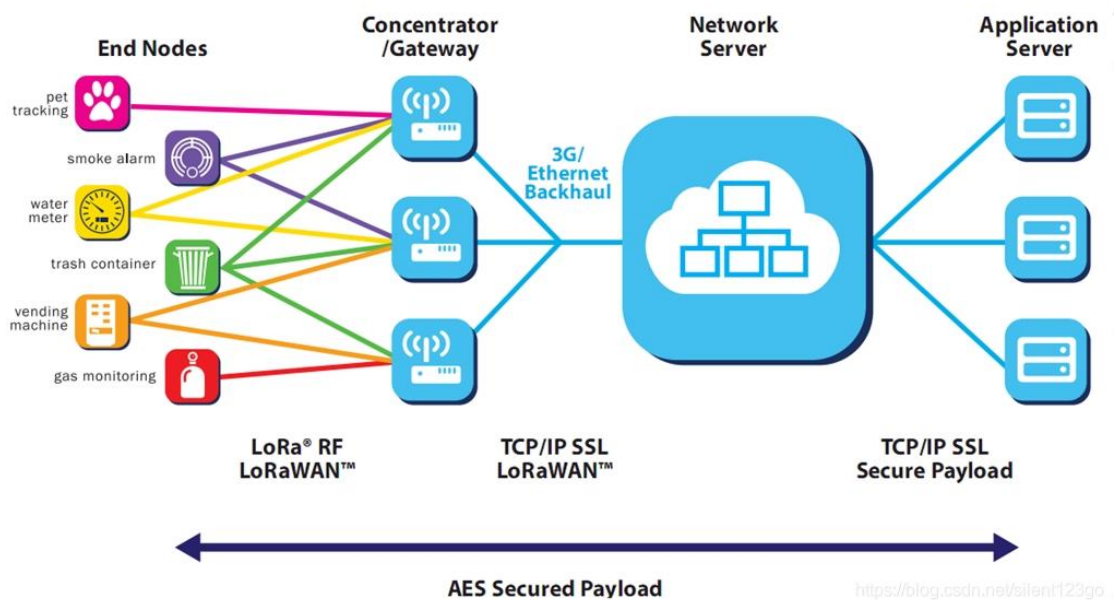
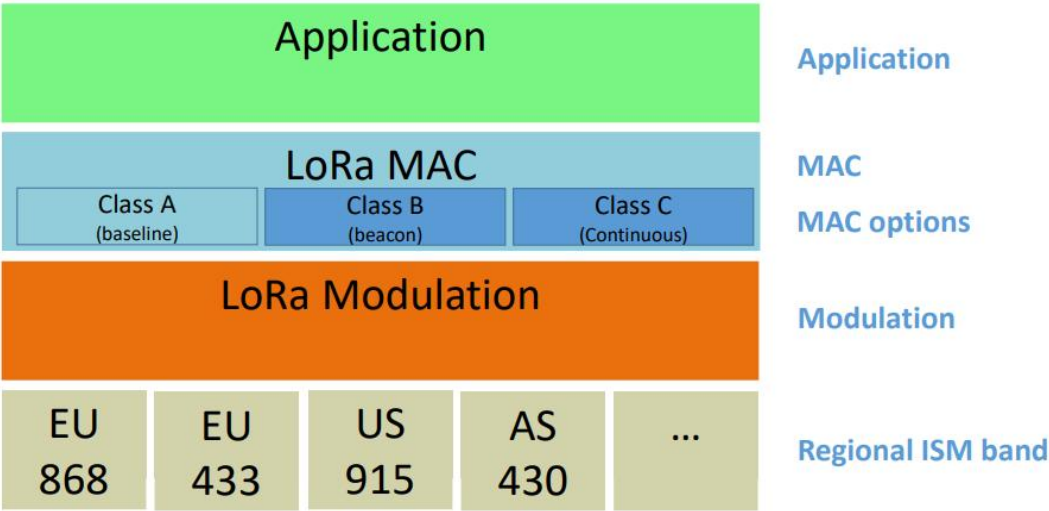


使用 E890 和 E78 建立 LoRaWAN 网络(chirpstack v4)

LoRa 调制是一种低功耗广域网通信技术，是 Semtech 公司专有的一种基于扩频技术的超远距离无线传输技术。LoRaWAN 是为 LoRa 远距离通信网络设计的一套通讯协议和系统架构。它是一种媒体访问控制（MAC）层协议。



LoRaWAN 在整个流程的中充当 MAC 的功能，而 LoRa 调制充当物理层。



LoRaWAN 网络主要优势体现在低成本、广域连接和低功耗，同时具有较多的开源平台可供使用。下文将简单描述使用亿佰特 E890-470LG11 和 E78-470LN22S 同开源服务器平台 -Chirpstack 快速搭建本地 LoRaWAN 网络。

一、 Chirpstack 服务器简介和搭建

Chirpstack 是一款多组件的、部署简单的开源服务器，同时也是使用最广泛的 LoRaWAN 服务器。本次安装使用 Ubuntu22.04。需要使用到的软件有 git vim 请自行安装。

1. 安装环境

快速搭建验证平台时直接使用 Chirpstack-docker 这个项目，可快速部署服务器。
在要搭建的服务器上安装 docker-compose。

在 Ubuntu 终端输入：sudo apt-get install -y docker-compose，输入 docker-compose version 时，会显示 docker-compose 版本，此时安装成功。

```
Processing triggers for man-db (2.10.2-1) ...
min@ubuntu:~$ docker-compose version
docker-compose version 1.29.2, build unknown
docker-py version: 5.0.3
CPython version: 3.10.6
OpenSSL version: OpenSSL 3.0.2 15 Mar 2022
```

