the main part of the application, but because the network server doesn't have any option to accept all devices, it is either necessary to register the end devices manually (which for a larger amount of devices would be quite tiresome to achieve), or with predefined information (which is in case of larger amount of devices again more complicated to maintain and not human error proof). In the end, the only easy solution is that the application server will also take care of the registration. Unfortunately, there is no API to react on the event of connection of a new device, but there is at least REST API, which can give out a list of unregistered devices. This list is checked each minute to obtain new devices. First, and to keep it simple, a storage for the devices is needed. A simple function, that will set a global variable to store the **devEUI**.

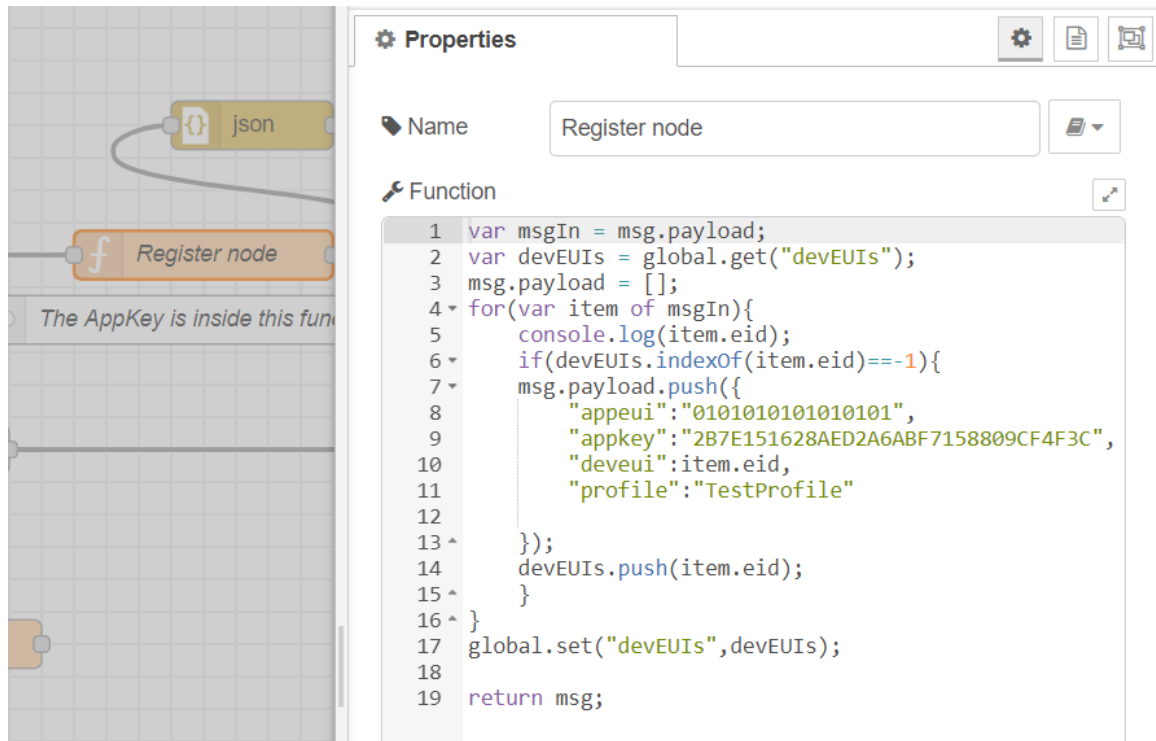


To register a node, the mentioned list of unregistered devices, which is accessible using GET on the following URL, is required:

