ED-GWL2110

用户手册

by EDA Technology Co., Ltd

built: 2025-08-01

1 产品概述

ED-GWL2110是一款基于Raspberry Pi CM4的室外网关，整机采用全铝合金外盒密封，拥有良好的防雨、防潮、防虫、防雷电性能，支持多种不同频段的LoRa模块(需搭配不同频段的外置天线)。支持选配4G和Wi-Fi/BT功能，保证在室外可以正常上传下载数据。
ED-GWL2110板载GNSS模块，可方便实现定位需求；内置硬件看门狗模块、RTC和加密芯片，提升了系统的安全和可靠性。

1.1 目标应用

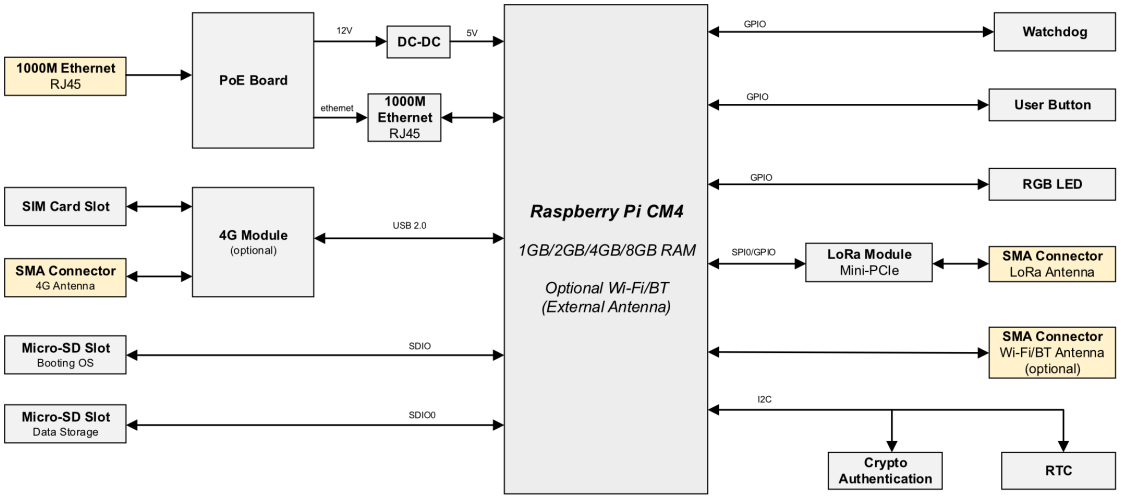
- 智慧城市
- 智慧交通
- 智慧农业

1.2 规格参数

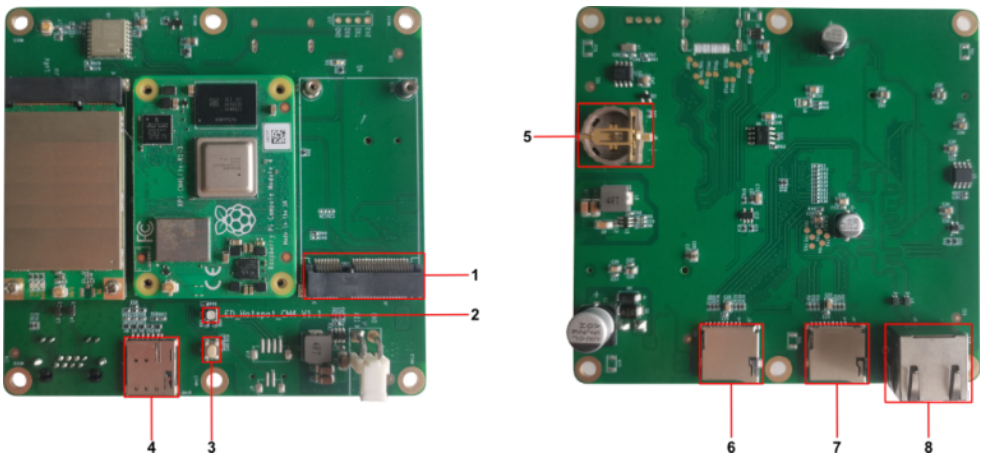
功能	参数
CPU	Broadcom BCM2711 4核Cortex A72 1.5GHz (ARM v8) 64-bit SoC
RAM	1GB/2GB/4GB/8GB可选
SD卡	32GB/64GB可选，从SD卡启动系统
1000M以太网口	1 x 1000M以太网口，RJ45端子，支持PoE，用于接入以太网和给设备供电
WiFi/BT（选配）	• 2.4GHz和5GHz双频Wi-Fi, 兼容IEEE 802.11 b/g/n/ac • 蓝牙5.0，支持BLE
4G（选配）	支持多种4G LTE模块
LoRa	兼容LoRa WAN协议，支持3种频段 • 868MHz(EU868) • 915MHz(US915) • 470MHz(CN470)
GNSS	内置GNSS功能，支持多卫星系统；兼容LoRa WAN协议，支持3种频段 • GPS L1 C/A：1575.42 ±1.023 MHz • BeiDou B1I：1561.098 ±2.046 MHz • GLONASS L1：1597.78~1605.66 MHz
内部IO	• 1 x Serial(TTL)，可用于系统默认控制台

功能	参数
	• 1 x 用户自定义按键 • 1 x RGB三色LED • 1 x RTC电池底座，可安装CR1220纽扣电池 • 1 x Nano SIM卡槽 • 2 x Micro SD卡槽
扩展功能	• 支持硬件看门狗功能 • 内置加密芯片 • 内置RTC
输入电源	PoE供电，支持802.3af标准
尺寸	194.2mm(长) x 194.2mm(宽) x 65.3 mm (高)
外壳	铸铝防水外壳，IP65防水等级
工作温度范围	-25°C~60°C

