Spare gateways 2024: a84041ffff25129c, a84041ffff27c238

May 21, 2022

Device Addr: 26 02 1B 08

Network Session Key:  
F6 15 64 61 96 55 46 54 37 B8 01 F8 31 94 E5 8A

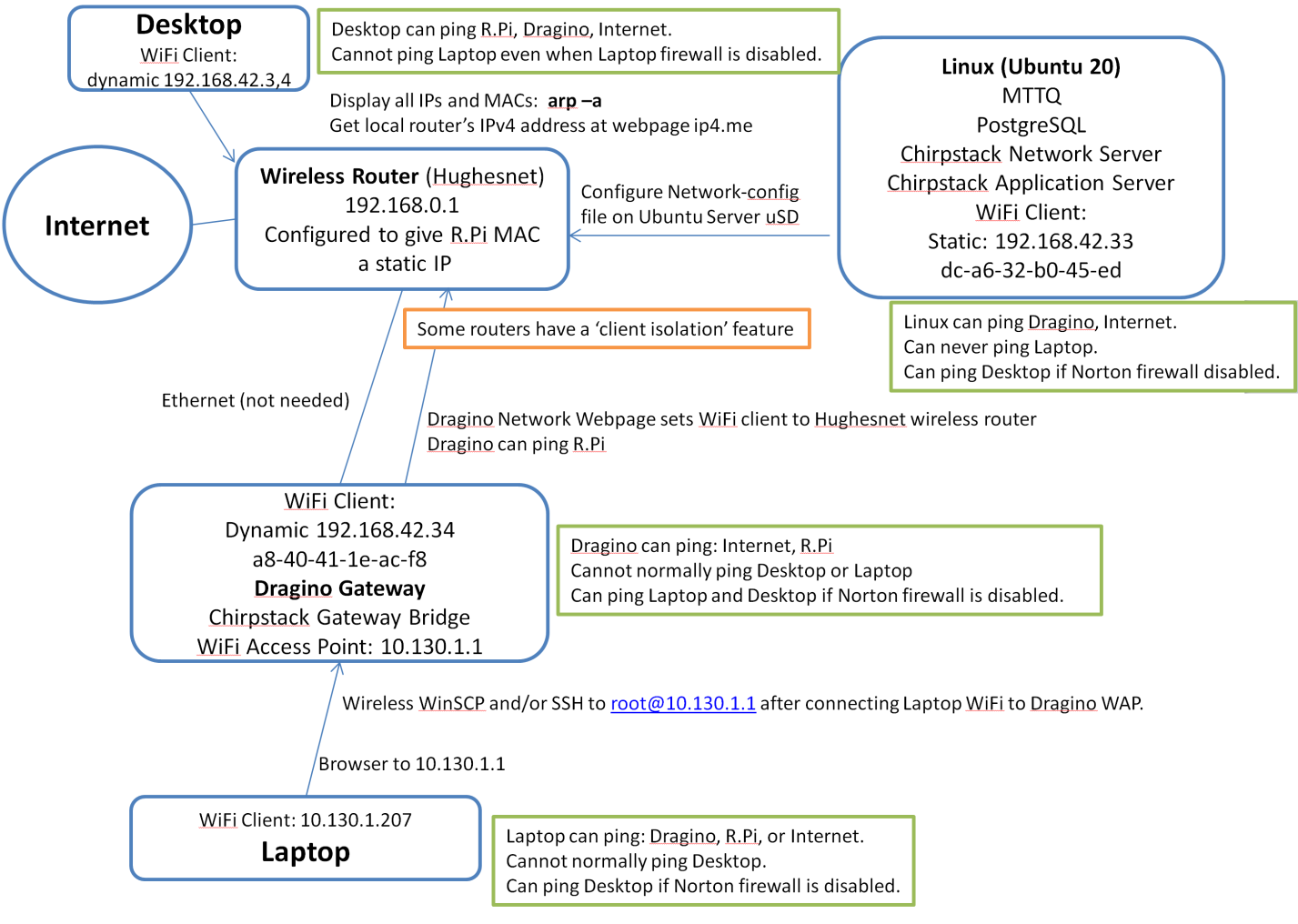
Application Session Key:  
56 59 33 B7 8D 09 D8 50 52 47 9B EE 94 AF AC 39

00 0B 33 39 4F B6 0B C8

Chirpstack webpage: <http://192.168.42.38:8080>  
or localhost:8080

Static IP IONOS July 2023: 62.151.182.141

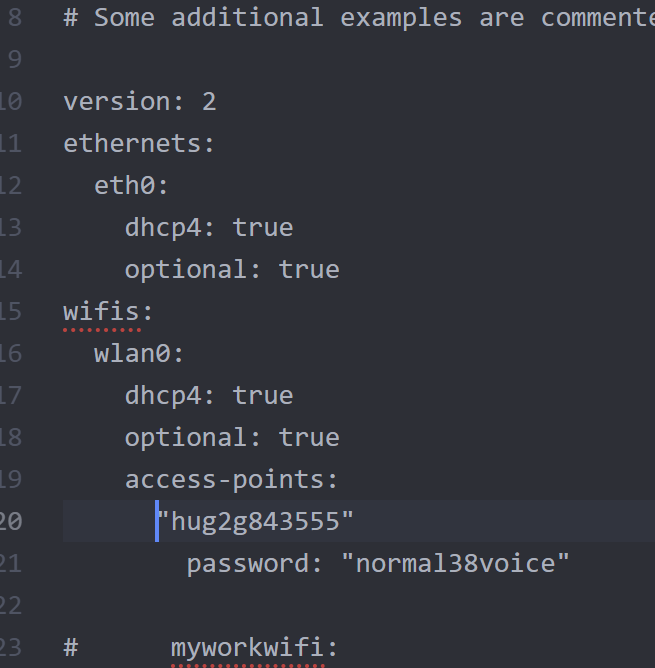
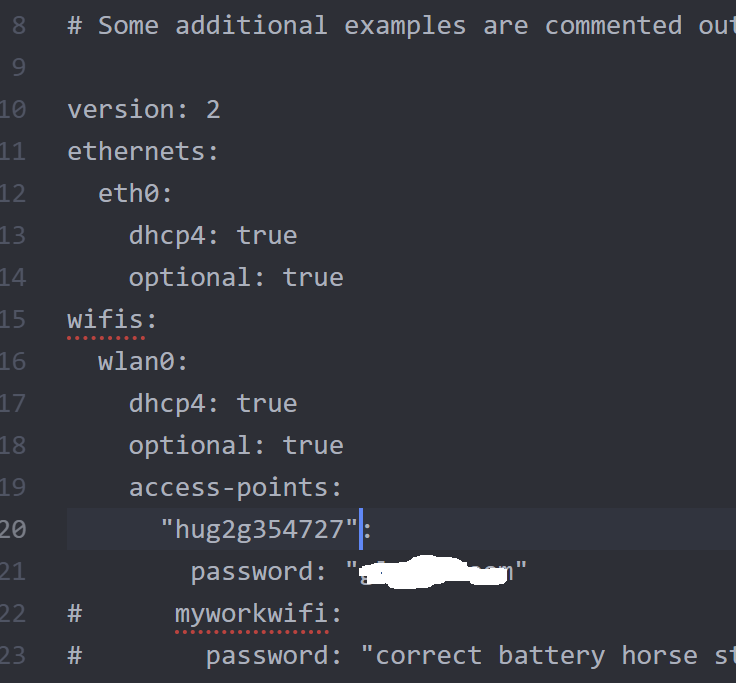
This is a map of how a local wireless network might be after installing and configuring everything:



**[to use an R.Pi instead of a Linux PC]** Create an R.Pi Ubuntu microSD card by running Raspberry Pi Imager desktop application (it’s installed on my Surface Studio). Choose Ubuntu Server 20.04 64-bit cached on desktop.   
While the card is still in the PC, use a text editor to edit the network-config file on the uSDcard so that the R.Pi’s WiFi wakes up and connects to the local wireless access point.

First boot in the RPi, wait at the login prompt a minute or two for a bunch of cloud-init and ssh fignerpint stuff to display. Then login with ubuntu ubuntu and change the password.  
**Ubuntu default login ubuntu, pw: ubuntu. Changed to qtqtqt321ag**

Try ping 8.8.8.8 to confirm online. I had to reboot again before it went online.



**[SIDE NOTE:** Setting the Wireless Access Point and pw here in network-config file only works with a virgin Ubuntu SD card. Because after it runs this, code in Ubuntu runs a cloud-init thing that creates netplan files that are used for future networking. So if you want to change an existing Ubuntu Chirpstack installation you have to do this crap:  
**Addendum to SIDE NOTE: after doing this the postgres pw stopped working. Had to create new uSD from scratch]**  


**[END SIDE NOTE. Addendum to SIDE NOTE: after doing this the postgres pw stopped working. Had to create a virgin Ubuntu uSD card from scratch.]**

Pw: qtqtqt321ag

From PC: ssh [Ubuntu@192.168.42.33](mailto:Ubuntu@192.168.42.33) get this number from ifconfig command below.sudo

sudo apt-get update

[doing this locked up ability to install anything else for 10-15 minutes. It will report another process using frontend lock]

sudo apt-get install net-tools or sudo apt intstall net-tools (allows use of ifconfig command).

ifconfig should print the 192.168.42.33 or whatever under wlan0 inet. If local WiFi router has been set up to issue the R.Pi a static IP (instructions on how to do that later in this doc), then this should report that static IP.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* MQTT and DATABASE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sudo apt install mosquitto [two ts, don’t let autocorrect change to one t]

sudo apt install postgresql

sudo apt install redis-server

sudo apt install mosquitto mosquitto-clients redis-server redis-tools postgresql

sudo -u postgres psql

-- set up the users and the passwords

-- (note that it is important to use single quotes and a semicolon at the end!)

create role chirpstack\_as with login password 'gnpasas';  
[2021: this contains typo, it was entered as:  
create role chirpstack\_as with login password 'gnpass';

create role chirpstack\_ns with login password ‘gnpnsns’;

[note single quotes, semicolon at end. Should respond with “CREATE ROLE”]

-- create the database for the servers

create database chirpstack\_as with owner chirpstack\_as;

create database chirpstack\_ns with owner chirpstack\_ns;

-- change to the ChirpStack Application Server database

\c chirpstack\_as

-- enable the pq\_trgm and hstore extensions

-- (this is needed to facilitate the search feature)

create extension pg\_trgm;

-- (this is needed to store additional k/v meta-data)

create extension hstore;

-- exit psql

\q

sudo apt install apt-transport-https dirmngr

sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 1CE2AFD36DBCCA00

sudo echo "deb https://artifacts.chirpstack.io/packages/3.x/deb stable main" | sudo tee /etc/apt/sources.list.d/chirpstack.list

sudo apt update

apt list --upgradable

sudo apt upgrade [to proceed with all of the listed updatesj. Took ten minutes.]

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GATEWAY BRIDGE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**[There are recommendations to install this on Dragino instead of R.Pi, but when I got it working it was on RPi. So proceed with this installation on RPi.]**

sudo apt install chirpstack-gateway-bridge

# start chirpstack-gateway-bridge

sudo systemctl start chirpstack-gateway-bridge

# start chirpstack-gateway-bridge on boot

sudo systemctl enable chirpstack-gateway-bridge

to confirm:

sudo systemctl status chirpstack-gateway-bridge

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* NETWORK SERVER \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sudo apt install chirpstack-network-server

# start chirpstack-network-server

sudo systemctl start chirpstack-network-server

# start chirpstack-network-server on boot

sudo systemctl enable chirpstack-network-server

to confirm: sudo systemctl status chirpstack-network-server

[at this point, running sudo systemctl status chirpstack-network-server will show that it’s running, but with password errors to the postgres db.  
ACTUALLY: second thought – don’t run status until after editing the config file below – that might avoid having to restart the R.Pi.]

to create a copy of the default or existing config file:

sudo chirpstack-network-server configfile > chirpstack-network-server.toml

to edit the config file:

sudo nano /etc/chirpstack-network-server/chirpstack-network-server.toml

[general]

log\_level=4

[postgresql]

dsn="postgres://chirpstack\_ns:gnpnsns@localhost/chirpstack\_ns?sslmode=disable"

[network\_server]

net\_id="000000"

[network\_server.band]