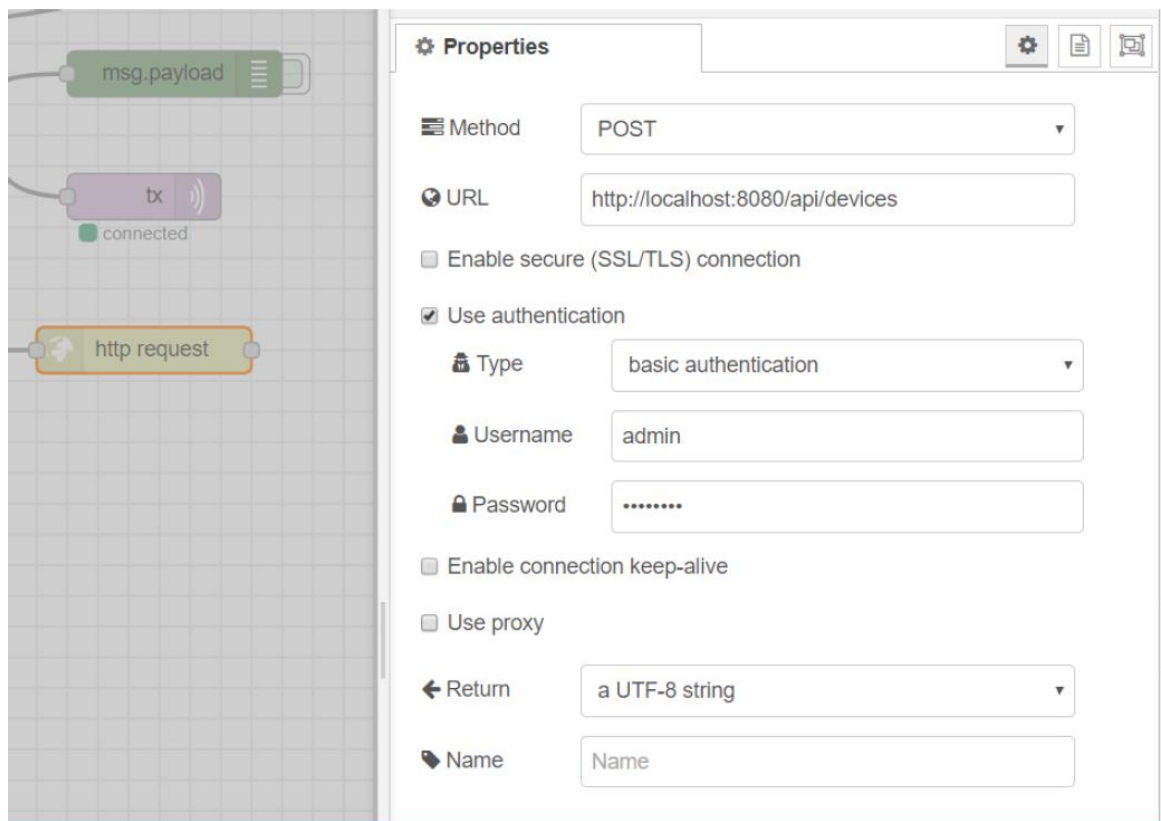
[http://localhost:8080/api/events?_filters={\"text\":\"unknown_deveui\"}](http://localhost:8080/api/events?_filters={\)



Then the payload is processed by JSON converter and then a function follows to store the information into the storage variable created before and to prepare the data for the registration request, again using REST API.



The information is converted to JSON, http headers are added and a POST is generated to the following url: <http://localhost:8080/api/devices>

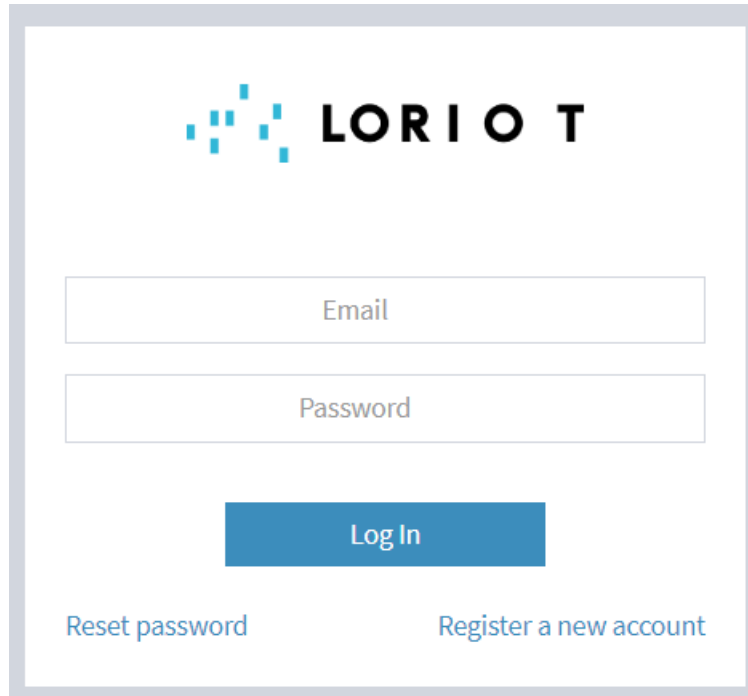


Appendix – Lorient LoRaWAN Server

The purpose of this appendix is the **basic setup** of LoRaWAN network infrastructure using the solution of commercial LoRaWAN service provider: Lorient. It is possible to create a **free account** and use a predefined **sample network** for LoRaWAN network integration following the given free network capacity and feature limitations.