2. 获取文件

获取 chirpstack-docker 文件有两种办法，

第一种直接从 github 下载，然后拷贝到服务器上进行解压即可。

第二种使用 git 指令获取。github 地址：

<https://github.com/chirpstack/chirpstack-docker.git>

使用 git 指令获取项目，输入指令：

git clone <https://github.com/chirpstack/chirpstack-docker.git>

```
min@ubuntu:~$ git clone https://github.com/chirpstack/chirpstack-docker.git
Cloning into 'chirpstack-docker'...
remote: Enumerating objects: 397, done.
remote: Counting objects: 100% (232/232), done.
remote: Compressing objects: 100% (89/89), done.
remote: Total 397 (delta 175), reused 158 (delta 140), pack-reused 165
Receiving objects: 100% (397/397), 75.71 KiB | 833.00 KiB/s, done.
Resolving deltas: 100% (239/239), done.
```

3. 切换到项目文件夹中:cd chirpstack-docker

4. 到目前为止已经可以运行 sudo docker-compose up 开始部署服务器.首次运行会花费一点时间下载部署需要的环境.若网络没有问题.下载完成后,将会启动服务器.

5. 在下载 ERROR: Get "https://registry-1.docker.io/v2/": EOF 错误

输入:dig @114.114.114.114 registry-1.docker.io 查看可用 ip

```
min@ubuntu:~/chirpstack-docker$ dig @114.114.114.114 registry-1.docker.io

;<<>> DiG 9.18.1-ubuntu1.3-Ubuntu <<>> @114.114.114.114 registry-1.docker.io
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 48334
;; flags: qr aa rd; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;registry-1.docker.io.      IN      A

;; ANSWER SECTION:
registry-1.docker.io.     60      IN      A       44.205.64.79
registry-1.docker.io.     60      IN      A       3.216.34.172
registry-1.docker.io.     60      IN      A       34.205.13.154

;; Query time: 4 msec
;; SERVER: 114.114.114.114#53(114.114.114.114) (UDP)
;; WHEN: Wed Mar 22 13:52:51 CST 2023
;; MSG SIZE rcvd: 86
```

打开 sudo vim /etc/hosts 文件在文件后面添加

34.205.13.154 registry-1.docker.io

```

127.0.0.1      localhost
127.0.1.1      ubuntu

# The following lines are desirable for IPv6 capable hosts
::1          ip6-localhost ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
34.205.13.154 registry-1.docker.io.

```

输入命令重启 docker: `sudo systemctl restart docker.service`
再次运行: `sudo docker-compose up` 若依旧不成功, 请重复步骤 5

当出现以下提示, 服务器已经成功运行: 在浏览器输入 localhost:8080 即可进入服务器登录界面

```

min@ubuntu: /chirpstack-docker$ sudo vim /etc/hosts
min@ubuntu:~/chirpstack-docker$ sudo docker-compose up
Pulling chirpstack-rest-api (chirpstack/chirpstack-rest-api:4)...
4: Pulling from chirpstack/chirpstack-rest-api
59bf1c3509f3: Pull complete
d81113c0a608: Pull complete
c7ee6feaf9db: Pull complete
Digest: sha256:e6a8dff6e3e643f1c894ec05564c3fbecd5f483704cead66caeb6e64b2c3c90ee
Status: Downloaded newer image for chirpstack/chirpstack-rest-api:4
Creating chirpstack-docker_redis_1 ... done
Creating chirpstack-docker_mosquitto_1 ... done
Creating chirpstack-docker_postgres_1 ... done
Creating chirpstack-docker_chirpstack-gateway-bridge-eu868_1 ... done
Creating chirpstack-docker_chirpstack_1 ... done
Creating chirpstack-docker_chirpstack-rest-api_1 ... done
Attaching to chirpstack-docker_mosquitto_1, chirpstack-docker_postgres_1, chirpstack-docker_redis_1, chirpstack-docker_chirpstack-gateway-bridge-eu868_1, chirpstack-docker_chirpstack_1, chirpstack-docker_chirpstack-rest-api_1
chirpstack-gateway-bridge-eu868_1 | time="2023-03-22T06:06:48.640342757Z" level=info msg="starting ChirpStack Gateway Bridge"
chirpstack-gateway-bridge-eu868_1 | time="2023-03-22T06:06:48.640419048Z" level=info msg="backend/semtechudp: starting gateway"
chirpstack-gateway-bridge-eu868_1 | time="2023-03-22T06:06:48.642859303Z" level=warning msg="[store] memorystore wiped"
chirpstack-gateway-bridge-eu868_1 | time="2023-03-22T06:06:48.642864283Z" level=info msg="integration/mqtt: connected to mosquitto"
chirpstack-rest-api_1 | Starting ChirpStack REST API server
chirpstack_1 | 2023-03-22T06:06:48.655608Z INFO chirpstack::cmd::root: Starting ChirpStack LoRaWAN Network
chirpstack_1 | 2023-03-22T06:06:48.655646Z INFO chirpstack::region: Setting up regions
chirpstack_1 | 2023-03-22T06:06:48.655673Z INFO setup{common_name=US915 region_id=us915_0}: chirpstack
chirpstack_1 | 2023-03-22T06:06:48.655744Z INFO setup{common_name=EU433 region_id=eu433}: chirpstack
chirpstack_1 | 2023-03-22T06:06:48.655788Z INFO setup{common_name=CN470 region_id=cn470_10}: chirpstack
mosquitto_1 | 1679465208: mosquitto version 2.0.15 starting
mosquitto_1 | 1679465208: Config loaded from /mosquitto/config/mosquitto.conf.
mosquitto_1 | 1679465208: Opening ipv4 listen socket on port 1883.
mosquitto_1 | 1679465208: Opening ipv6 listen socket on port 1883.
postgres_1 | The files belonging to this database system will be owned by user "postgres".