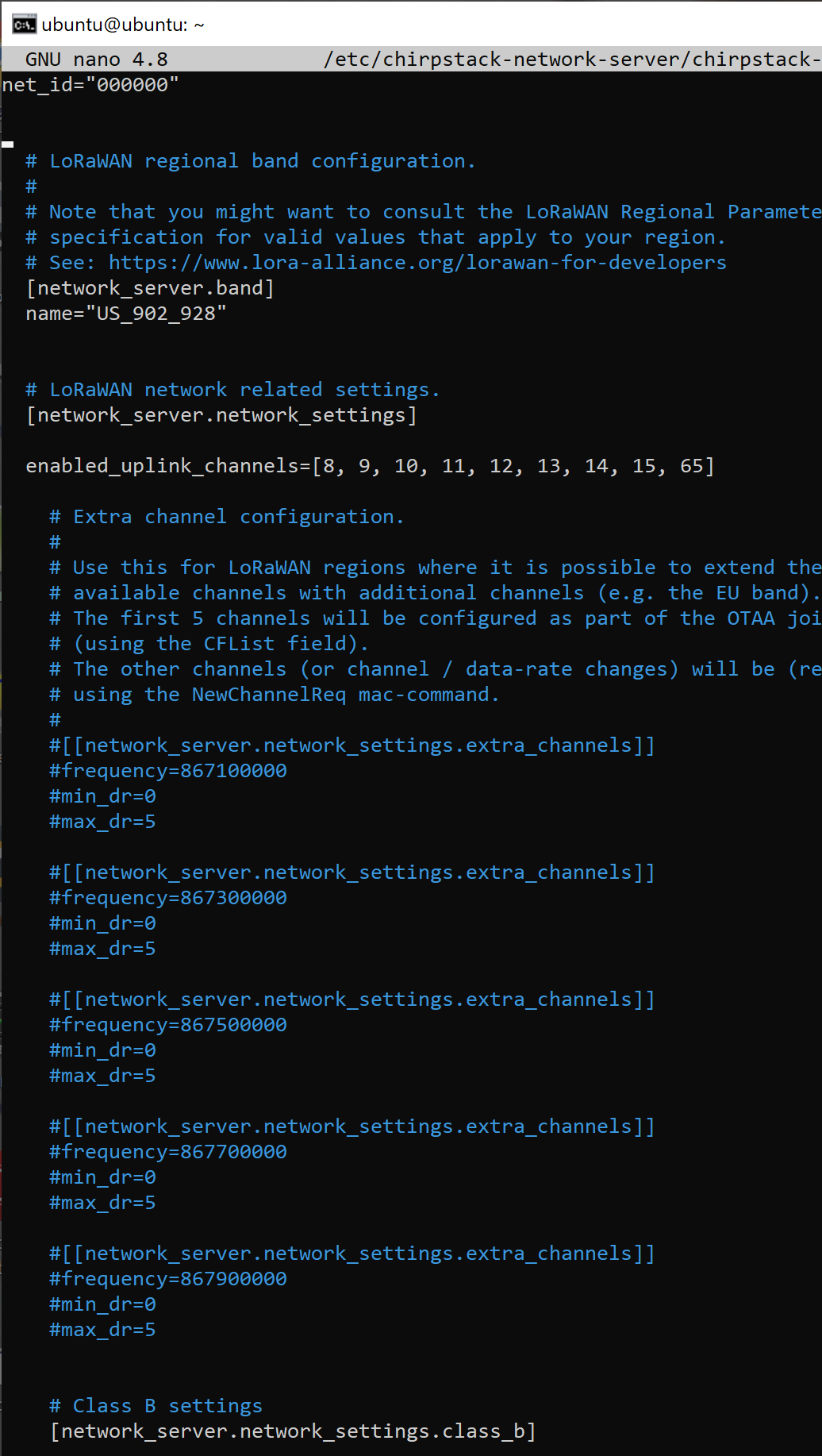
name="US\_902\_928"

[network\_server.network\_settings]

enabled\_uplink\_channels=[8, 9, 10, 11, 12, 13, 14, 15, 65]

[see screenshot below]





[at this point, running sudo systemctl status chirpstack-network-server will show that it’s running, but with password errors to the postgres db.  
Have to shutdown and restart the RPi (maybe just restart network-server but I don’t know how to do that), but after restarting the new config file with the postgres pw should work. Now I get status msgs about gateway/mqtt stats packet received. Then handle stats packet error.]

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* APPLICATION SERVER \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sudo apt install chirpstack-application-server

to edit the config file:

sudo nano /etc/chirpstack-application-server/chirpstack-application-server.toml

[general]

log\_level=4

[postgresql]

dsn="postgres://chirpstack\_as:gnpasas@localhost/chirpstack\_as?sslmode=disable"

2021: dsn="postgres://chirpstack\_as:gnpass@localhost/chirpstack\_as?sslmode=disable"

[application\_server.external\_api]

jwt\_secret="verysecret"

# start chirpstack-application-server

sudo systemctl start chirpstack-application-server

# start chirpstack-application-server on boot

sudo systemctl enable chirpstack-application-server

[at this point (assuming made the .toml config file edits before starting the application-server), running status will display msgs about “integration/mqtt…”, and “starting join-server”]

Enter the PostgreSQL as the postgres user:

sudo -u postgres psql

Within the PostgreSQL prompt, enter the following queries:

-- create the chirpstack\_as\_events user

create role chirpstack\_as\_events with login password 'dbpassword'; (should this be gnpasevents?)

-- create the chirpstack\_as\_events database

create database chirpstack\_as\_events with owner chirpstack\_as\_events;

-- enable the hstore extension

\c chirpstack\_as\_events

create extension hstore;

-- exit the prompt

\q

To verify if the user and database have been setup correctly, try to connect to it:

psql -h localhost -U chirpstack\_as\_events -W chirpstack\_as\_events

[application\_server.integration]

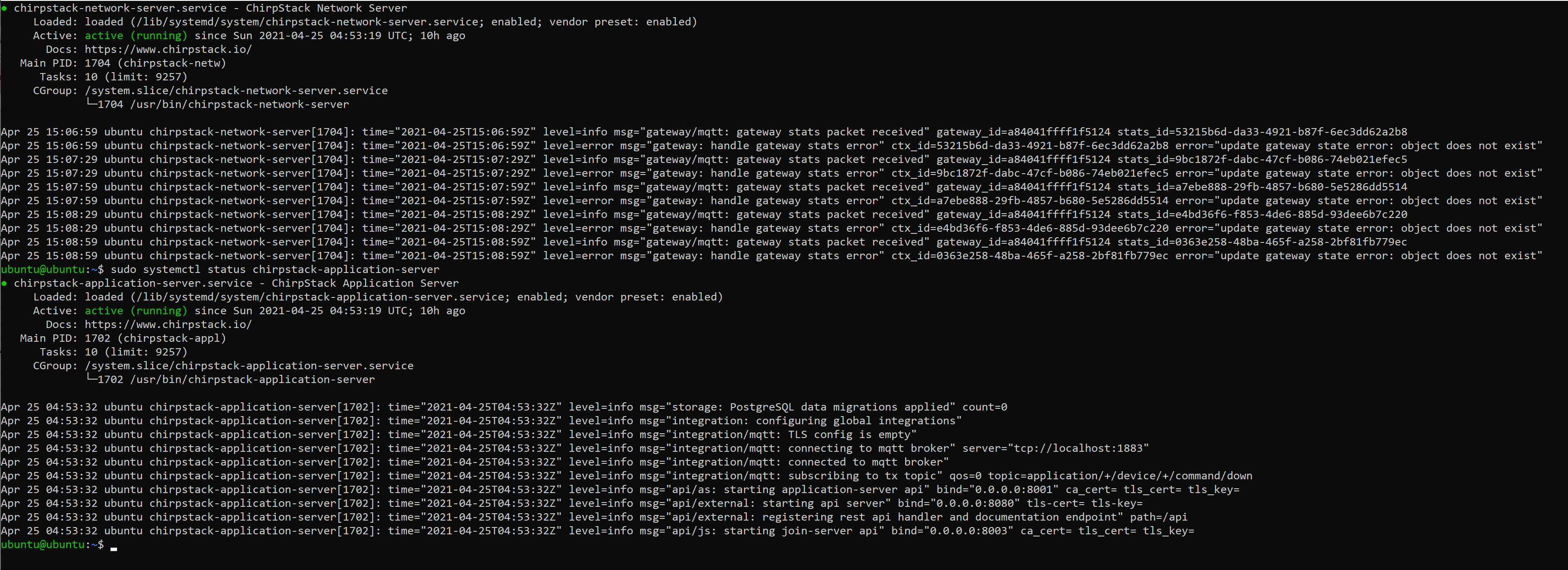
# Enabled integrations.

Find enabled=["mqtt"]  
Change to enabled=["mqtt", "postgresql"]

Add:

[application\_server.integration.postgresql]

dsn="postgres://chirpstack\_as\_events:gnpasevents@localhost/chirpstack\_as\_events?sslmode=disable "



From a host PC on the same wireless network, should be able to go to <http://192.168.42.38:8080> and get the chirpstack webpage.   
Login with **admin admin**.  
  
If no need to realty-check the gateway and device on TTN, then it’s okay to skip from here down to REGISTER GATEWAY ON CHIRPSTACK NETWORK SERVER

## Routing-profile

When creating a new Network Server, ChirpStack Application Server will create a Routing Profile on the given Network Server, containing the hostname:ip of the ChirpStack Application Server installation. In case your ChirpStack Application Server installation is not reachable on localhost, make sure this hostname:ip is configured correctly in your [Configuration](https://www.chirpstack.io/application-server/install/config/). This Routing Profile is updated on Network Server updates and deleted on Network Server deletes.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* DRAGINO GATEWAY RECOVERY \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

To recover a boinked DLOS08 gateway, follow these instructions:

http://wiki.dragino.com/xwiki/bin/view/Main/How%20to%20Recover%20Gateway%20if%20can%27t%20access%20it/

Download the latest firmware from: https://dragino.com/downloads/index.php?dir=LoRa\_Gateway/

Recovery will require an Ethernet cable and PC with this static Ethernet setting:  
 using cmd with admin:

netsh interface ip set address name=”Ethernet 2” static 192.168.255.2 255.255.255.0

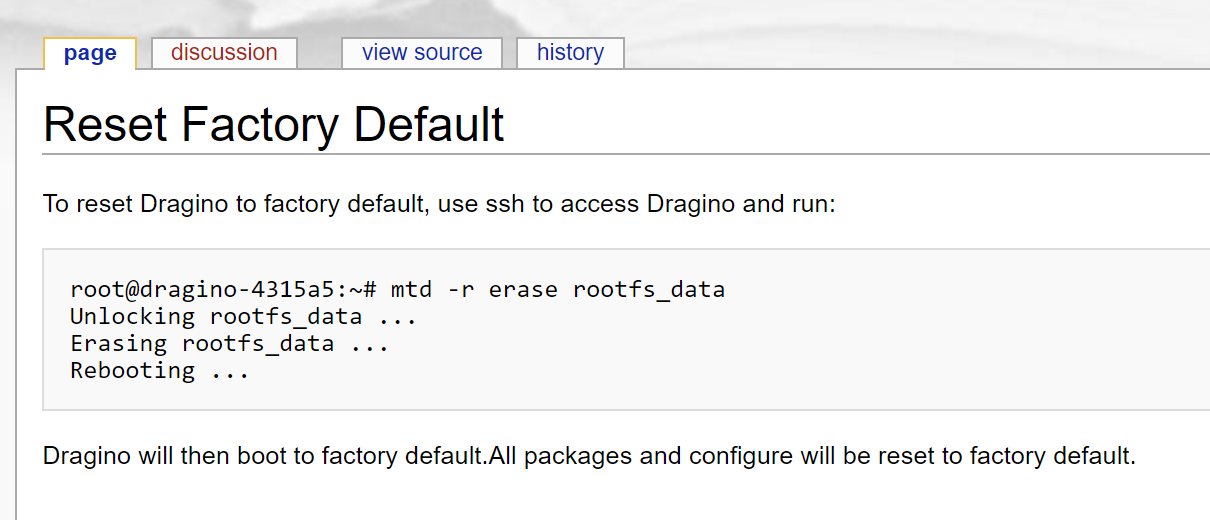
some computers: netsh interface ip set address name=”Ethernet” static 192.168.255.2 255.255.255.0

Recovery will require downloading latest firmware from Dragino, holding button down while applying power, then connect with browser to 192.168.255.1, select file and upgrade firmware.

Leave Dragino powered up until LEDs (including green) start flashing.  
Then connect with WiFi to the Dragino hot spot. Then 10.130.1.1

Menu: Logout when finished.

Default Dragino WiFi pw dragino+dragino



/\*\*\*\*\*\*\*\*\*\*\* SSH KNOWN HOSTS ISSUE \*\*\*\*\*\*\*\*\*\*\*\*/

Open c:/users/doug/.ssh/known\_hosts in a text editor and remove 10.130.1.1 line

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* BASIC GATEWAY AND TTN \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Connect PC wireless to Dragino. 10.130.1.1. (un: root pw: dragino+dragino)

Menu: Network:WiFi  
 Add check to Client Settings.  
 WiFi Survey: Drop down and select. Add password. Save and apply.  
 Menu: Home: should see four green checks.  
 Dragino Menu: LoRaWAN: Server Address: TTN-router-US. Copy Gateway ID to register gateway   
 on TTN.