1. Register the Lorient account

Go to link: <https://eu1.loriot.io>

The image shows a web interface for the Lorient LoRaWAN server. At the top, there is a logo consisting of a cluster of blue squares of varying sizes to the left of the word "LORIENT" in a bold, black, sans-serif font. Below the logo, there are two input fields: the first is labeled "Email" and the second is labeled "Password". Both labels are in a light gray font and are centered within their respective input boxes. Below these fields is a blue rectangular button with the text "Log In" in white. At the bottom of the form, there are two links: "Reset password" on the left and "Register a new account" on the right, both in a blue font.

2. Configure the Gateway

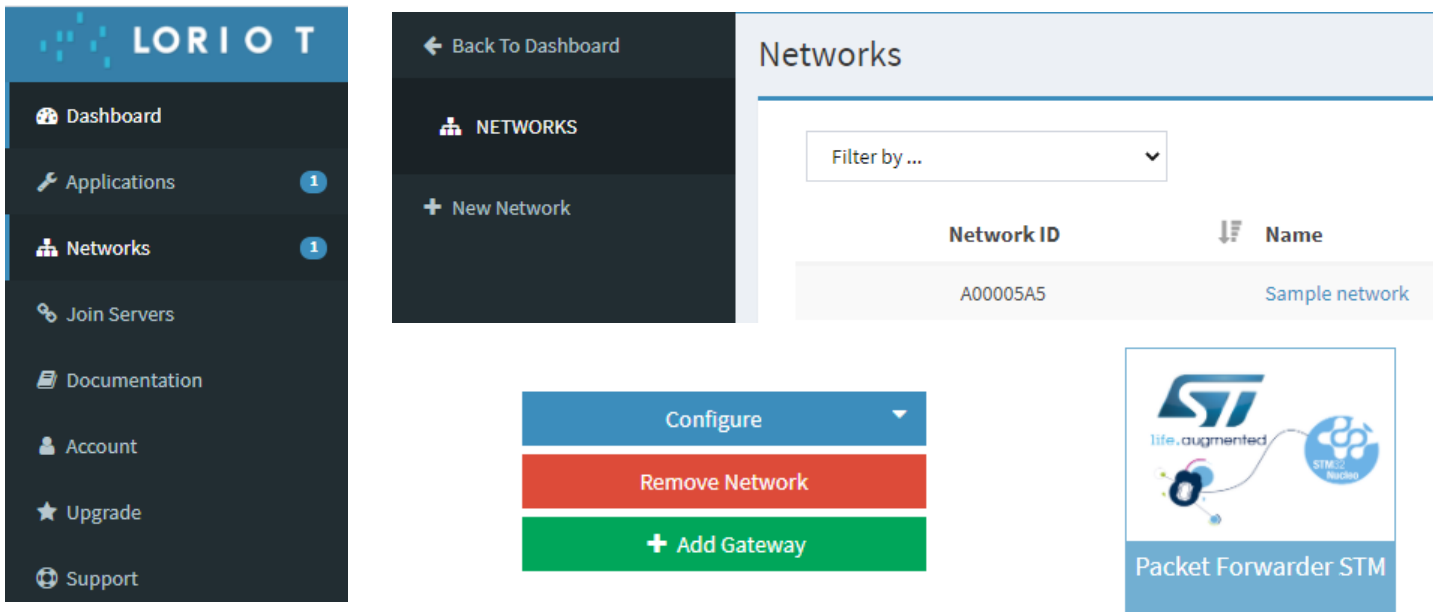
Refer to "Configuration of the STM32F7 Gateway" section of the tutorial, with the exception of connecting the Gateway to the Internet and pointing the GW to the Lorient servers. Connect to the GW by ST Link USB VCP (in a terminal app e.g. Termite use **115200, 8, N, 1**) and use command:

AT+PKTFWD= eu1.loriot.io, 1780, 1780

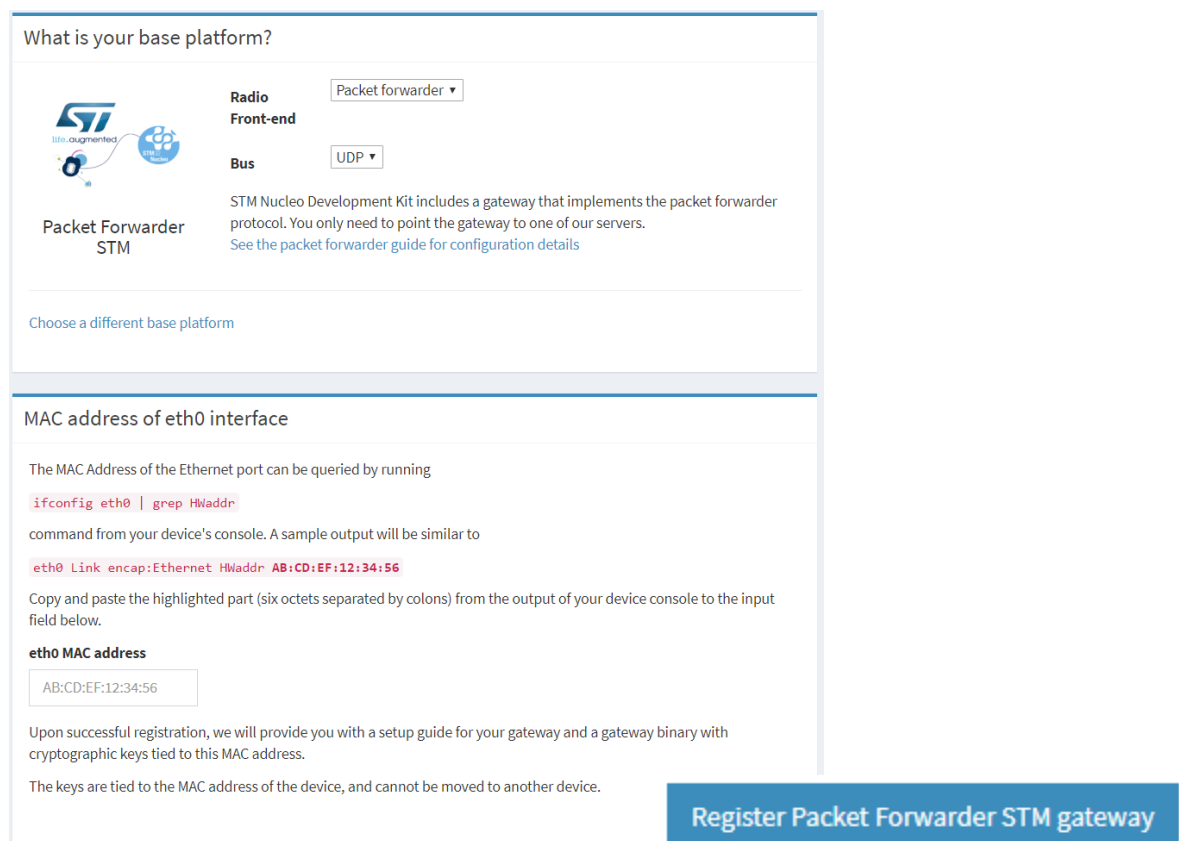
Gateway MAC can be found in the startup of the GW.

3. Adding the GW to the network

In the application, go to the Networks where should be a Sample network created. In the next step, simply open the Sample network and add a new Gateway – find Packet forwarder STM.



The configuration will ask for the Gateway MAC – fill out the acquired MAC from the GW and click Register.