```

`sudo docker-compose up` 在当前控制台运行 使用 `ctrl+c` 停止
`sudo docker-compose up -d` 在后台运行, 使用 `sudo docker-compose stop` 停止

6. 修改服务器配置文件

切换路径到 `chirpstack-docker` 下,

输入命令 `vim configuration/chirpstack/chirpstack.toml` 找到


```
# Enabled regions.
#
# Multiple regions can be enabled simultaneously. Each region must match
# the 'name' parameter of the region configuration in '[[regions]]'.
enabled_regions=[
  "as923",
  "as923_2",
  "as923_3",
  "as923_4",
  "au915_0",
  "cn470_10",
  "cn779",
  "eu433",
  "eu868",
  "in865",
  "ism2400",
  "kr920",
  "ru864",
  "us915_0",
  "us915_1",
]
```

添加需要支持的地区文件.此处可以将需要你所需要的地区添加进去.也可添加已经支持的全部地区(后续不在修改).将会在配置界面提供选项.

下面根据各个地区不同.进行不同的配置: 以下地区代码需要系统配置中支持才行查看已经支持的地区 查看命令 `ls configuration/chirpstack/`

```
min@ubuntu: /chirpstack-docker$ ls configuration/chirpstack/
chirpstack-gateway-bridge.toml
min@ubuntu: /chirpstack-docker$ ls configuration/chirpstack/
chirpstack.toml  region_au915_0.toml  region_au915_5.toml  region_cn470_11.toml  region_cn470_5.toml  region_cn779.toml  region_kr920.toml  region_us915_3.toml
region_as923_2.toml  region_au915_1.toml  region_au915_6.toml  region_cn470_1.toml  region_cn470_6.toml  region_eu433.toml  region_ru864.toml  region_us915_4.toml
region_as923_3.toml  region_au915_2.toml  region_au915_7.toml  region_cn470_2.toml  region_cn470_7.toml  region_eu868.toml  region_us915_0.toml  region_us915_5.toml
region_as923_4.toml  region_au915_3.toml  region_cn470_0.toml  region_cn470_3.toml  region_cn470_8.toml  region_in865.toml  region_us915_1.toml  region_us915_6.toml
region_as923.toml   region_au915_4.toml  region_cn470_10.toml  region_cn470_4.toml  region_cn470_9.toml  region_ism2400.toml  region_us915_2.toml  region_us915_7.toml
min@ubuntu: /chirpstack-docker$
```

打开 Vim `configuration/chirpstack-gateway-bridge/chirpstack-gateway-bridge.toml`

```
# See https://www.chirpstack.io/gateway-bridge/install/config/ for a full
# configuration example and documentation.

[integration.mqtt.auth.generic]
servers=["tcp://mosquitto:1883"]
username=""
password=""

[integration.mqtt]
event_topic_template="eu868/gateway/{{ .GatewayID }}/event/{{ .EventType }}"
state_topic_template="eu868/gateway/{{ .GatewayID }}/state/{{ .StateType }}"
command_topic_template="eu868/gateway/{{ .GatewayID }}/command/#"
```

修改为 **cn470_0**

打开 `vim docker-compose.yml`

```
- redis
environment:
  - MQTT_BROKER_HOST=mosquitto
  - REDIS_HOST=redis
  - POSTGRES_HOST=postgres
ports:
  - 8080:8080

chirpstack-gateway-bridge-eu868:
  image: chirpstack/chirpstack-gateway-bridge:4.470_0
  restart: unless-stopped
  ports:
```

修改为
cn470_0

7. 再次启动

输入指令：sudo docker-compose up，

部署完成后如下图，当启动文件有切换的地区信息则切换成功。

```
min@ubuntu:~/chirpstack-docker$ ^C
min@ubuntu:~/chirpstack-docker$ sudo docker-compose up
[sudo] password for min:
WARNING: Found orphan containers (chirpstack-docker_chirpstack-gateway-bridge-eu868_1) for this project. If you removed
lean it up.
Starting chirpstack-docker_mosquitto_1 ... done
Starting chirpstack-docker_postgres_1 ... done
Starting chirpstack-docker_redis_1 ... done
Creating chirpstack-docker_chirpstack-gateway-bridge-cn470_0_1 ... done
Starting chirpstack-docker_chirpstack_1 ... done
Starting chirpstack-docker_chirpstack-rest-api_1 ... done
Attaching to chirpstack-docker_postgres_1, chirpstack-docker_mosquitto_1, chirpstack-docker_redis_1, chirpstack-docker_
1
chirpstack-gateway-bridge-cn470_0_1 | time="2023-03-22T06:33:43.494565897Z" level=info msg="starting ChirpStack Gateway
chirpstack-gateway-bridge-cn470_0_1 | time="2023-03-22T06:33:43.494638547Z" level=info msg="backend/semtechudp: startin
```