Static IP: 2022: IONOS cloud chirpstack Oct.2022: 62.151.182.141  
(2021) Hughesnet: 72.170.244.111  
 LoRa: Frequency Plan: US  
 Frequency Sub Band: 2

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GATEWAY for TTN (reality check) and CHIRPSTACK \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Virgin Dragino Gateway:

Connect PC wireless to Dragino WiFi Access Point. Pw: dragino+dragino.

Browser to 10.130.1.1. login: root pw dragino (or is it dragino+dragino?).

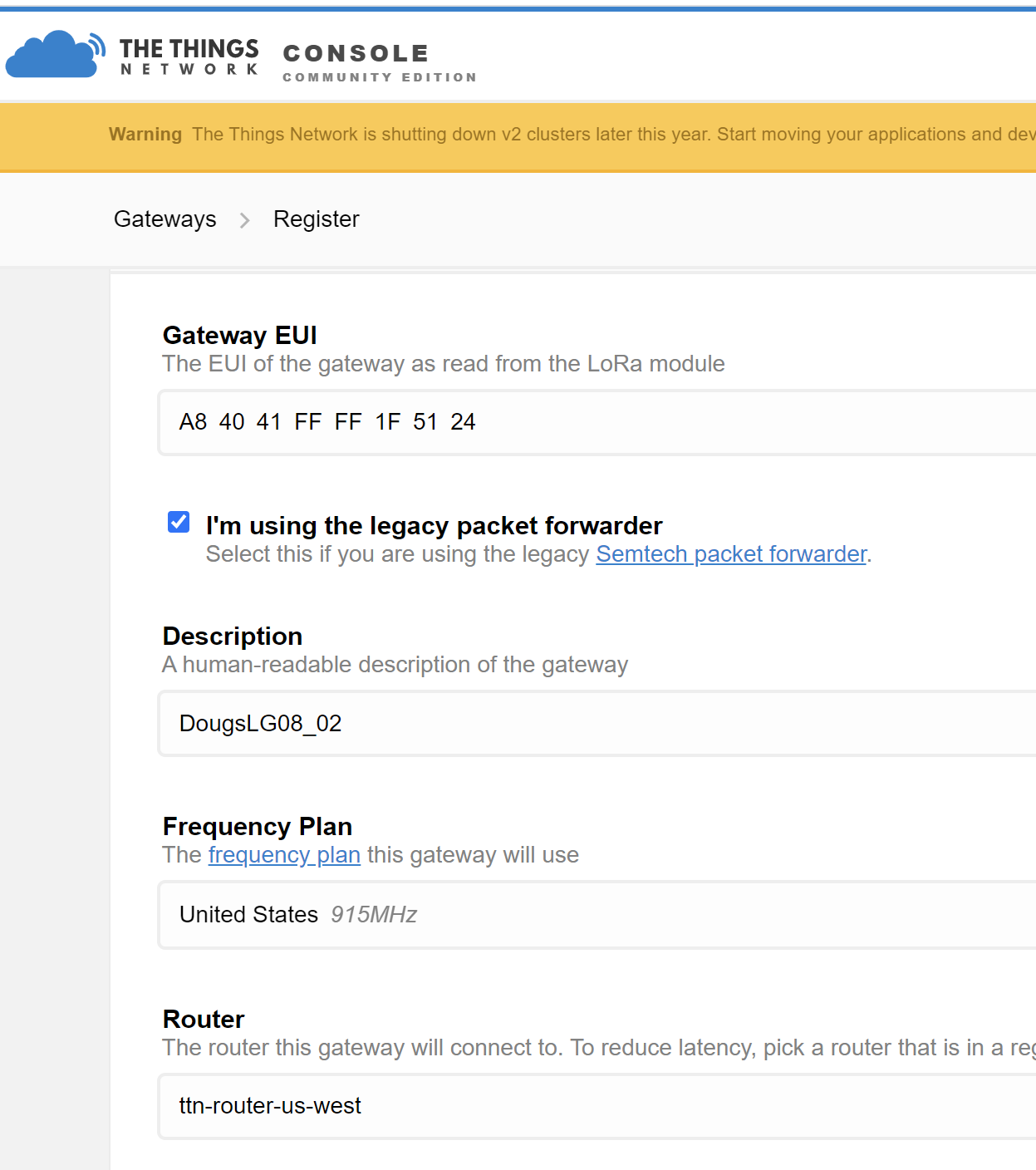
Menu: LoRaWAN: Gateway ID: copy it (ie a84041ffff1f5124)

Server Address: set to TTN-router-US, US 902-928 (for TTN reality check)

Save&Apply

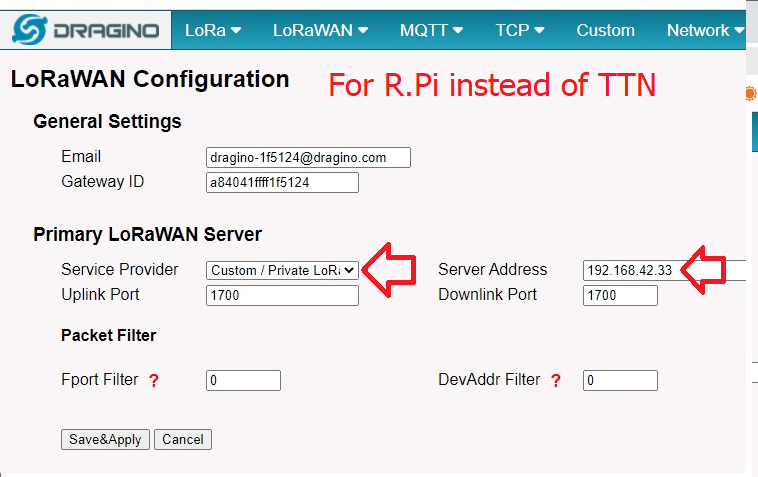
Menu: Network:WiFi  
 Add check to Client Settings.  
 WiFi Survey: Drop down and select. Add password. Save and apply.  
 Menu: Home: should see four green checks.  
  
 Dragino Menu:   
 LoRaWAN:Primary LoRaWAN Server: Service Provider: Custom / Private(for R.Pi Chirpstack)  
 Server Address: localhost  
 Uplink Port: 1700 Downlink Port: 1700  
 Save and Apply  
   
Copy Gateway ID to register gateway on TTN. (for TTN reality check)  
 LoRa: Frequency Plan: US  
 Frequency Sub Band: 2

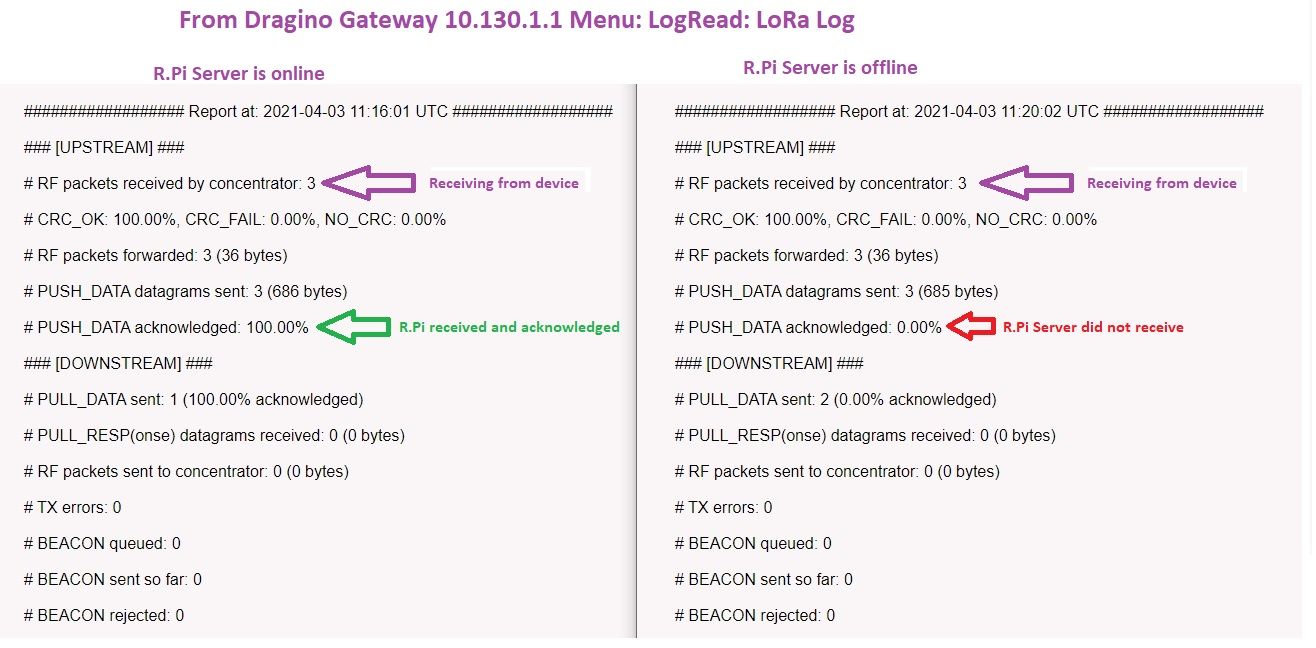
From TTN Console: Gateways: (for TTN reality check)



On TTN click ‘Register Gateway’. It should appear as green Status: connected.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GATEWAY for CHIRPSTACK running on LAPTOP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

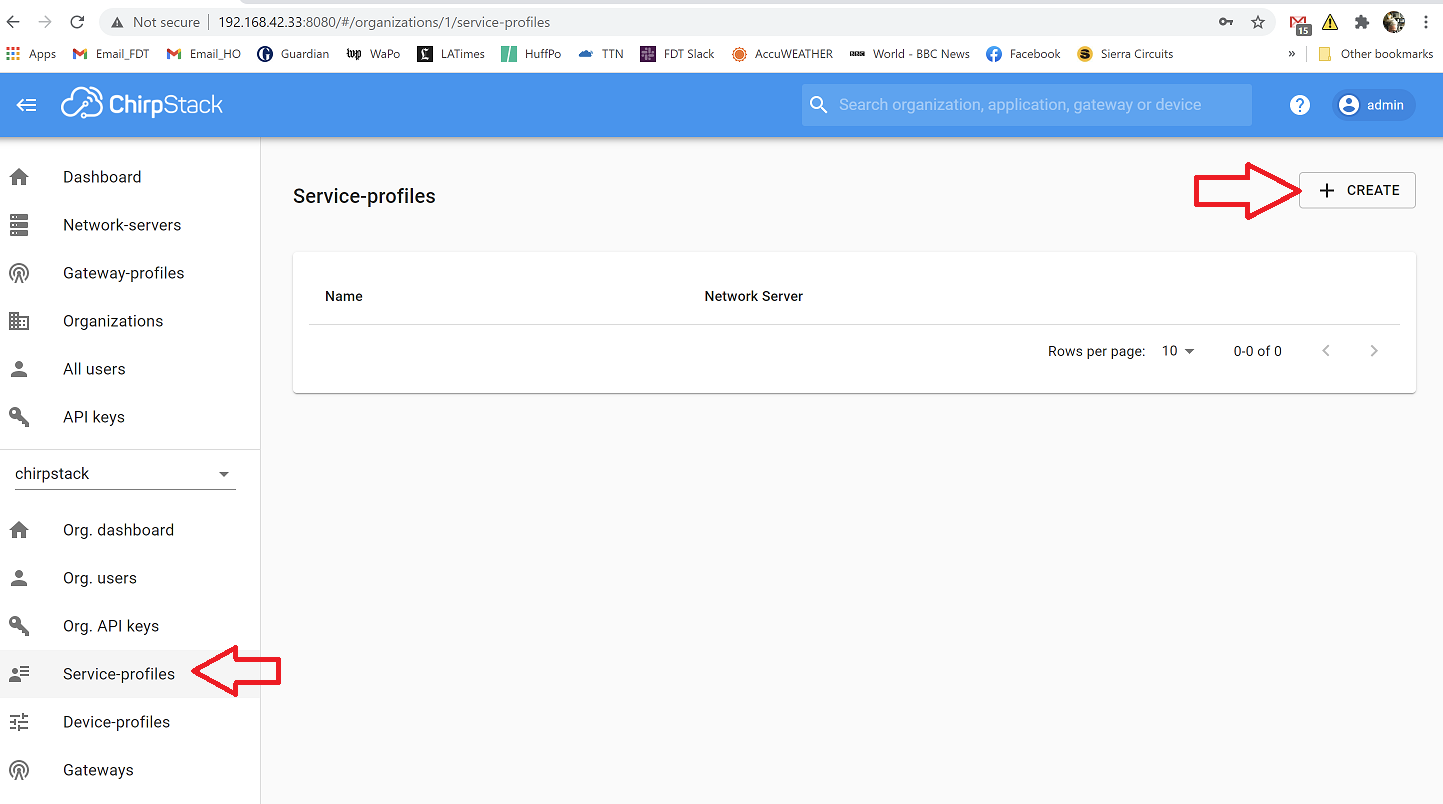
192



/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* REGISTER GATEWAY ON CHIRPSTACK NETWORK SERVER \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

From host PC on the same WLAN as RPi, browser to <http://192.168.42.33:8080>. Un: admin, pw: admin.  
If a TLS certificate has been generated and installed, then use https:// instead of http://.