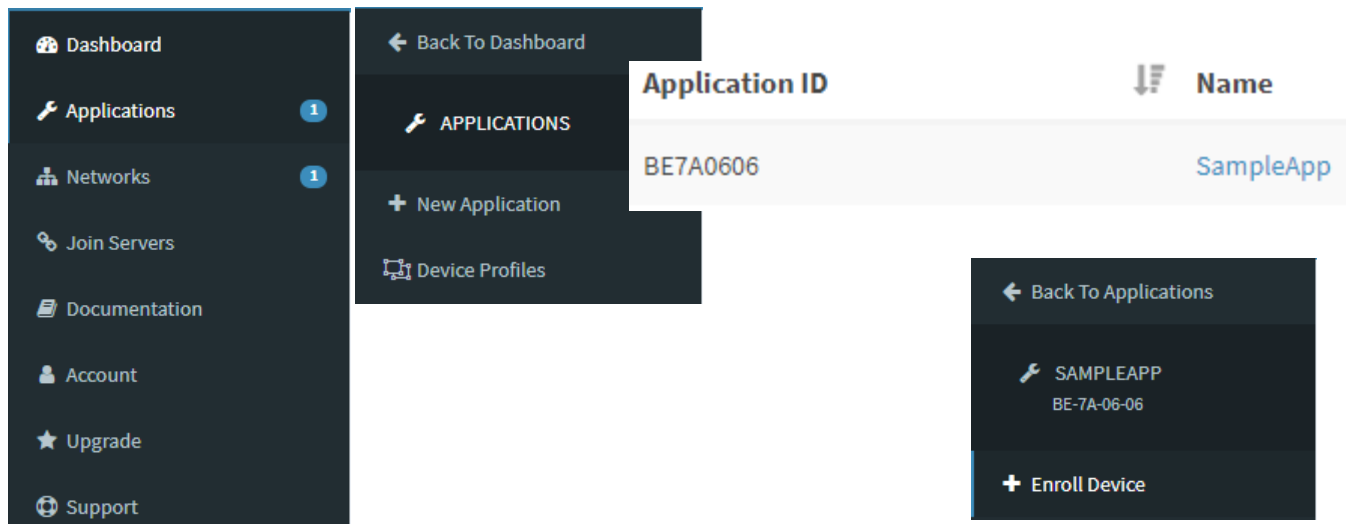


The Gateway should appear as green and online if the connection was successful.

Name	Status
08-00-27-FF-FF-0D-2B-1F	Online

4. Adding the End Node device

Return to the Lorient dashboard and Enroll a device in the SampleApp, which can be found in Applications.



To fill out the information, flash your STM32WL device with included LoRaWAN_End_Node.bin. As described in section “End node application – STM32WL”. Connect to the device using VCP of embedded ST-Link and a terminal application (115200,8,N,1), get the node credentials.

```
APP_VERSION: U1.0.0
MW_LORAWAN_VERSION: U2.2.1
MW_RADIO_VERSION: U0.6.1
##### OTAA #####
##### AppKey: 2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### NwkKey: 2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### ABP #####
##### AppSKey: 2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### NwkSKey: 2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### DevEui: 00-80-E1-15-00-00-4D-BF
##### AppEui: 01-01-01-01-01-01-01-01
```

Fill out the enrollment form:

LoRaWAN® Version: LoRaWAN® 1.0.x

Enrollment Process: OTAA

Location: Disabled

You can define coordinates for static devices enabling this option.

Details

Title: MyNode1

Description:

Device EUI: 0080E11500004DBF

Application EUI: 0101010101010101

Application Key: 2B7E151628AED2A6ABF7158809CF4F3C

Device Profile:

☐ Create Another

5. Joining the device

In order to join the network, press the Nucleo-WL reset button and wait until the end of JOIN process, follow the red LED blinking and relevant terminal log.

Red LED blinks here

```
APP_VERSION:      U1.0.0
MW_LORAWAN_VERSION: U2.2.1
MW_RADIO_VERSION:  U0.6.1
##### OTAA #####
##### AppKey:  2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### NwkKey:  2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### ABP #####
##### AppSKey: 2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### NwkSKey: 2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
##### DevEui:  00-80-E1-15-00-00-4D-BF
##### AppEui:  01-01-01-01-01-01-01-01
0s048:TX on freq 868100000 Hz at DR 0
1s552:MAC txDone
6s574:RX_1 on freq 868100000 Hz at DR 0
6s710:PRE OK
7s247:HDR OK
8s393:MAC rxDone

##### = JOINED = OTAA =====
```

6. Application data

Observe the uplink log & statistics on server side using SAMPLEAPP dashboard. It is also possible to analyze the received data using Devices option. Received uplink payload contains temperature ASCII characters i.e. 0x32 0x34 → **24°C**.

The screenshot shows the SAMPLEAPP dashboard interface. On the left, a sidebar menu contains several options: 'Back To Applications', 'SAMPLEAPP BE-7A-06-06' (highlighted with a red box), 'Enroll Device', 'Enroll Multicast Device', 'Bulk Import', 'Devices' (highlighted with a red box), 'Multicast Devices', 'Device', 'Output', 'API Data', 'Websocket', 'Statistics', 'Join Server', 'Access Tokens', and 'Log'. The main area displays 'Last 25 frames received' in a table format.

Device EUI	FCntUp	Time	Port	Data
00-80-E1-15-00-00-4D-BF	23	a few seconds ago	2	32 34
00-80-E1-15-00-00-4D-BF	22	a few seconds ago	2	32 34
00-80-E1-15-00-00-4D-BF	21	a few seconds ago	2	32 34
00-80-E1-15-00-00-4D-BF	20	a minute ago	2	32 34
00-80-E1-15-00-00-4D-BF	19	a minute ago	2	32 34