1.3 系统框图



1.4 主板接口

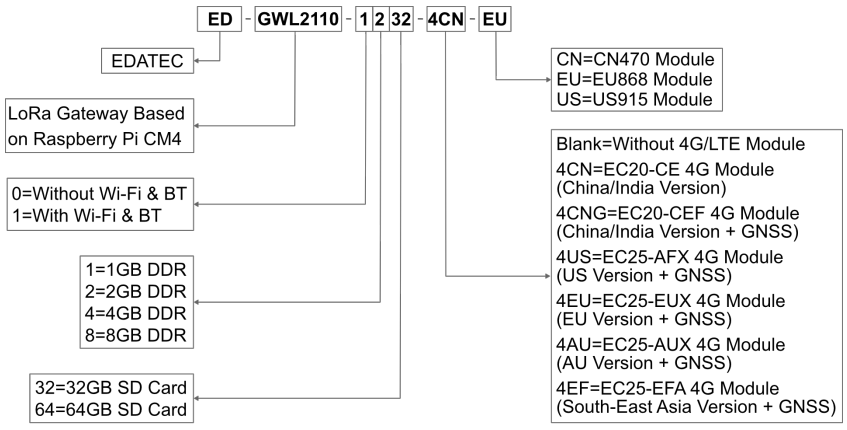


编号	功能说明
1	4G mini-PCIe接口
2	RGB LED
3	用户按键
4	Nano SIM卡槽
5	RTC电池底座
6	Micro SD卡槽（存储用户数据）
7	Micro SD卡槽（系统启动）
8	千兆网口

1.5 包装清单

- 1 x ED-GWL2110主机
- 1 x LoRa天线
- [选配Wi-Fi/BT版本] 1 x 2.4GHz/5GHz Wi-Fi/BT天线
- [选配4G版本] 1 x 4G/LTE天线

1.6 订购编码



Example

P/N: **ED-GWL2110-1232-4CN-EU**
Configuration: An outdoor LoRa Gateway Based on Raspberry Pi CM4, with Wi-Fi & Bluetooth, 2GB DDR, 32GB SD Card, 4G(EC20-CE Module) and EU868 LoRa Module.

2 快速启动

介绍ED-GWL2110的启动及部分开机设置。

2.1 设备清单

- 1 x ED-GWL2110系列主机
- 1 x Wi-Fi/BT外置天线（选配）
- 1 x LoRa外置天线
- 1 x 4G外置天线（选配）
- 1 x 网线

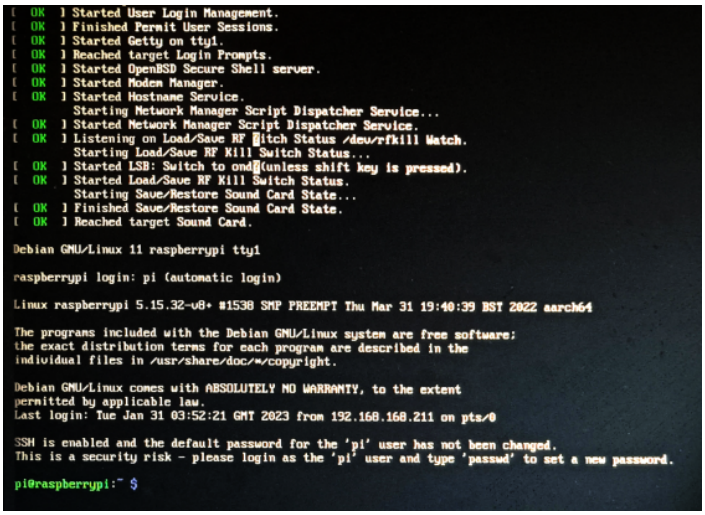
2.2 硬件连接

1. 分别安装Wi-Fi、LoRa或4G外置天线。
2. 连接网线：
   - 一端连接设备的以太网口
   - 另一端连接可上网的带PoE功能路由器或交换机

2.3 首次启动

ED-GWL2110无电源开关，接入PoE供电后，系统将会开始启动。

如果使用设备出厂时默认的Raspberry Pi OS (Lite)操作系统，系统启动后会使用用户名pi自动登入，默认密码为raspberry。



> 提示
>
> 默认的操作系统已使能SSH功能。

2.4 查找设备IP

2.4.1 登录路由器查询

当设备正常启动后，可以登录路由器查看当前设备IP。

前提条件：

- 设备已通过路由器接入网络。
- 已获取所在网络的路由器的IP和网络密码，IP地址如192.168.X.X。

操作步骤：

1. 打开浏览器，在地址栏中输入设备所在网络的路由器IP：192.168.X.X，按Enter键进入路由器登录界面。
2. 按照界面提示，输入网络密码，进入路由器管理界面。
3. 在管理界面的终端设备中找到设备的IP地址。