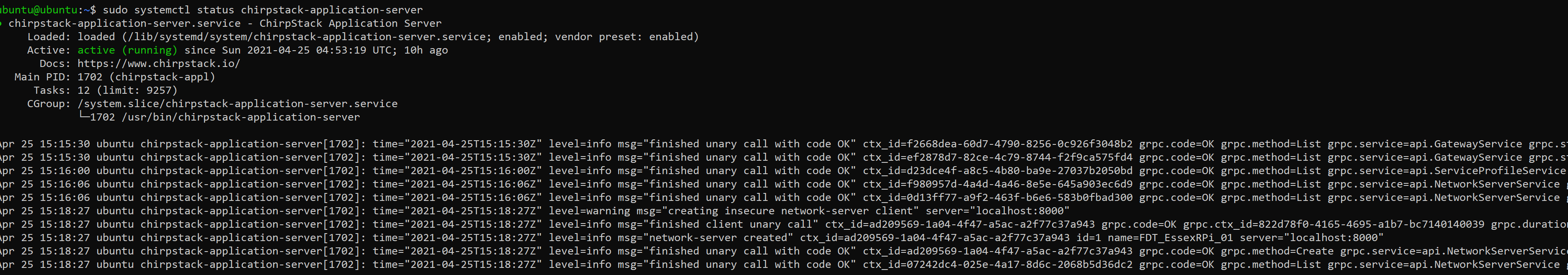
With a virgin Chirpstack, must first create a Service Profile, AND connect the Application Server to an instance of Network Server.  
Click to add Service Profile, and then click to Add Network-Server.

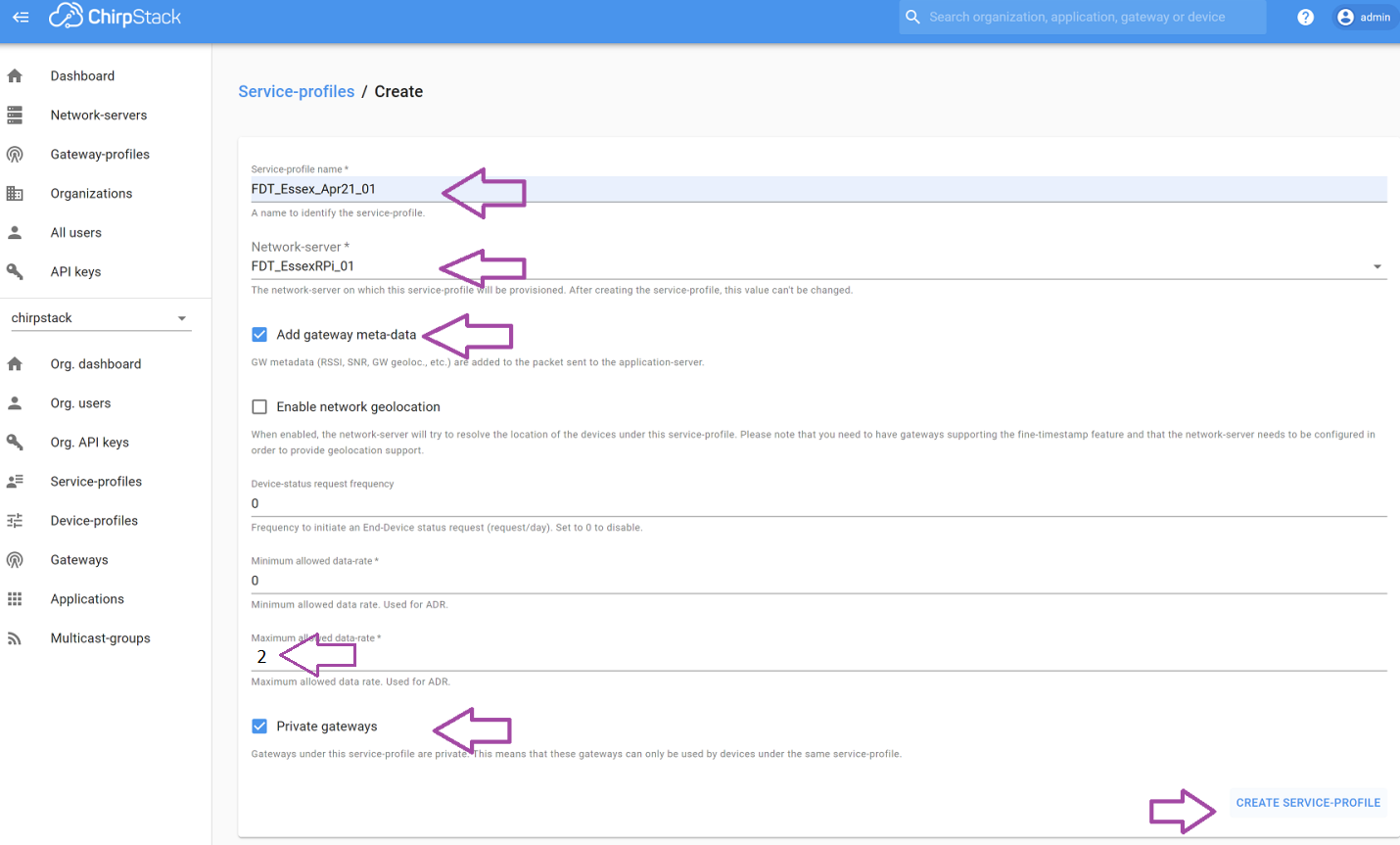


It will ask to connect a network server:



2022: cloud installation is also bind=”0.0.0.0:8000”

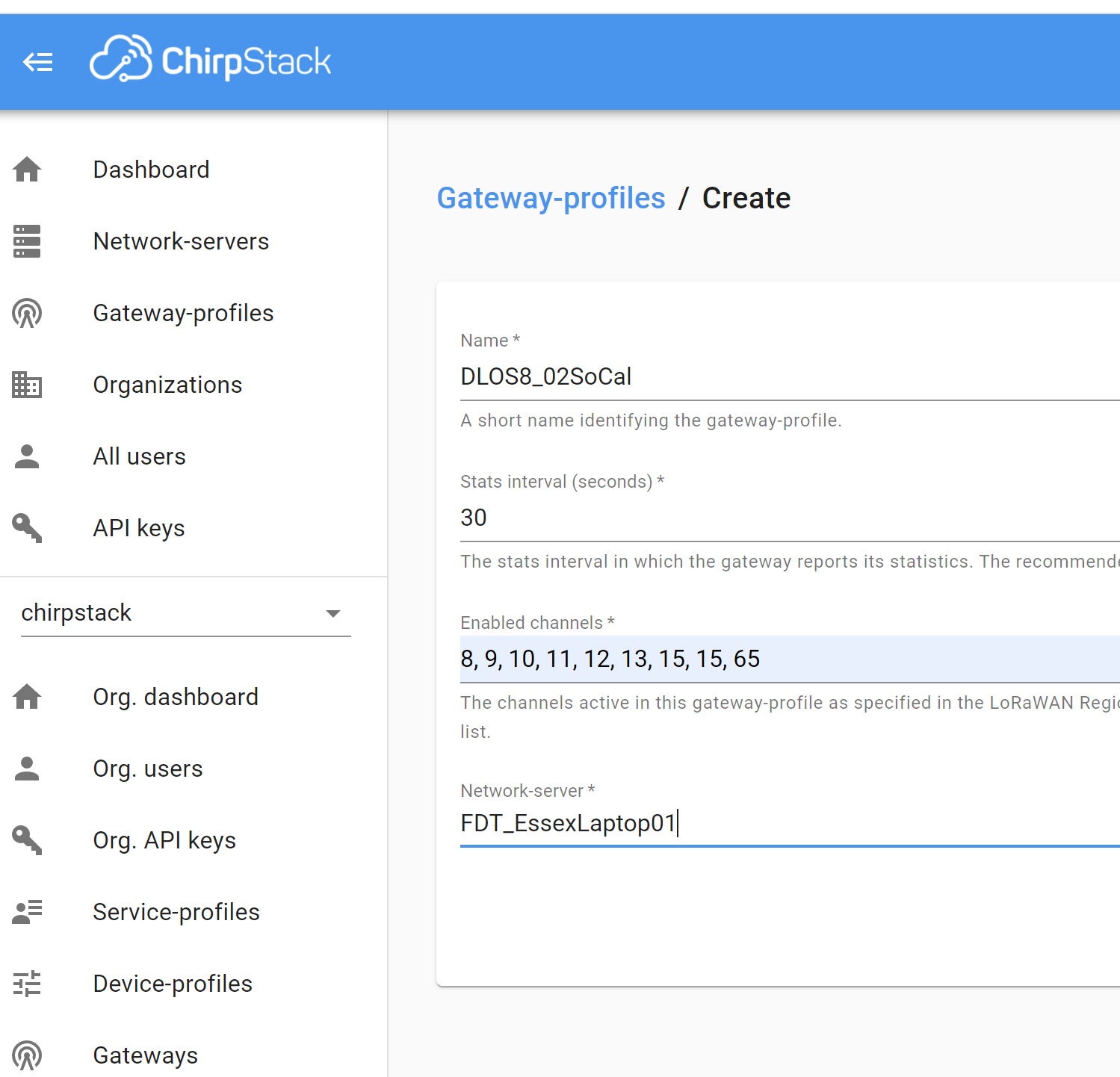
Application-server status had new msgs after adding a Network-server (network-server status msgs didn’t change):

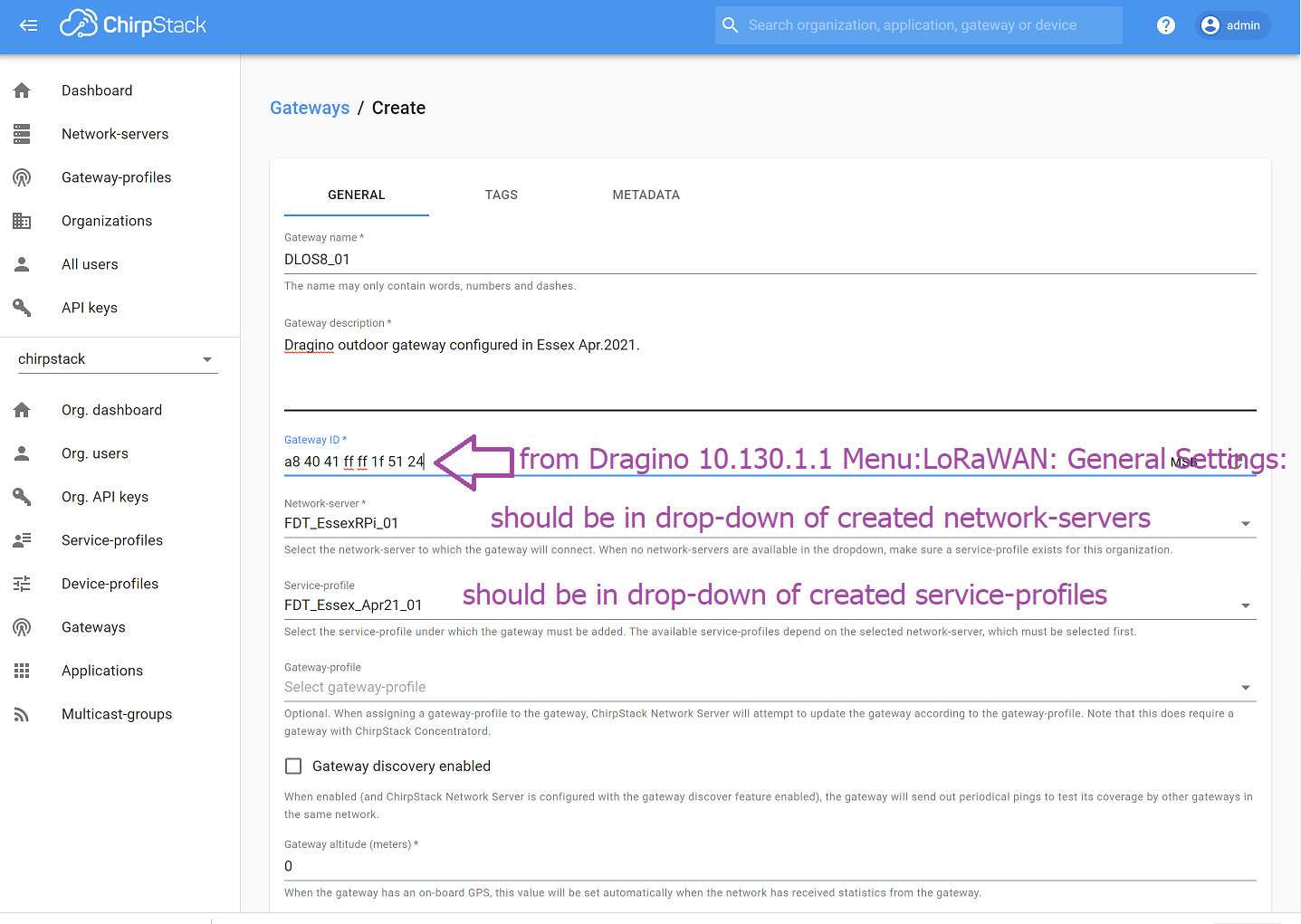
Then add Service-profile.