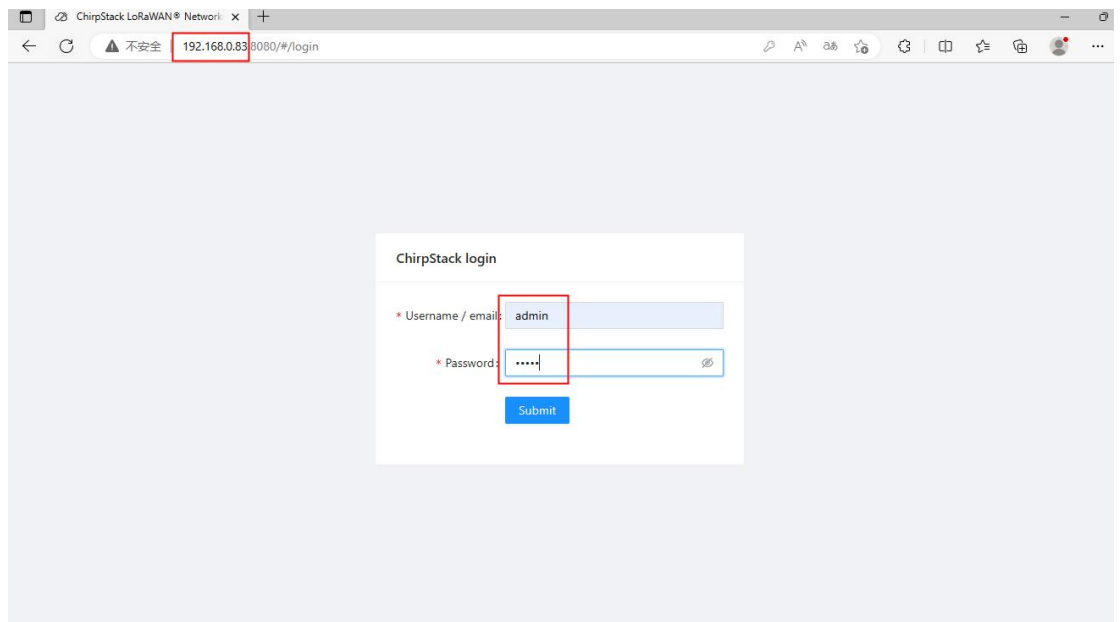
在浏览器输入 localhost:8080 即可进入服务器登录界面

二、 网页配置网关和节点信息，并通讯。

首先需要保证所使用的的电脑和网关能顺利连接服务器，比如本地服务器时，确保配置电脑、E890 网关和服务器处在同一网段，又如果是公网服务器，确保配置电脑和 E890 网关能连接外网。

1. 网页配置

打开浏览器输入：服务器 IP 地址:8080。默认账号和密码均为 admin，请第一次使用时注意修改密码。



2. 生成服务器、网关和节点信息

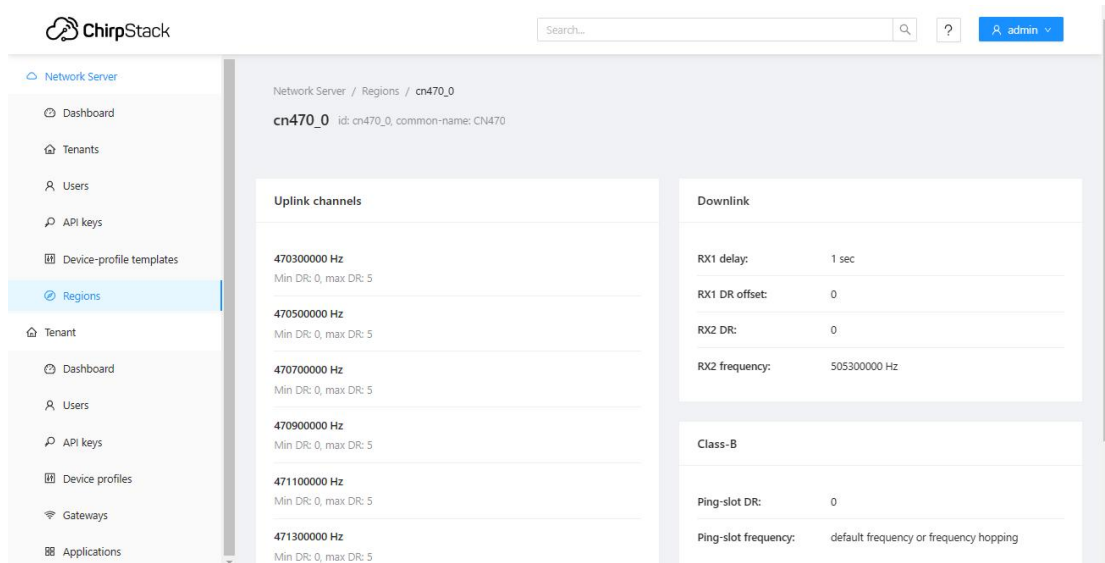
A. 生成服务器信息

在搭建服务器时，已经切换到服务器的地区文件为 CN470-0 频段，所以这个时候服务器运行在 CN470-0 频段

点击 Regions，可以看到目前服务器支持哪些频段

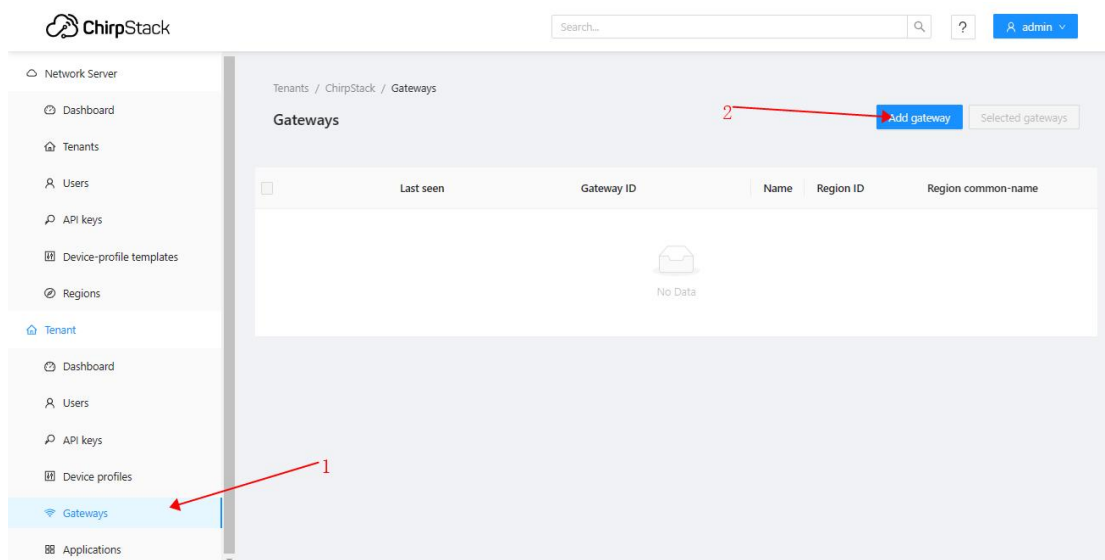
ID	Region	Description
as923	AS923	as923
as923_2	AS923_2	as923_2
as923_3	AS923_3	as923_3
as923_4	AS923_4	as923_4
au915_0	AU915	au915_0
au915_1	AU915	au915_1
au915_2	AU915	au915_2
au915_3	AU915	au915_3
au915_4	AU915	au915_4