2.4.2 使用nmap工具扫描获取

当设备正常启动后，可以使用nmap工具扫描当前网络下的IP来获取设备IP信息，nmap支持Linux、macOS、Windows等多个平台。

前提条件：

- 设备已通过路由器接入网络。
- 已获取所在网络的路由器的IP网段和掩码，例如192.168.X.X/24，其中24为子网掩码。

操作步骤：

例如使用nmap扫描192.168.3.0~255的网段，则可以使用如下步骤：

1. 打开nmap工具，扫描192.168.X.X/24网段中的主机。

> 注意
>
> nmap工具在不同的操作系统中的操作不同，请根据实际界面或命令的提示来操作即可。

2. 根据扫描出的结果，获取设备IP。

2.5 通过SSH远程登录设备

设备正常启动后，可以选择通过SSH远程连接到设备对其进行配置或调试。远程登录的工具由用户自己选择，下文以通过MobaXterm登录为例进行说明。

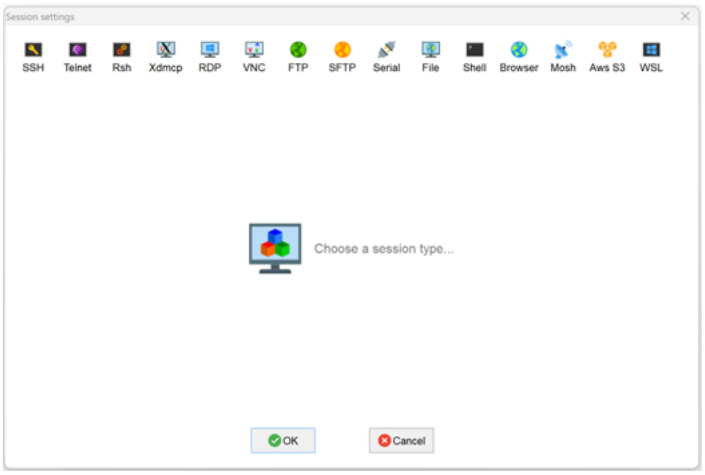
前提条件：

- 已在PC上安装MobaXterm工具。

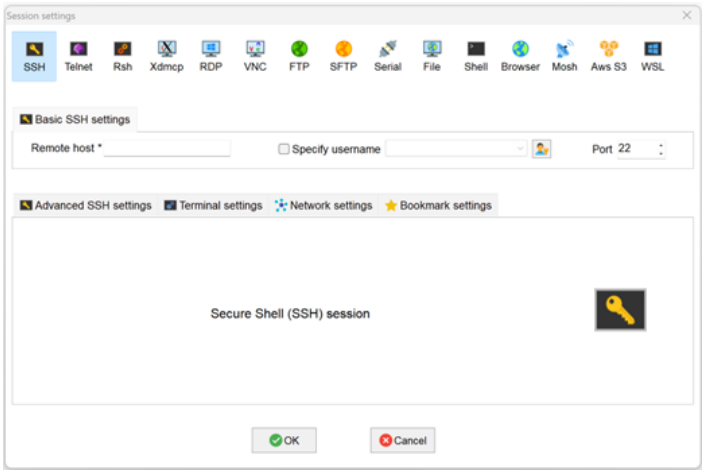
- 设备已通过路由器接入网络。
- 已获取设备的IP地址。

操作步骤

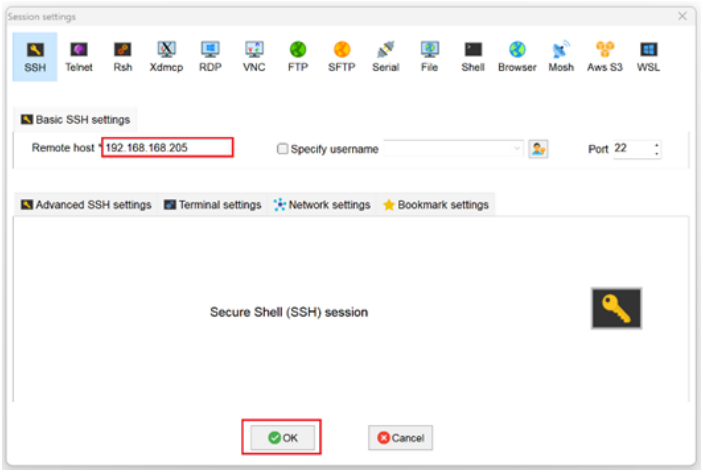
1. 打开MobaXterm，单击  Session，打开创建连接的窗口，如下图所示。



2. 单击左上角的  SSH，打开SSH连接界面。

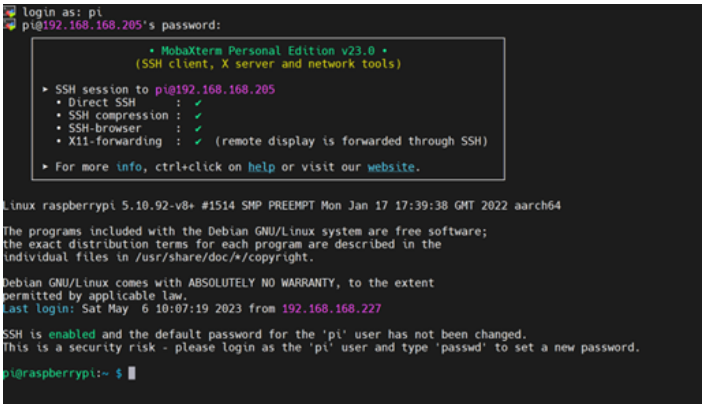


3. 输入已获取的设备IP地址后，单击“OK”。



4. 在弹出的提示框中单击“Accept”，进入系统登录界面。
5. 根据提示输入用户名和密码，完成登录后进入系统。

> 提示
>
> 默认用户名：pi，默认密码：raspberry。



3 软件操作指引

3.1 按键

ED-GWL2110包含1个用户自定义按键，在设备内部，连接到CPU的GPIO23管脚，默认状态下为高电平，当按键按下时，该管脚为低电平。

可使用`raspi-gpio`命令进行查询按键对应GPIO的状态。

- 按键未按下时，查询GPIO23管脚

```sh
raspi-gpio get 23
GPIO 23: level=1 fsel=0 func=INPUT
```

level为1表示GPIO23管脚为高电平。

- 按键按下时，查询GPIO23管脚

```sh
raspi-gpio get 23
GPIO 23: level=0 fsel=0 func=INPUT
```

level为0表示GPIO23管脚为低电平。

3.2 LED指示灯

ED-GWL2110包含一个RGB三色LED指示灯，相连的GPIO管脚对应如下：

RGB LED管脚	对应GPIO
蓝色	GPIO11
绿色	GPIO12
红色	GPIO13

GPIO输出为低时，对应LED有效。

支持使用`raspi-gpio`命令进行查询GPIO的状态。

- 配置参数为op，表示设置为输出；
- 配置参数为dl，表示设备管脚为低电平；
- 配置参数为dh，表示设置管脚为高电平。

LED显示为蓝色

```sh
sudo raspi-gpio set 11 op dl
sudo raspi-gpio set 12 op dh
sudo raspi-gpio set 13 op dh
```

LED显示为绿色

```sh
sudo raspi-gpio set 11 op dh
sudo raspi-gpio set 12 op dl
sudo raspi-gpio set 13 op dh
```

LED显示为红色

```sh
sudo raspi-gpio set 11 op dh
sudo raspi-gpio set 12 op dh
sudo raspi-gpio set 13 op dl
```