To see 51-byte data to arrive from devices, Maximum allowed data rate must be set to 2.

[status msgs did not change after creating Service-profile]

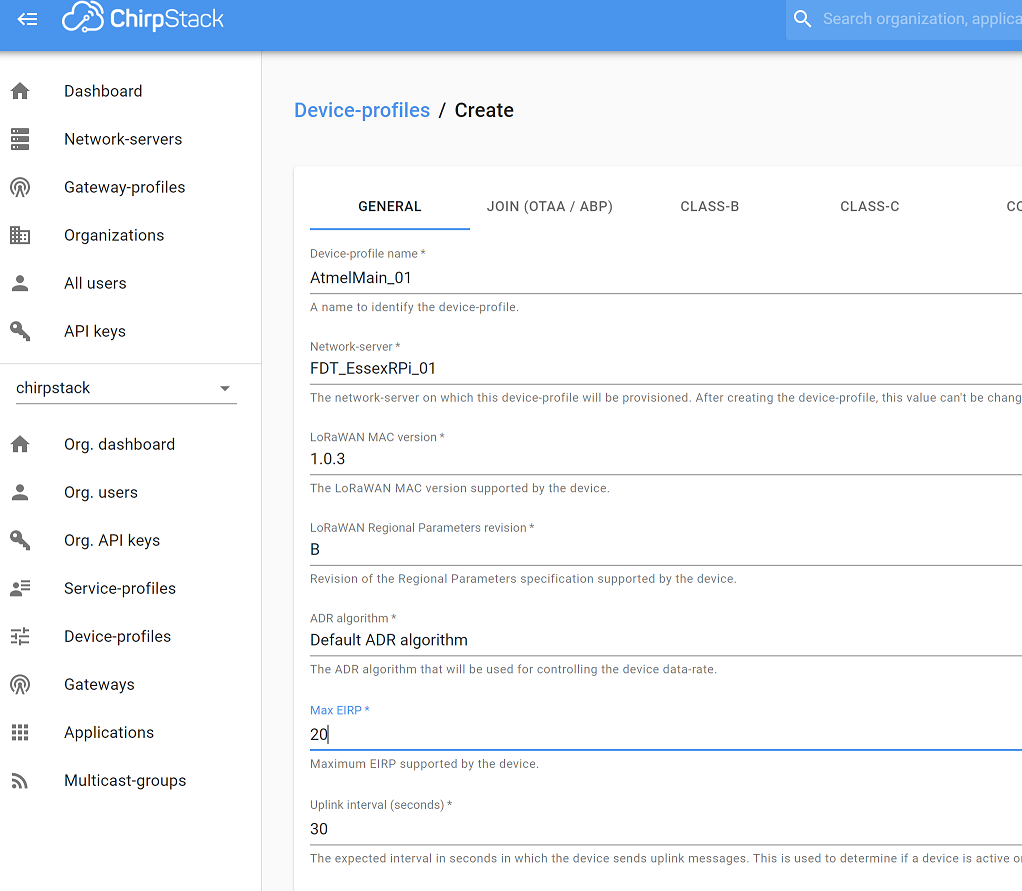
There is a menu option to create a Gateway Profile. Gateway Profiles are optional. It allows for configuring groups of gateways.

Left Menu:Gateways. Upper right: +CREATE.  
  


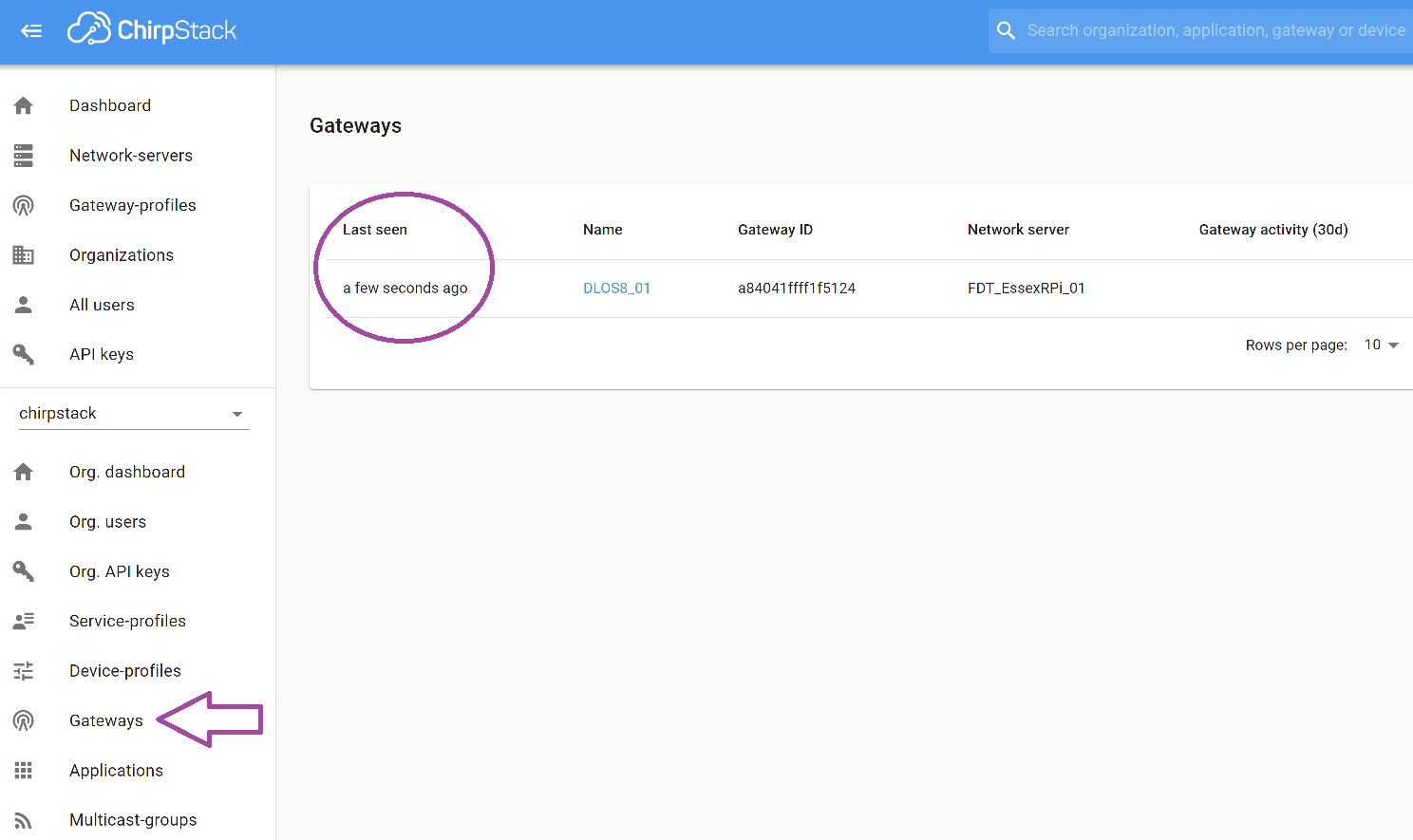


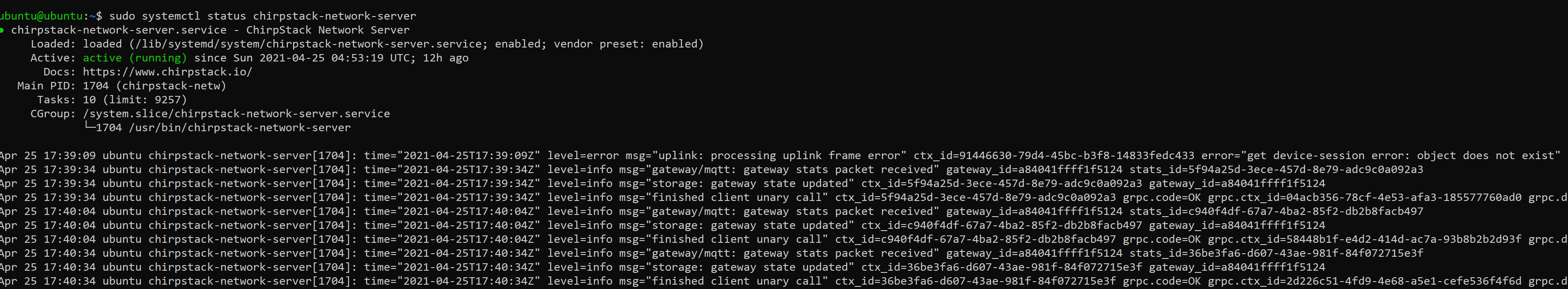
[msgs didn’t change after adding gateway]  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* DEVICE PROFILE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

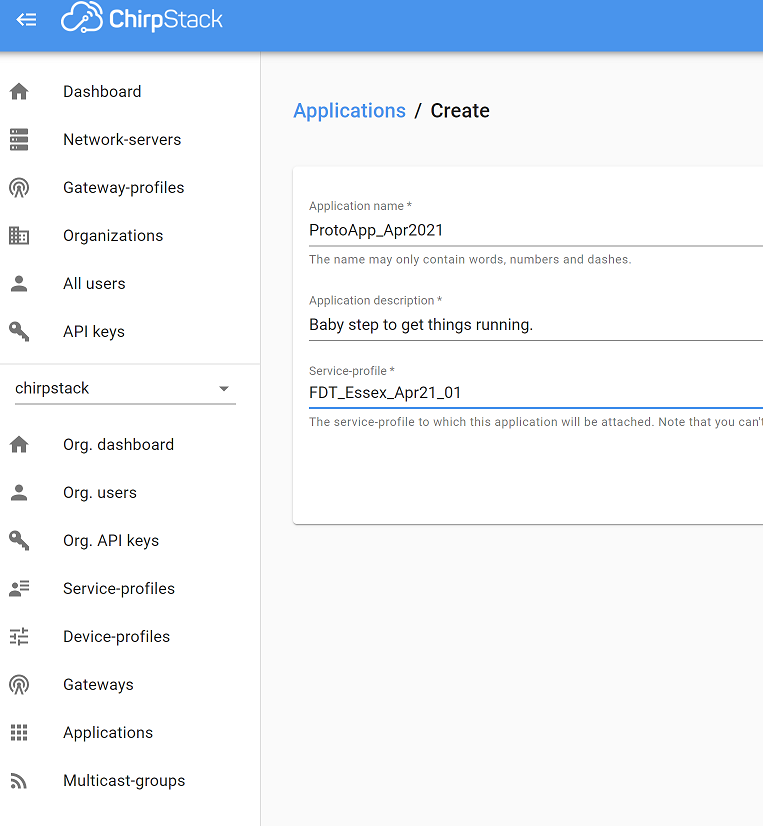
The Device Profile page includes access to the CODEC where we can add custom JavaScript decoding functions.



Oct 2022: changed Max EIRP to 22

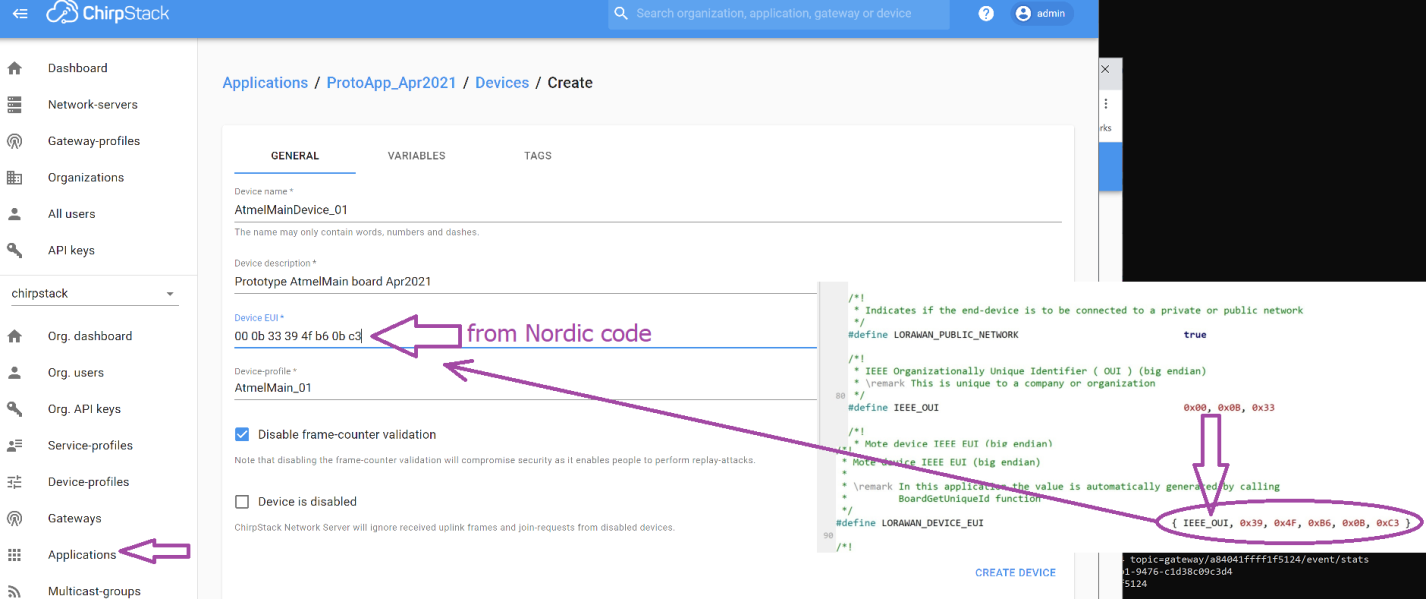
[At this point, if a gateway is online, the ‘Gateways’ screen will show activity. And when I turned on a device, the network-server log showed some uplink messages.]