点击 cn470-0，可以看到此频段的详情

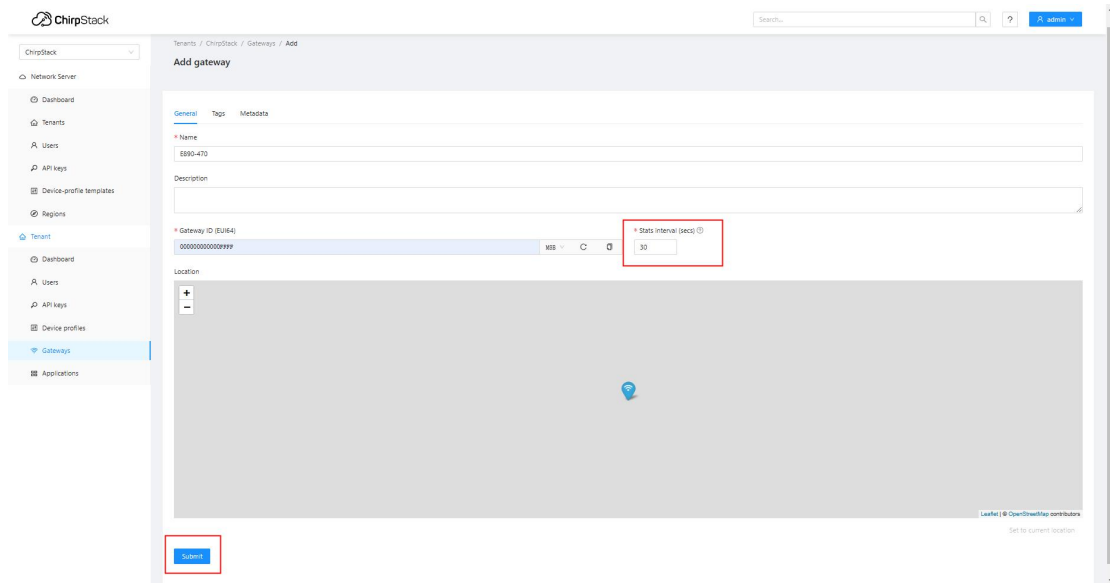


B. 生产网关和节点信息

点击左侧栏中 Gateways，然后点击右上角 Add gateway 新建一个网关种类。



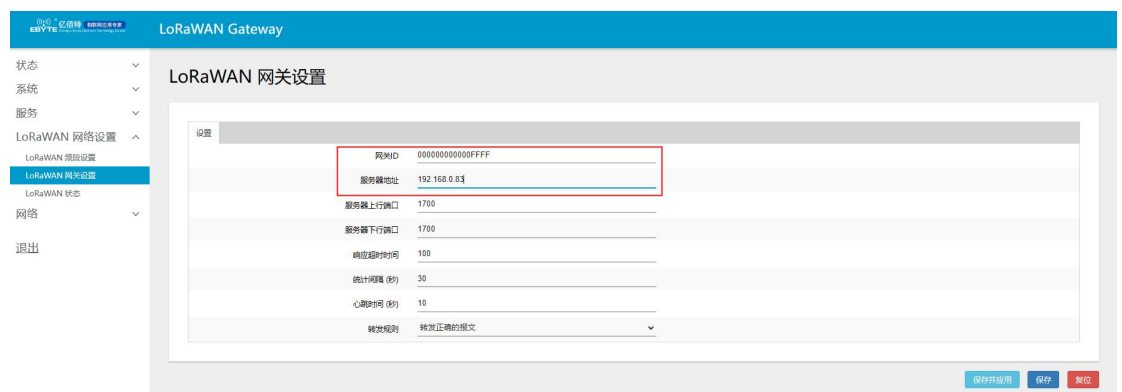
网关类型为 E890，填写网关 ID 也可以自动生成 ID，同个服务器不能使用相同的网关 ID，E890 网关默认使用 00 00 00 00 00 00 FF FF。红框为网关的状态信息上报周期，E890 和网关均使用默认的 30 秒。点击 submit 创建网关。



The image shows the 'Add gateway' form in the ChirpStack web interface. The form is divided into three tabs: 'General', 'Tags', and 'Metadata'. The 'General' tab is active. It contains the following fields:

- Name:** E890-470
- Description:** (empty)
- Gateway ID (EUI64):** 000000000000FFFF. To its right, there is a dropdown menu with 'MB', 'C', and '0' options, and a 'Stats interval (secs)' field set to '30'.
- Location:** A map area with a blue location pin and a 'Set to current location' link.
- Submit:** A blue button at the bottom left of the form.