LED显示为黄色

```sh
sudo raspi-gpio set 11 op dh
sudo raspi-gpio set 12 op dl
sudo raspi-gpio set 13 op dl
```

3.3 配置以太网IP

出厂默认为自动获取IP地址，如果需要重新配置IP，可通过NetworkManager工具来配置。

> 提示
>
> 出厂默认的Desktop和Lite操作系统已使能NetworkManager，可直接使用NetworkManager进行配置。

3.3.1 Raspberry Pi OS(Desktop)

在Desktop版本的操作系统中，建议使用图形化的NetworkManager工具来配置IP。

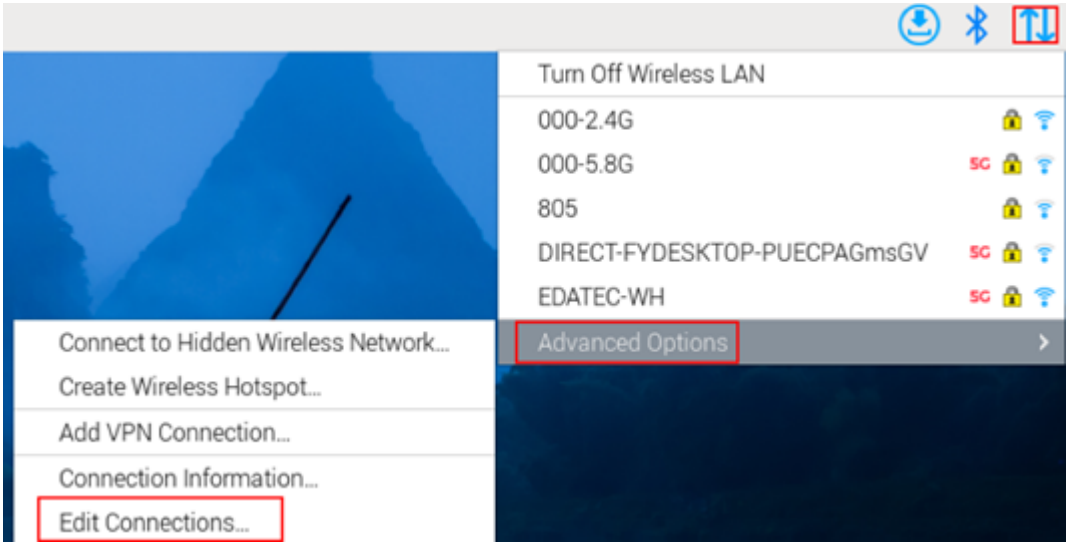
> 提示
>
> Desktop版本的操作系统已默认安装NetworkManager图形化工具