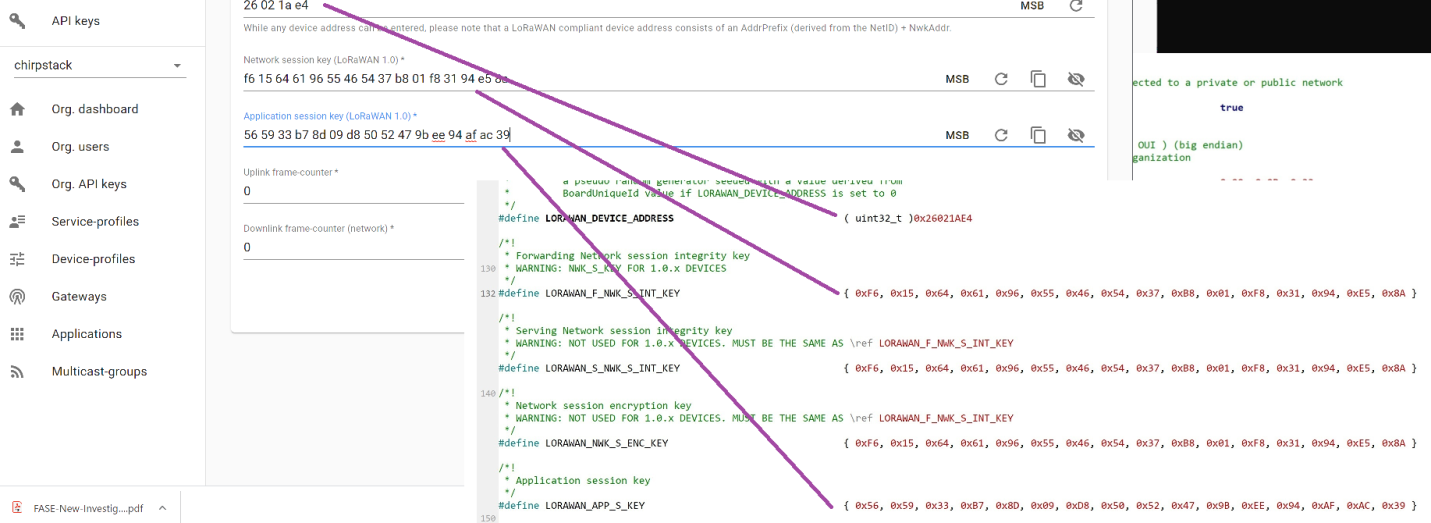
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* CREATE APPLICATION \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  


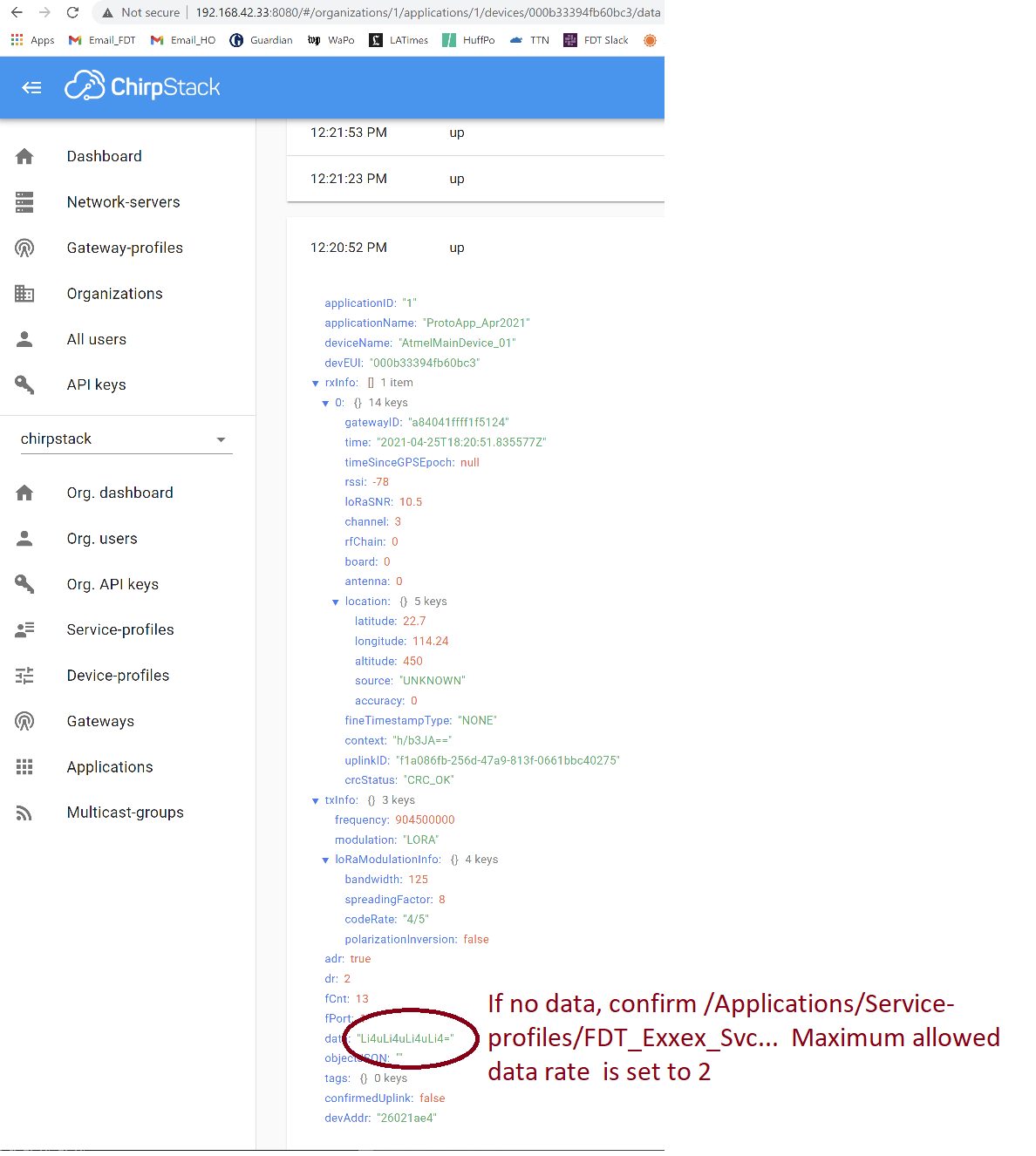
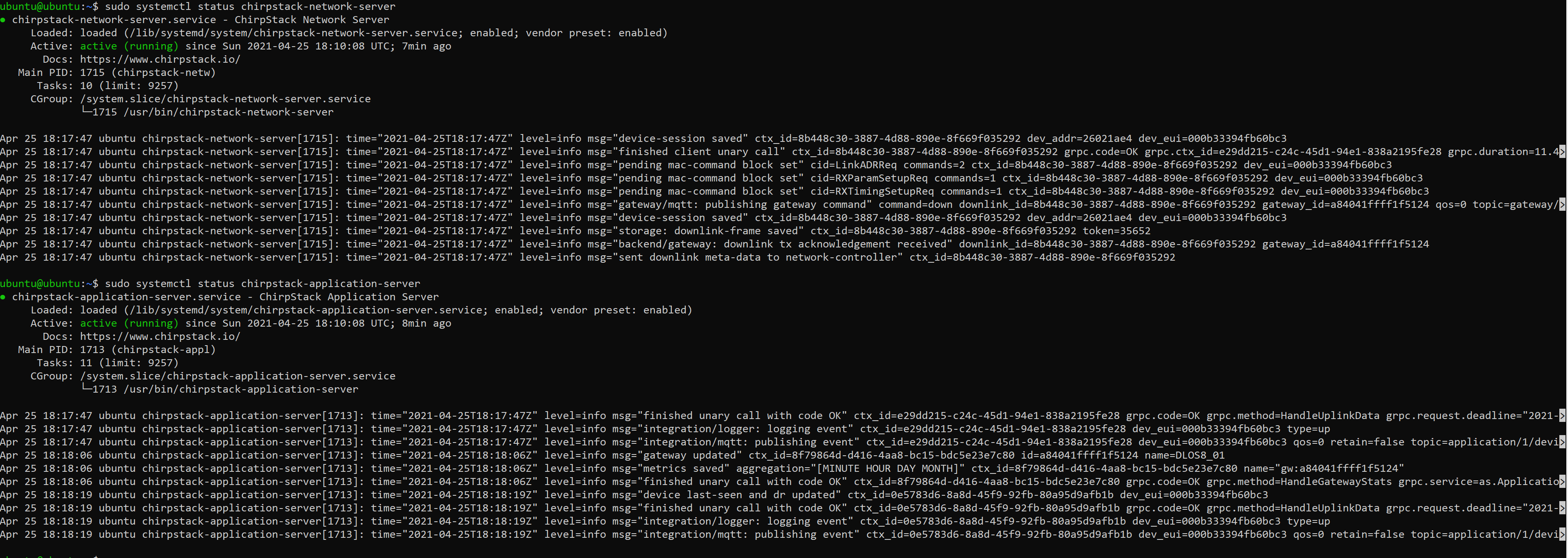
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* CREATE DEVICE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

(also see CREATE DEVICE PROFILE above)  
After creating the application, click on the application and go to Menu: Devices. +Create.



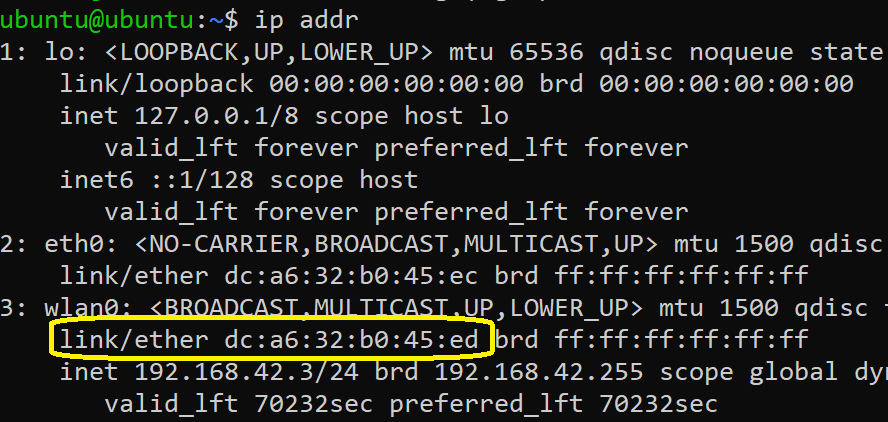
Click create, it will open the next screen:



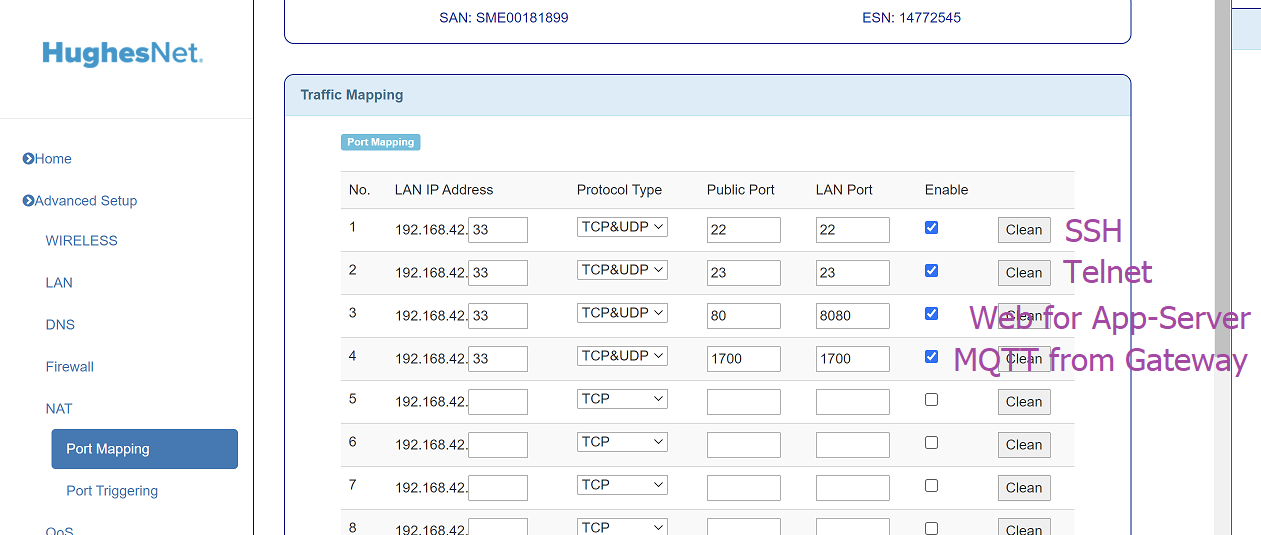
[Had to start Nordic SW with debug cable before saw data arriving at R.Pi. But after got data through the status msgs looked like this:]

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* STATIC IP FOR R.PI \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
 For a Dragino to access an R.Pi on the cloud, the Dragino needs to be on a LAN and router with a static IP. This needs to be established by assigning an external Internet static IP.  
  