打开 WiFi，连接网关的 WiFi，名称为 EBT-E890-XXXX。浏览器输入 192.168.10.1 进入配置页面。密码为 root。网关 ID 默认 0000000000FFFF，并修改 IP 地址为服务器的 IP 地址。

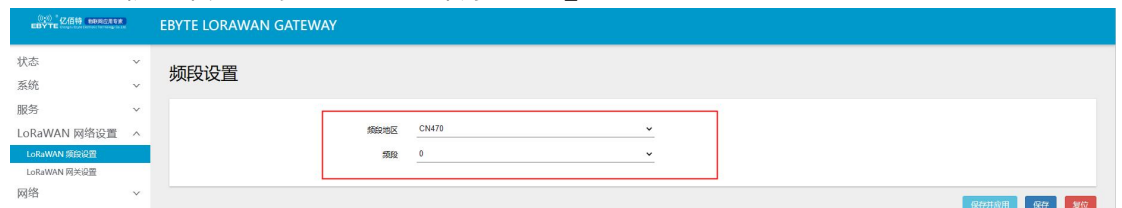


The image shows the 'LoRaWAN Gateway' configuration page in the EBYTE web interface. The page has a sidebar on the left with the following menu items: '状态', '系统', '服务', 'LoRaWAN 网络设置', 'LoRaWAN 网络设置', 'LoRaWAN 状态', '网络', and '退出'. The 'LoRaWAN 网络设置' menu item is selected. The main content area is titled 'LoRaWAN 网关设置' and contains a '设置' (Settings) section with the following fields:

- 网关ID:** 000000000000FFFF
- 服务器地址:** 192.168.0.83
- 服务器上端口:** 1700
- 服务器下行端口:** 1700
- 响应超时时间:** 100
- 统计周期 (秒):** 30
- 心跳时间 (秒):** 10
- 转发规则:** 转发正确的报文

At the bottom right of the settings section, there are three buttons: '保存并应用' (Save and Apply), '保存' (Save), and '复位' (Reset).

网关频段根据服务器的配置文件选择为 cn470_0

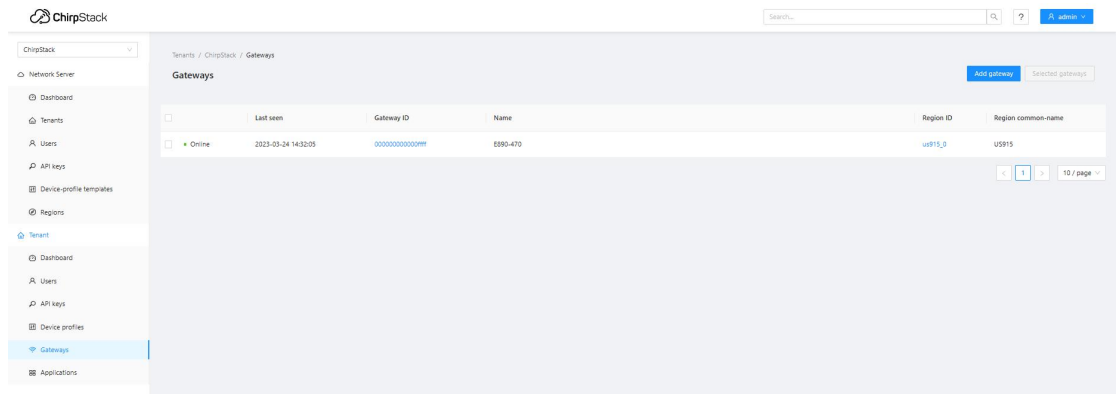


The image shows the 'LoRaWAN Gateway' frequency settings page in the EBYTE web interface. The page has a sidebar on the left with the following menu items: '状态', '系统', '服务', 'LoRaWAN 网络设置', 'LoRaWAN 网络设置', 'LoRaWAN 状态', '网络', and '退出'. The 'LoRaWAN 网络设置' menu item is selected. The main content area is titled '频段设置' (Frequency Settings) and contains a '设置' (Settings) section with the following fields:

- 频段地区:** CN470
- 频段:** 0

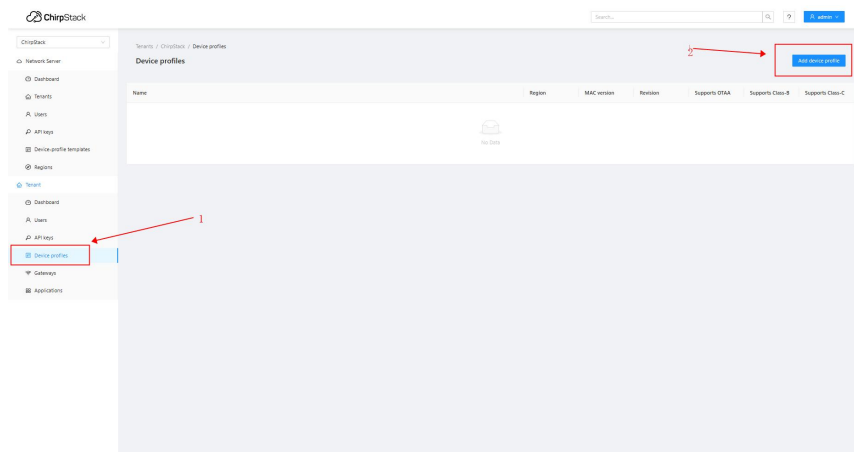
At the bottom right of the settings section, there are three buttons: '保存并应用' (Save and Apply), '保存' (Save), and '复位' (Reset).