前提条件：

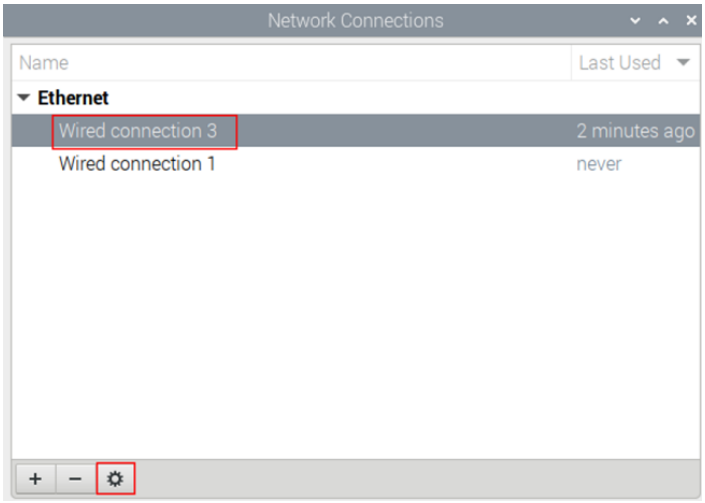
已使能Wi-Fi功能。

操作步骤：

1. 左键单击桌面右上角的  图标，在菜单中选择“Advanced Options→Edit Connections”。

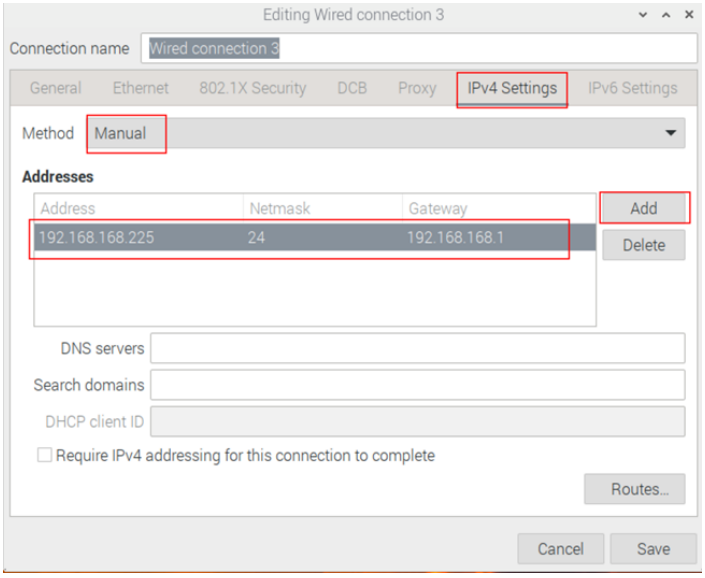


2. 在弹出的Network Connections窗格中选中要修改的连接名称，再单击下方的设置按钮。

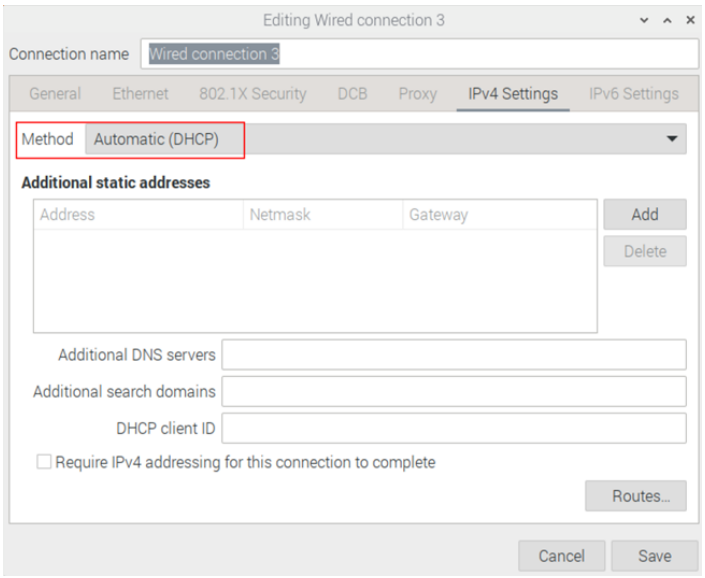


3. 在弹出的Editing Wired connection中选择IPv4 Settings配置页，按需设置IP地址。

- 如果要将IP设置为静态IP，则设置Method为Manual，在Addresses中增加一个条目并输入对应的IP地址信息。



- 如果要将IP设置为自动获取，则仅需要将Method设置为Automatic(DHCP)。



4. 单击save返回至Network Connections，关闭页面。
5. 在命令窗格中执行 `sudo reboot` 命令，重启设备。

3.3.2 Raspberry Pi OS(Lite)

在Lite版本的操作系统中，支持通过命令行来配置IP。

前提条件：

已使能NetworkManager。

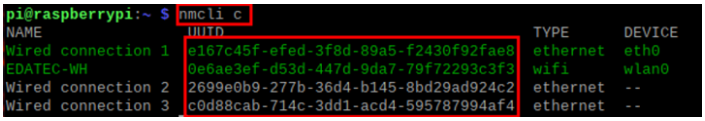
操作步骤：

设置静态IP地址

1. 获取分配的IP地址、子网掩码和网关地址，例如IP地址为192.168.1.101/24，网关IP为192.168.1.1。

2. 获取待修改的连接的名称，例如e167c45f-efed-3f8d-89a5-f2430f92fae8，可在命令窗格中执行如下命令查询连接名称。

```sh
nmcli c
```



3. 在命令窗格执行如下命令，将对应连接的IP地址设置为已获取的IP地址。

```sh
sudo nmcli connection modify e167c45f-efed-3f8d-89a5-f2430f92fae8 ipv4.addresses 192.168.1.101
```

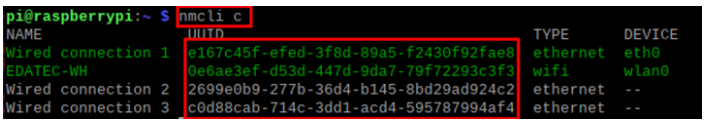
4. 执行如下命令，将网关IP设置为已获取的网关IP。

```sh
sudo nmcli connection modify e167c45f-efed-3f8d-89a5-f2430f92fae8 ipv4.gateway 192.168.1.1
```

设置自动获取IP

1. 获取待修改的连接的名称，例如e167c45f-efed-3f8d-89a5-f2430f92fae8，可在命令窗格中执行如下命令查询连接名称。

```sh
nmcli c
```



2. 执行如下命令，将对应连接的IP地址的方式设置自动获取IP。

```sh
sudo nmcli connection modify e167c45f-efed-3f8d-89a5-f2430f92fae8 ipv4.method auto
```

3.4 配置Wi-Fi (可选)

配置Wi-Fi

3.5 配置蓝牙 (可选)

配置蓝牙

3.6 配置 4G (可选)

配置4G

3.7 配置存储设备（SD卡）

配置存储设备

3.8 配置 RTC

配置RTC

3.9 GNSS

ED-GWL2110网关集成L76K GPS模块，其与CPU的UART0串口相连。模块通过NMEA 0183通用协议输出语句上报GNSS信息。

3.9.1 引脚配置

- L76K GPS模块的WakeUp信号与GPIO4相连，拉低该管脚模块将进入待机模式，拉高或悬空该管脚模块将返回连续模式。
- Reset信号与GPIO5相连，拉低该管脚并持续100ms将复位模块。
- SET信号与GPIO6相连，用于配置卫星组合。
  - 当该管脚为悬空或高电平时，卫星组合为GPS和北斗；
  - 当管脚为低电平时，卫星组合为GPS和GLONASS。

编号	Signal	CM4 Pinout
1	GPS_WakeUp	GPIO4
2	GPS_Reset	GPIO5
3	GPS_Set	GPIO6

3.9.2 修改 `config.txt` 使能串口

1. 执行如下命令，打开 `config.txt` 文件。

```sh
sudo nano /boot/config.txt
```

2. 在文件的末尾在最后面添加“enable_uart=1”。
3. 输入 `ctrl+o` 保存配置文件，再按 `Enter` ，最后输入 `ctrl+x` 退出配置文件。

3.9.3 查看GNSS信息

```sh
sudo cat /dev/ttyS0
```

显示GPS数据如下：

```text
$BDGSV,3,1,11,04,29,117,20,10,,,19,16,75,160,,24,51,328,,0*4C
$BDGSV,3,2,11,25,,,27,26,,,21,34,12,198,,35,45,063,,0*76
$BDGSV,3,3,11,39,62,159,17,41,,,25,59,44,137,,0*7A
$GNRMC,053557.000,A,3027.47401,N,11424.34027,E,1.17,186.64,070223,,,A,V*05
$GNVTG,186.64,T,,M,1.17,N,2.17,K,A*2D
$GNZDA,053557.000,07,02,2023,00,00*4F
$GPTXT,01,01,01,ANTENNA OPEN*25
$GNGGA,053558.000,3027.47438,N,11424.34119,E,1,07,1.5,75.0,M,-14.1,M,,*52
$GNGLL,3027.47438,N,11424.34119,E,053558.000,A,A*4F
$GNGSA,A,3,07,08,16,31,195,,,,,,,,2.1,1.5,1.5,1*05
$GNGSA,A,3,04,39,,,,,,,,,,,,2.1,1.5,1.5,4*39
$GPGSV,3,1,12,04,54,241,16,07,19,314,15,08,63,208,15,09,38,291,,0*67
$GPGSV,3,2,12,16,51,029,17,18,07,046,,21,08,175,,26,24,063,,0*6A
$GPGSV,3,3,12,27,77,065,,31,09,122,22,194,61,058,,195,46,125,21,0*66
```