 When both the Dragino gateway and R.Pi are on the same local network, then only need to set a local static IP. Two ways to do it: 1) configuring the .yaml file on the R.Pi, or 2) configuring the local router to always assign the same IP to the R.Pi. Option 2 is preferred. Instructions of configuring from the R.Pi’s .yaml file at the end of this document.

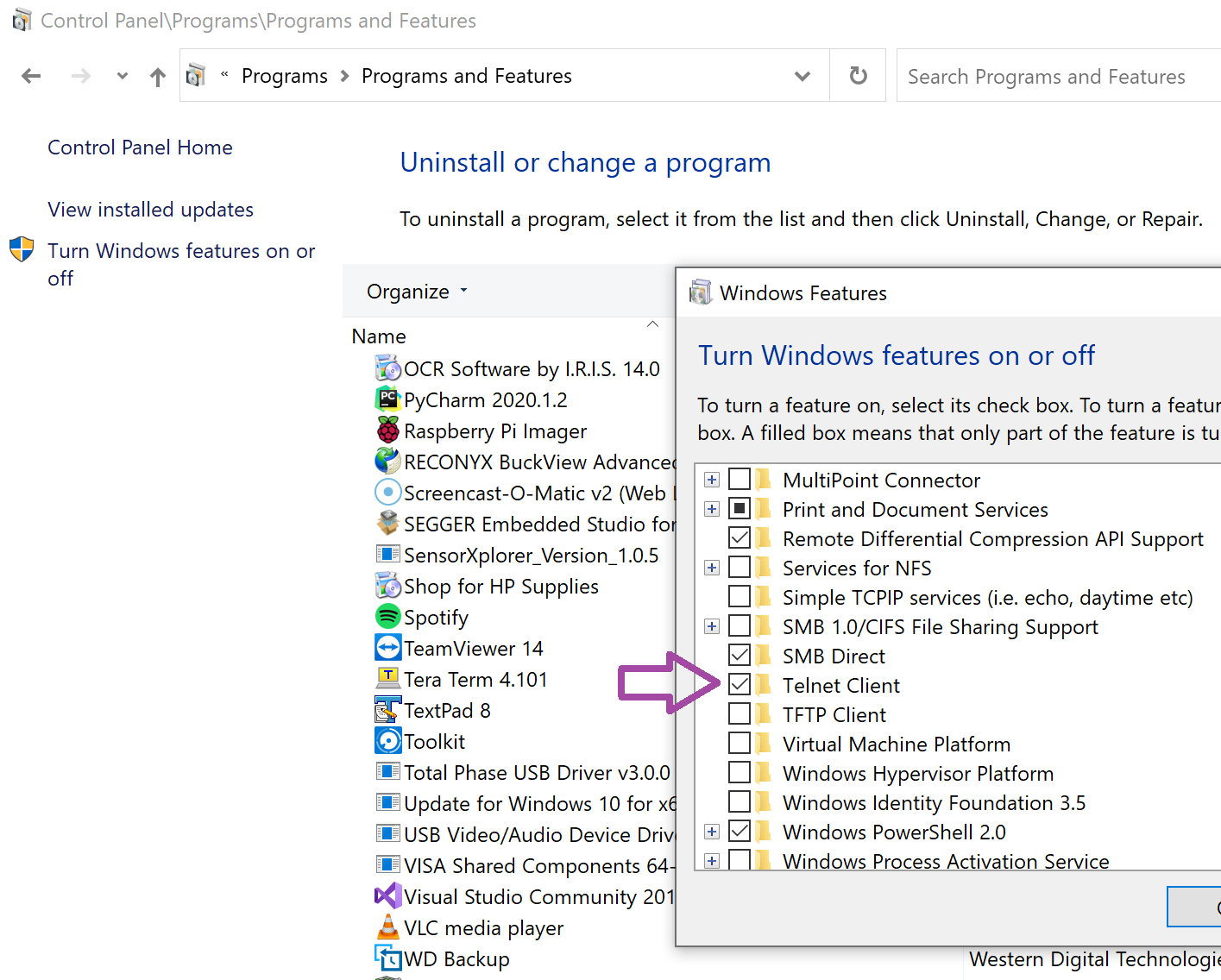
From Hughesnet setup webpage 192.168.0.1, WiFi settings, pw “admin”.

Get R.Pi WiFi MAC with “ip addr”

To confirm, power cycle R.Pi, log-in, new IP should appear as “IPv4 address for wlan0: xxx.xxx.x.xx”



/\*\*\*\*\*\* TEST EXTERNAL PORT FORWARDING \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
Windows Control Panel: Programs: Programs and Features: Turn Windows features on or off.

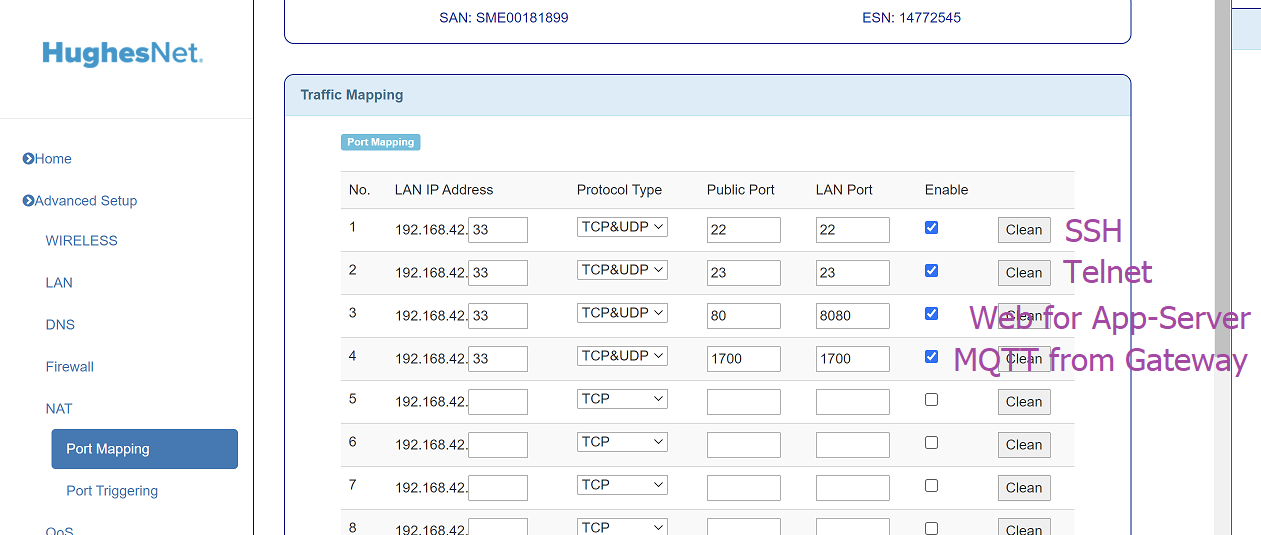


/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INSTALL TELNET ON R.PI (to test port fwd) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

On R.Pi:

sudo apt-get install telnetd –y  
sudo systemctl status inetd  
[do we need to enable? sudo systemctl enable inetd ]

If hughesnet router forwards ssh on Public Port 22 LAN 22, then from any other computer:   
 ssh [ubuntu@72.170.244.111](mailto:ubuntu@72.170.244.111)  
Or if hughesnet routher forwards telnet on Public Port 23 LAN 23, then from any other computer:   
 telnet 72.170.244.111 login doug pw.   
to leave, “exit”.



/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INSTALL GATEWAY BRIDGE ON DRAGINO \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**[Apr2021: seems to be working with the gateway bridge installed on the R.Pi instead of the Dragino. Try skipping this section during the next Dragino configuration.]**

Use laptop to connect Dragino’s WiFi Access Point.

Also run an Ethernet cable to internet router.

ssh [root@10.130.1.1](mailto:root@10.130.1.1)  
pw: dragino

go to <https://artifacts.chirpstack/io/vendor/dragino> to find the latest version of gateway bridge for Dragino.

cd /tmp

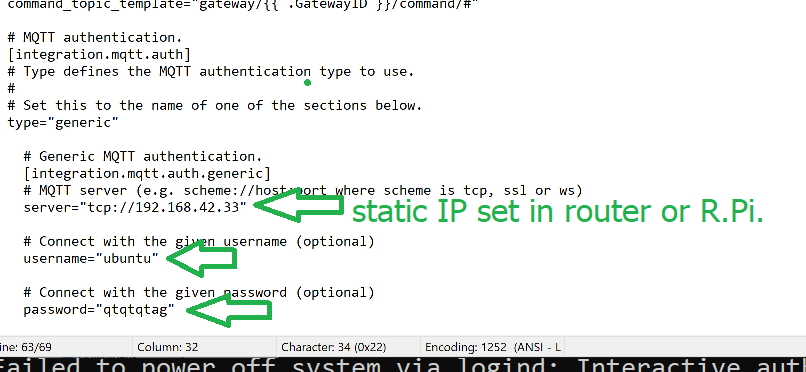
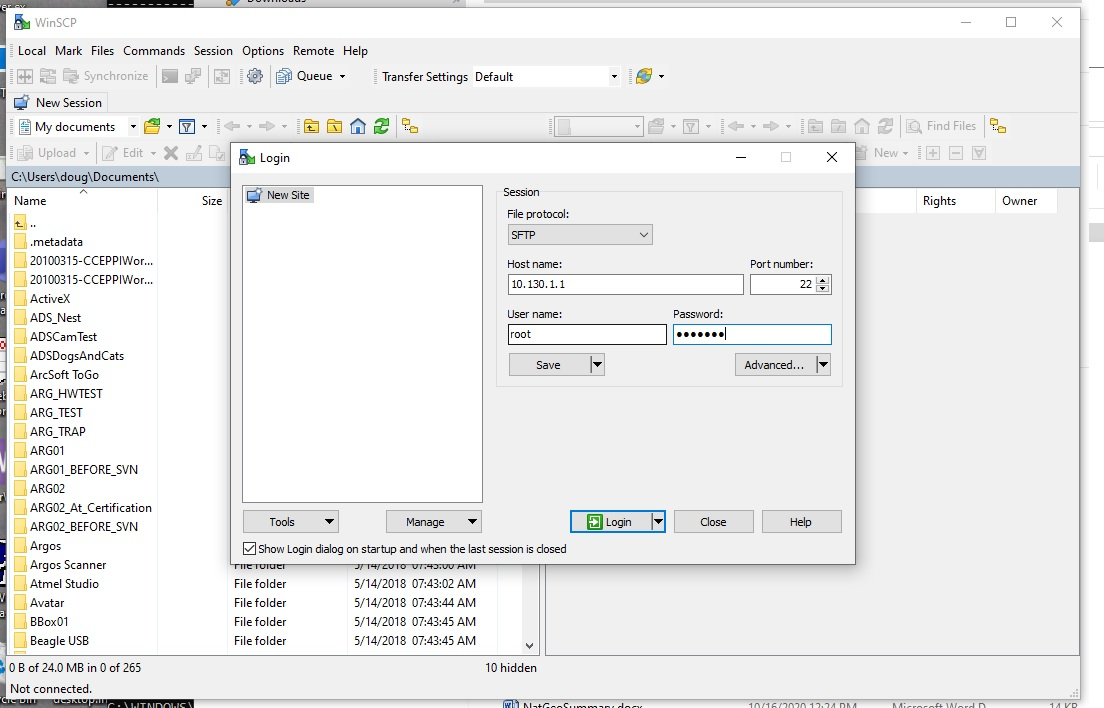
wget <https://artifacts.chirpstack.io/vendor/dragino/LG308/chirpstack-gateway-bridge_3.10.0-r1_mips_24kc.ipk> [but using the latest version when surfed there]

opkg install chirpstack-gateway-bridge\_3.10.0-r1\_mips\_24kc.ipk

**Note:** In case of an upgrade, it is recommended to first uninstall the chirpstack-gateway-bridge package using opkg remove ... and then install the new version using opkg install .... Configuration files will be maintained.