点击保存并应用，即可看到网关 LINK 灯亮起，即表示网关已连上服务器。



C. 生成节点并通信测试

随后生成节点种类和节点信息。点击左侧框中 Device-profile, 点击右上角“Add device profiles”新建一个节点种类。



下图是 E78-470LN22S 使用的参数信息。

General Join (OTAA / ABF) Class-B Class-C Codec Tags Measurements Select device-profile template

Name E78-470

Description

Region CN470 Region configuration cn470_0

MAC version LoRaWAN 1.0.2 Regional parameters revision 8

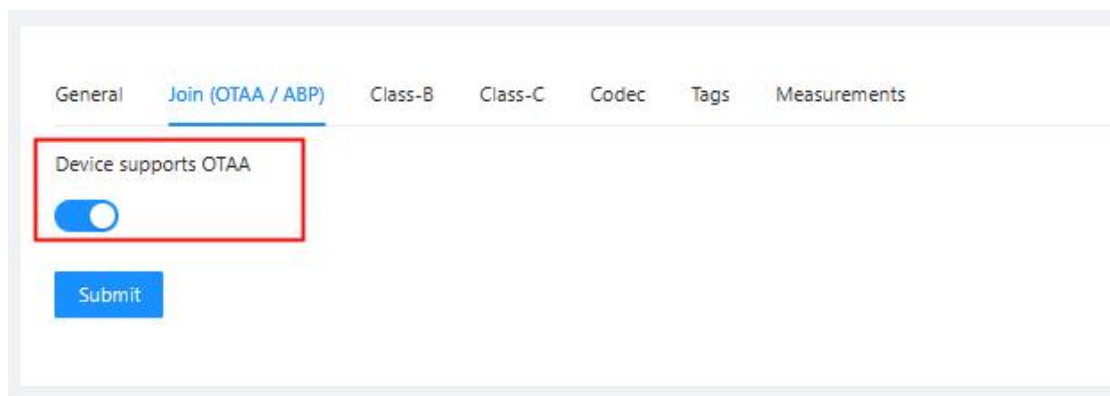
ADR algorithm Default ADR algorithm (LoRa only)

Flush queue on activate ☒ Expected uplink interval (secs) 3600 Device status request frequency (req/day) 1

Submit

这里填写的参数应与节点本身的频段和 LoRaWAN 版本一致，这里 E78-470LN22S 的频段是

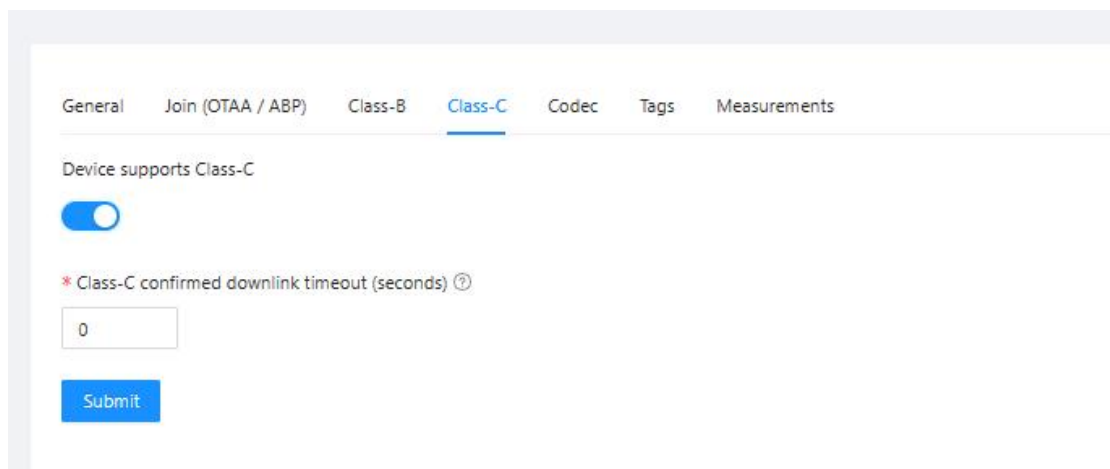
470_0, LoRaWAN 版本选择 1.0.2-B



General Join (OTAA / ABP) Class-B Class-C Codec Tags Measurements

Device supports OTAA

Submit



General Join (OTAA / ABP) Class-B Class-C Codec Tags Measurements

Device supports Class-C

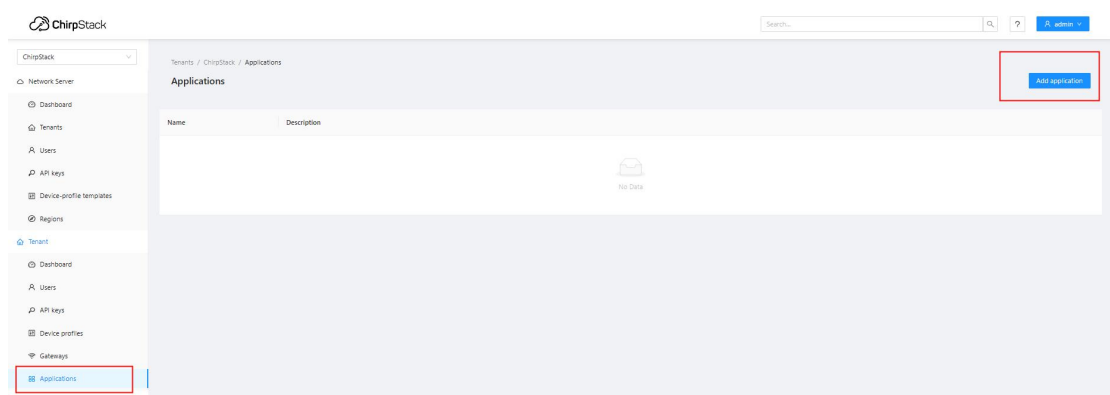
* Class-C confirmed downlink timeout (seconds) ?