NMEA 0183通用语句说明如下：

- BDGSV：可视北斗卫星信息
- $GNRMC：推荐的GNSS数据
- $GNVTG：相对地面航向和速度信息
- $GNZDA：时间和日期，UTC格式
- $GPTXT：文本传送
- $GNGGA：多星联合定位数据
- $GNGLL：地理位置纬度和经度
- $GNGSA：表示GNSS精度因子和有效卫星
- $GPGSV：可视的GNSS卫星

3.9.4 使用u-center工具查看定位信息

3.9.4.1 安装串口转网络工具ser2net

```sh
sudo apt-get update
sudo apt-get install ser2net
```

启用ser2net服务

ser2net配置文件为 `/etc/ser2net.yaml` ，默认已配置 `/dev/ttyS0` ，波特率为9600，无校验，对应的TCP端口为2000。

```sh
connection: &con0096
    accepter: tcp,2000
    enable: on
    options:
      banner: *banner
      kickolduser: true
      telnet-brk-on-sync: true
    connector: serialdev,
              /dev/ttyS0,
              9600n81,local
```

3.9.4.2 检查ser2net端口转发服务

执行如下命令，查询ser2net是否已启动2000端口转发。

```sh
sudo netstat -ltnp | grep 2000
```

- 如果已启动端口转发，将显示如下信息。

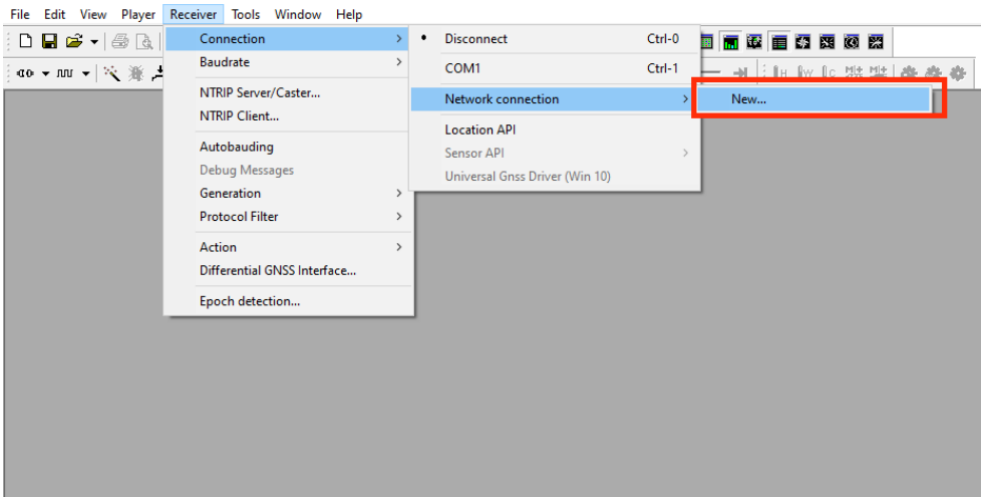
```text
tcp6    0    0 :::2000         :::*          LISTEN    720/ser2net
```

- 如果无信息显示，则重启ser2net服务。

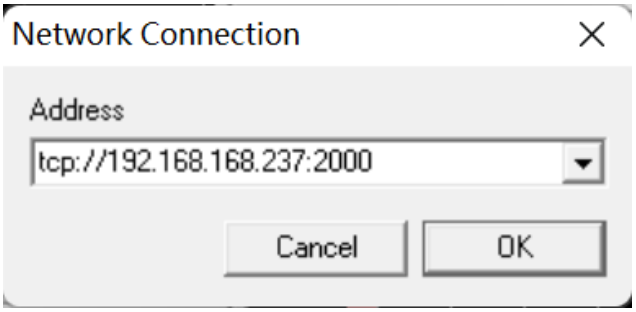
```sh
sudo systemctl restart ser2net
```

配置定位

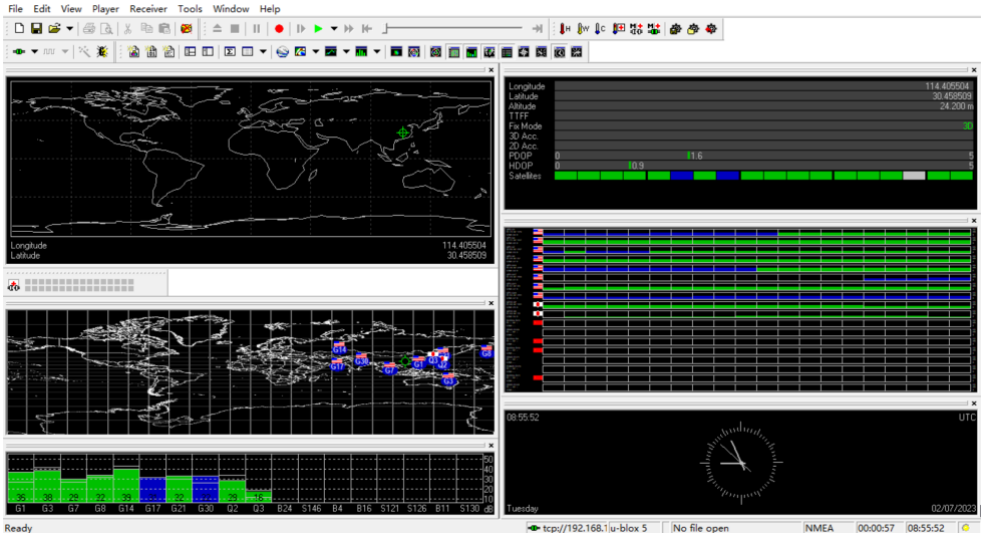
1. 下载u-center (https://www.u-blox.com/en/product/u-center) 工具并安装，如果提示缺少MSVCR120.dll文件，请安装vcredist_x86.exe (https://www.microsoft.com/en-us/download/details.aspx?id=40784) 。
2. 打开u-center，选择“Receiver”→“Connection”→“Network connection”→“New…”。