Download and run WinSCP on a host PC. Connect to the Dragino by establishing a WiFi connection to the Dragino, then launch WinSCP and enter Host: 10.130.1.1 pw: dragino.

Navigate to /etc/chirpstack-gateweay-bridge and open the .toml file for editing.



/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ERROR LOG on R.Pi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

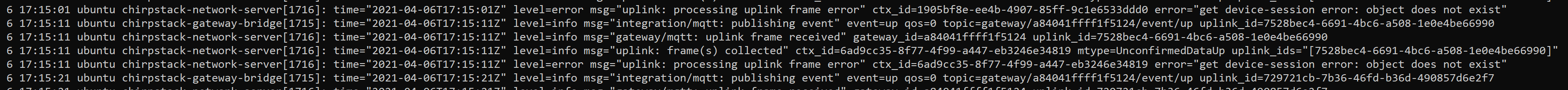
/var/log/messages (easy to view on WinSCP after logging into R.Pi with a PC)

Alternatively, with SSH to R.Pi: sudo tail -f /var/log/syslog [web says “messages” instead of “syslog”. My Ubuntu 20.04 has “syslog”

To filter messages for a particular service:

sudo tail -f /var/log/syslog |grep chirpstack-concentratord

After network-server, Service Profile, Gateway and Device are added:



/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Devices \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Must be OTAA or ABP. I’m using ABP for now.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Decoding Data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

When selecting the Custom JavaScript codec functions option, you can write your own (JavaScript) functions to decode an array of bytes to a JavaScript object and encode a JavaScript object to an array of bytes. Package [otto](https://github.com/robertkrimen/otto), which targets ES5, is used as a JavaScript interpreter, so ES6 features (e.g. Typed Arrays) are not supported.

#### Decoder function skeleton

// Decode decodes an array of bytes into an object.

// - fPort contains the LoRaWAN fPort number

// - bytes is an array of bytes, e.g. [225, 230, 255, 0]

// - variables contains the device variables e.g. {"calibration": "3.5"} (both the key / value are of type string)

// The function must return an object, e.g. {"temperature": 22.5}

function Decode(fPort, bytes, variables) {

return {};

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ORGANIZATIONS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

An organization can be used to let organizations or teams manage their own applications and optionally their own gateways.

An organization can have:

* Service Profiles
* Device Profiles
* Gateways (when allowed)
* Applications
* Users

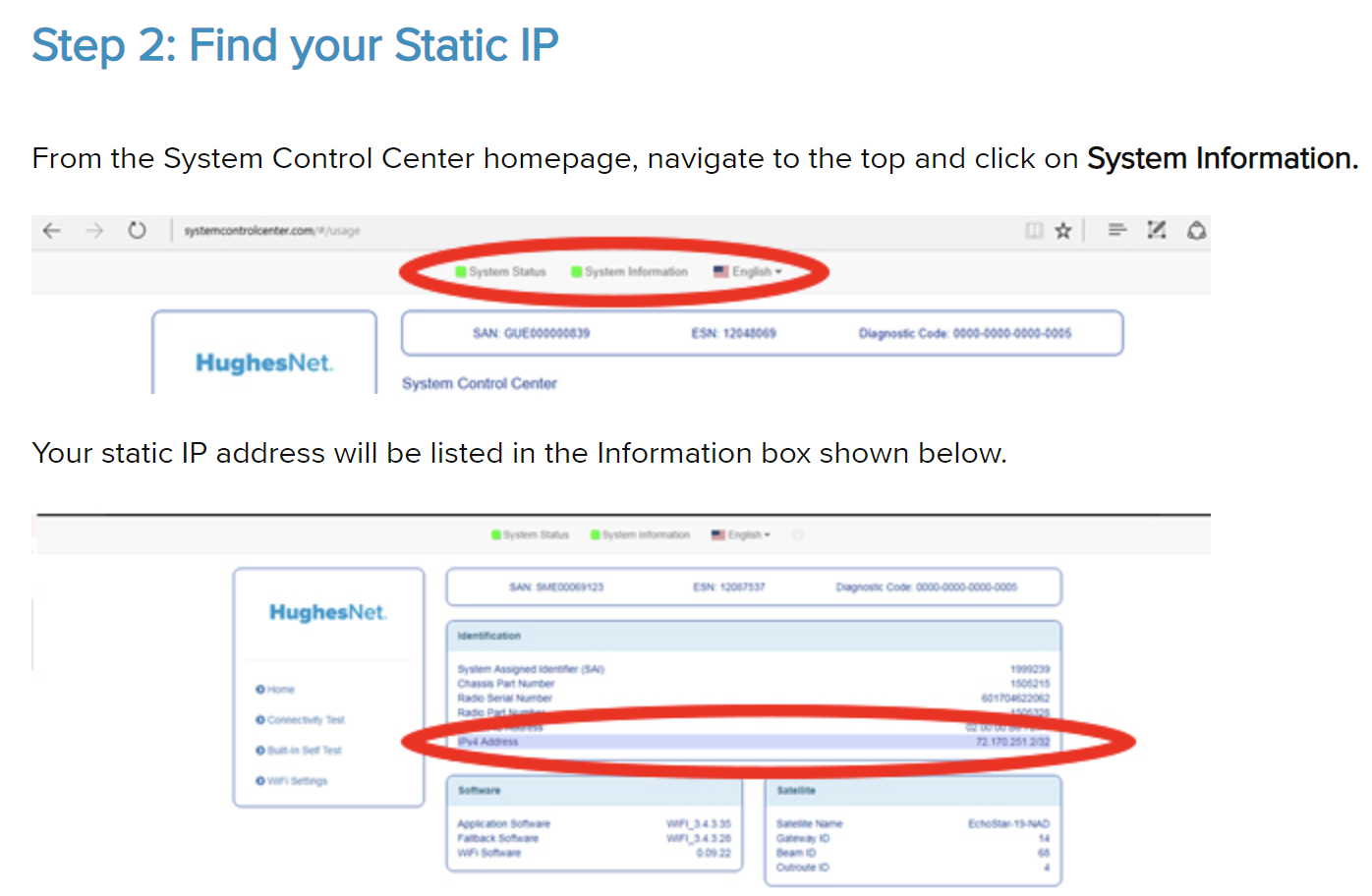
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* BACKUP COPY OF R.Pi uSD CARD \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Use Win32DiskImager utility to clone the uSD card.

https://sourceforge.net/projects/win32diskimager/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* STATIC IP FOR HUGHESNET \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Hughesnet static IP for business acct (April 24, 2021) [www.IP4.me](http://www.IP4.me) doesn’t work.  
Hughesnet modem (per instructions below under System Information) reports IPv4 is 72.170.244.111/32.  
Or Hughesnet modem under WiFi Settings says WAN IP: 72.170.244.111.



## Configuring Static IP address on Ubuntu Server [#](https://linuxize.com/post/how-to-configure-static-ip-address-on-ubuntu-20-04/#configuring-static-ip-address-on-ubuntu-server)

## [not needed if local wireless router is configured to issue a static IP for the R.Pi.]

On Ubuntu 20.04, the system identifies network interfaces using ‘predictable network interface names’.

The first step toward setting up a static IP address is identifying the name of the ethernet interface you want to configure. To do so, use the [ip link](https://linuxize.com/post/linux-ip-command/) command, as shown below:

ip linkCopy

The command prints a list of all the available network interfaces. In this example, the name of the interface is ens3:

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

2: ens3: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc fq\_codel state UP mode DEFAULT group default qlen 1000

link/ether 08:00:27:6c:13:63 brd ff:ff:ff:ff:ff:ff

Copy

Netplan configuration files are stored in the /etc/netplan directory. You’ll probably find one or more YAML files in this directory. The name of the file may differ from setup to setup. Usually, the file is named either 01-netcfg.yaml, 50-cloud-init.yaml, or NN\_interfaceName.yaml, but in your system it may be different.

If your Ubuntu cloud instance is provisioned with cloud-init, you’ll need to disable it. To do so create the following file:

sudo nano /etc/cloud/cloud.cfg.d/99-disable-network-config.cfgCopy

/etc/cloud/cloud.cfg.d/99-disable-network-config.cfg

**network**: {**config**: disabled}

Copy

To assign a static IP address on the network interface, open the YAML configuration file with your [text editor](https://linuxize.com/post/how-to-install-visual-studio-code-on-ubuntu-20-04/) :

sudo nano /etc/netplan/01-netcfg.yamlCopy

/etc/netplan/01-netcfg.yaml

**network**:

**version**: 2

**renderer**: networkd

**ethernets**:

**ens3**:

**dhcp4**: **yes**

Copy

Before changing the configuration, let’s explain the code in a short.

Each Netplan Yaml file starts with the network key that has at least two required elements. The first required element is the version of the network configuration format, and the second one is the device type. The device type can be ethernets, bonds, bridges, or vlans.

The configuration above also has a line that shows the renderer type. Out of the box, if you installed Ubuntu in server mode, the renderer is configured to use networkd as the back end.

Under the device’s type (ethernets), you can specify one or more network interfaces. In this example, we have only one interface ens3 that is configured to obtain IP addressing from a DHCP server dhcp4: yes.

To assign a static IP address to ens3 interface, edit the file as follows:

* Set DHCP to dhcp4: no.
* Specify the static IP address. Under addresses: you can add one or more IPv4 or IPv6 IP addresses that will be assigned to the network interface.
* Specify the gateway.
* Under nameservers, set the IP addresses of the nameservers.

/etc/netplan/01-netcfg.yaml

**network**:

**version**: 2

**renderer**: networkd

**ethernets**:

**ens3**:

**dhcp4**: **no**

**addresses**:

- 192.168.121.221/24

**gateway4**: 192.168.121.1

**nameservers**:

**addresses**: [8.8.8.8, 1.1.1.1]

Copy

When editing Yaml files, make sure you follow the YAML code indent standards. If the syntax is not correct, the changes will not be applied.

Once done, save the file and apply the changes by running the following command:

sudo netplan applyCopy

Verify the changes by typing:

ip addr show dev ens3Copy

2: ens3: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc mq state UP group default qlen 1000

link/ether 08:00:27:6c:13:63 brd ff:ff:ff:ff:ff:ff

inet 192.168.121.221/24 brd 192.168.121.255 scope global dynamic ens3

valid\_lft 3575sec preferred\_lft 3575sec

inet6 fe80::5054:ff:feb0:f500/64 scope link

valid\_lft forever preferred\_lft forever

Copy

That’s it! You have assigned a static IP to your Ubuntu server.

<https://sourceforge.net/projects/win32diskimager/files/latest/download>

TO copy the database from the Linux server (this worked December 2021)  
sudo –u postgres pg\_dump chirpstack\_as > chirpstack\_as.tar

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OPENMV INSTALL CLONE ON LINUX \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Cd into repository folder.

From command terminal in Linux: git clone –recursive <https://github.com/openmv/openmv.git>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PGAdmin4 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Download from the PGAdmin site. Can install as a web service or locally.  
To install, copy the sudo… commands to a terminal.

When using a local installation of PGAdmin, Servername/address is simply localhost.

User: chirpstack\_as pw gnpass

Static IP# at IONOS =  62.151.182.141

\* I anticipate you'll have to open the firewall for ports to postgreSQL (client/server) and for the

   Dragino gateway.

\* Please change your password ("doug") after you get it. I don't need a copy, as I have root

\* If you change the passwords on any of the other stuff, please let me know so I can record

  in my password vault.

1. Application Server Admin

WebuURL   ; [http://cloud.fdtchirp.com:8080](http://cloud.fdtchirp.com:8080/)

Username   : admin

Password    : FDTadm1n!   (note "1" and "!" at end)

2. Postgres databases

    Note: all are available in the configuation files (".toml").

-- application server  
-- network server  
-- gateway server

3. Cloud server admin (IONOS) (among other things you can setup firewall ports)

WebURL          : <https://login.ionos.com/>

Username        : [ionos@cyberchuck.org](mailto:ionos@cyberchuck.org)

Password         : !lI03mcXrAjzGuX3

4. PGADMIN4

WebURL          : <http://cloud.fdtchirp.com/pgadmin4>

Username        : charlie@fielddata.tech

Password         : pgAdmin-main1

5. Password for "doug"

Hostname    : [cloud.fdtchirp.com](http://cloud.fdtchirp.com/)

Username    : doug

Password    : M00nrock!

[ Use SSH as in "ssh [doug@cloud.fdtchirp.com](mailto:doug@cloud.fdtchirp.com)" ]

Logout

sshLs

Cd

As.toml:

Chirpstack\_as\_events fdtasedb