0

Submit

点击红色框, 开启 OTAA 模式, 并开启 CLASS C , 最后点击 Submit 创建 device Profiles。 OTAA 和 ABP 模式具体区别请参看 LoRaWAN 规范中的描述。简而言之, OTAA 比 ABP 模式更加灵活, 易于部署。

点击左侧栏中 Applications, 点击 Add application 新建一个应用, 命名为 E78-470。



ChirpStack

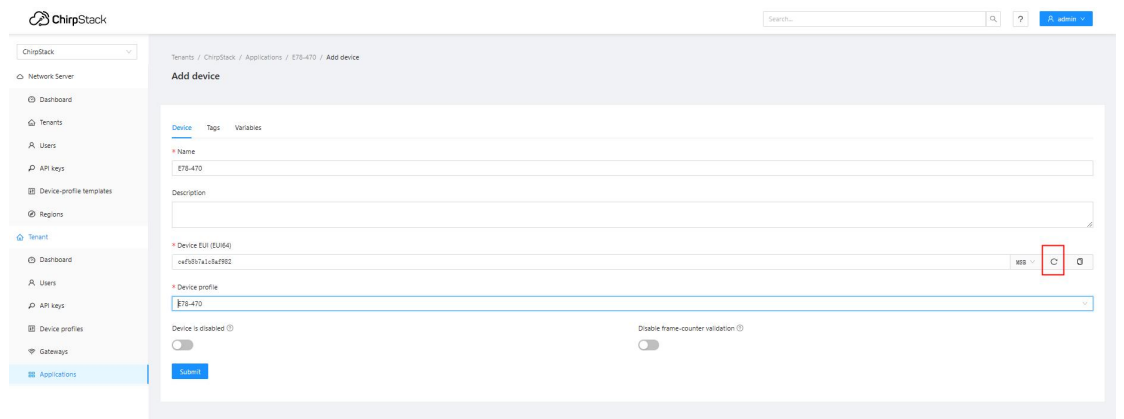
Search

Applications

Add application

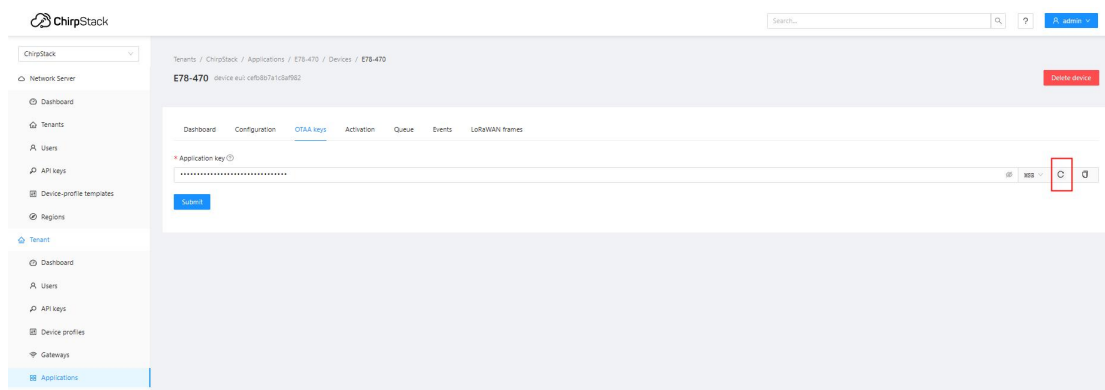
Name	Description
No Data	

退回上一级界面点击 e78, 点击右上角 Add device 生成一个节点。下图中红框为 DevEUI, deviceprofiles 选择刚才创建的 E78-470,之后设置节点信息需要这个参数。



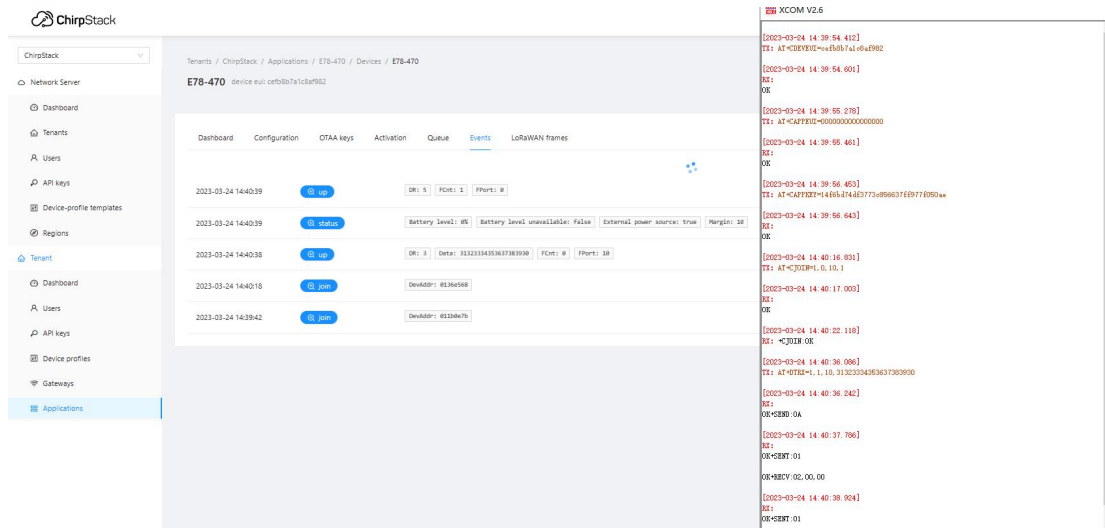
点击 submit 建立节点，随后会弹出添加 APPKEY 的界面

随后点击图中红框图标生成 APPKEY，



最后点击 submit 建立节点成功。

按照 E78 手册中的 AT 指令设置 E78 的入网信息，并申请入网，如下图。



此时成功搭建 E890、E78 和 Chirpstack 服务器组成的 LoRaWAN 网络。

三、结语

使用 chirpstack 和 E890、E78 能快速搭建 LoRaWAN 网络，对于一些需要快速成型

的项目有很大的优势,同时利用 Chirpstack 的拓展功能,能快速与其他服务器形成配合。