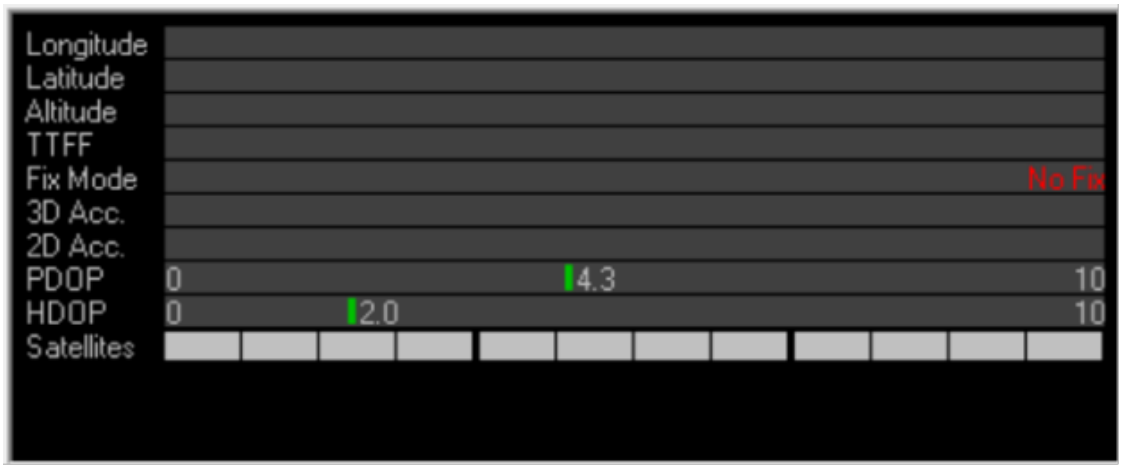
3. 输入设备IP和端口号2000。



4. 配置完成后会显示GPS定位信息。



- 如果Fix Mode显示为No Fix，表示未能成功定位，一般是由于天线在室内的原因造成，请将模块或天线置于户外进行试验。



> 注意
>
> 模块首次定位，在户外没有大型建筑遮挡的情况下，大概需要30秒才能定位成功，如果天气条件恶劣，可能需要更长的定位时间或无法定位。

3.10 LoRaWAN

ED-GWL2110支持LoRaWAN和开源服务平台ChirpStack，下文介绍具体的安装和配置。

> 提示
>
> 仅以US915 LoRa网关模块为例进行介绍。

3.10.1 Chirpstack客户端

设备在出厂时已安装firmware包、LoRa模块包以及chirpstack软件包，可通过 `dpkg -l | grep ed-` 命令检查对应的LoRa模块包。

如用户需要重新安装操作系统，请参考安装操作系统章节。

3.10.2 修改配置文件

3.10.2.1 修改json配置文件

1. 执行如下命令，启动 `ed-lora.service` 。

```sh
sudo systemctl enable --now ed-lora.service
```

2. 执行如下命令，获取json配置文件的路径。

```sh
systemctl status ed-lora.service
```

```
pi@GWL[illegible]:~ $ systemctl status ed-lora.service
● ed-lora.service - EDATec LoRa Packet Forwarder Service
     Loaded: loaded (/lib/systemd/system/ed-lora.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-03-04 11:12:18 CST; 5s ago
   Main PID: 2447 (start.sh)
      Tasks: 5 (limit: 8731)
        CPU: 174ms
     CGroup: /system.slice/ed-lora.service
             ├─2447 /bin/bash /opt/ed-lora/start.sh
             └─2452 /opt/ed-lora/lora_pkt_fwd -c /opt/ed-lora/conf/global_conf.json.US915

Mar 04 11:12:18 GWL2010 systemd[1]: Started ed-lora.service - EDATec LoRa Packet Forwarder Service.
Mar 04 11:12:18 GWL2010 start.sh[2447]: 2025-03-04 11:12:18 [info] pktfwd: started with conf file: /opt/ed-lora/conf/global_conf.json.US915
Mar 04 11:12:18 GWL2010 start.sh[2452]: ERROR: failed to configure uart port
Mar 04 11:12:19 GWL2010 start.sh[2454]: CoreCell reset through GPIO18...
```

3. 执行如下命令，打开json文件。

```sh
sudo nano /opt/ed-lora/conf/global_conf.json.US915
```

4. 在文件底部删除 `"gps_i2c_path": "/dev/i2c-1"` 。

```
            "bandwidth": 250000,
            "datarate": 100000
        }
    },
    "gateway_conf": {
        "gps_i2c_path": "/dev/i2c-1",
        "gateway_ID": "AA555A0000000000",
        /* change with default server address/ports */
        "server_address": "localhost",
        "serv_port_up": 1680,
        "serv_port_down": 1680,
        /* adjust the following parameters for your network */
        "keepalive_interval": 10,
        "stat_interval": 30,
        "push_timeout_ms": 100,
        /* forward only valid packets */
        "forward_crc_valid": true,
        "forward_crc_error": false,
        "forward_crc_disabled": false
    }
}
```

3.10.2.2 修改chirpstack配置文件

1. 执行如下命令，打开配置文件。

```sh
sudo nano /etc/chirpstack-gateway-bridge/chirpstack-gateway-bridge.toml
```

2. 修改如下部分的内容:
   - 修改 `udp_bind = "0.0.0.0:1700"` 为 `udp_bind = "0.0.0.0:1680"`
   - 在gateway前添加网关型号：
     - 修改 `event_topic_template="gateway/{{ .GatewayID }}/event/{{ .EventType }}"` 为 `event_topic_template="us915_0/gateway/{{ .GatewayID }}/event/{{ .EventType }}"` 。

- 修改 `command_topic_template="gateway/{{ .GatewayID }}/command/#"` 为 `command_topic_template="us915_0/gateway/{{ .GatewayID }}/command/#"` 。

> 提示
>
> 如果您使用EU868或CN470模块，请将前缀 `us915_0` 修改为 `eu868_0` 或 `cn470_10` 。

3. 执行如下命令，重启对应的服务。

```sh
sudo systemctl daemon-reload
sudo systemctl restart chirpstack-gateway-bridge.service ed-lora.service
```

3.10.3 Chirpstack服务端

Chirpstack服务端可以部署在当前设备上，也可以部署在同一局域网内的设备上。本文档以在当前设备上部署为例。（如部署在局域网内的其它设备上，请修改3.10.2.2 修改chirpstack配置文件中 `udp_bind` 的 `0.0.0.0` 为该设备的IP）

1. 部署Chirpstack服务端前，请执行如下命令安装docker-compose。

```sh
sudo apt install docker-compose -y
```

2. 采用docker容器的方式部署ChirpStack服务端。

```sh
git clone https://github.com/chirpstack/chirpstack-docker.git
```

```
pi@GWL[redacted]:~ $ git clone https://github.com/chirpstack/chirpstack-docker.git
Cloning into 'chirpstack-docker'...
remote: Enumerating objects: 572, done.
remote: Counting objects: 100% (413/413), done.
remote: Compressing objects: 100% (132/132), done.
remote: Total 572 (delta 357), reused 285 (delta 281), pack-reused 159 (from 4)
Receiving objects: 100% (572/572), 101.75 KiB | 166.00 KiB/s, done.
Resolving deltas: 100% (389/389), done.
```

3. 对 `chirpstack-docker的docker-compose.yml` 进行配置。

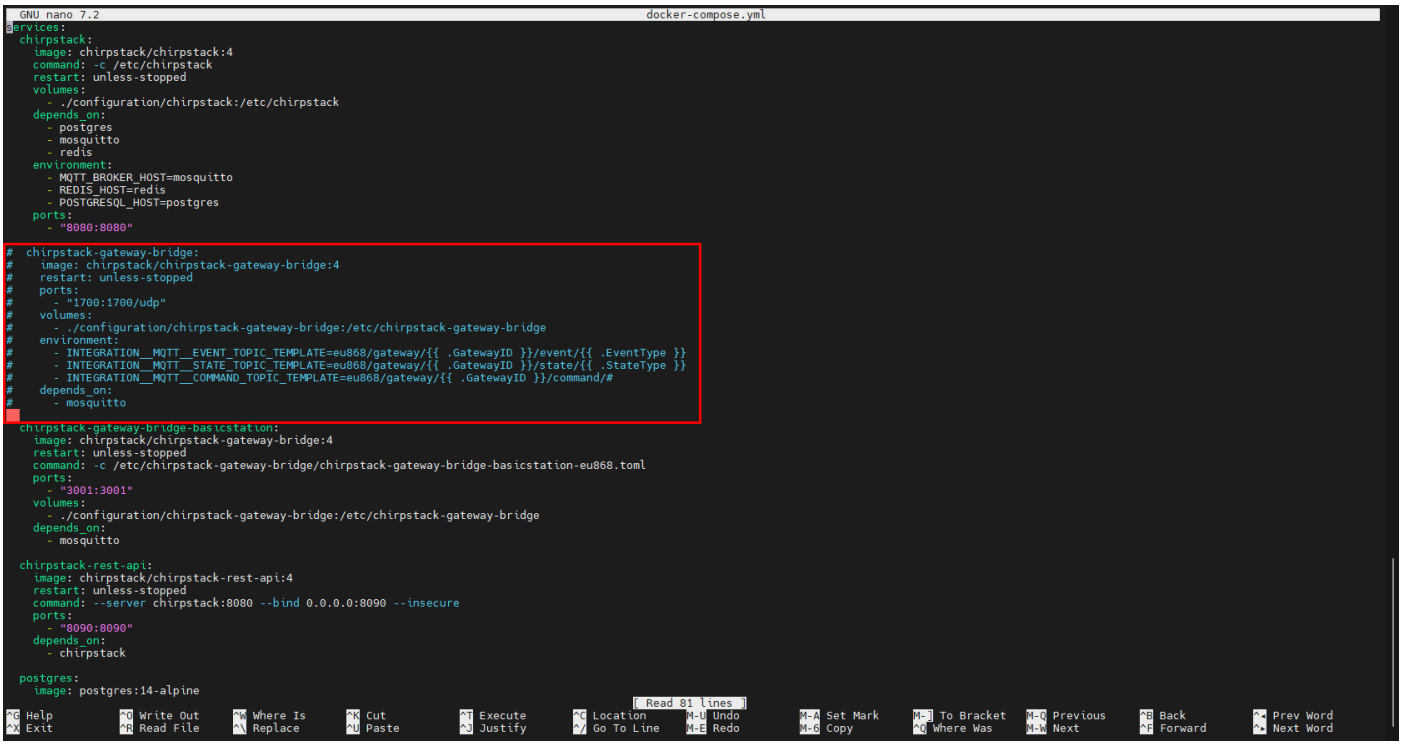
```sh
cd chirpstack-docker
nano docker-compose.yml
```

4. 对 `chirpstack-gateway-bridge` 部分进行注释。

```sh
services:
  chirpstack:
    image: chirpstack/chirpstack:4
    command: -c /etc/chirpstack
    restart: unless-stopped
    volumes:
      - ./configuration/chirpstack:/etc/chirpstack
    depends_on:
      - postgres
      - mosquitto
      - redis
    environment:
      - MQTT_BROKER_HOST=mosquitto
      - REDIS_HOST=redis
      - POSTGRESQL_HOST=postgres
    ports:
      - "8080:8080"

#  chirpstack-gateway-bridge:
#    image: chirpstack/chirpstack-gateway-bridge:4
#    restart: unless-stopped
#    ports:
#      - "1700:1700/udp"
#    volumes:
#      - ./configuration/chirpstack-gateway-bridge:/etc/chirpstack-gateway-bridge
#    environment:
#      - INTEGRATION__MQTT__EVENT_TOPIC_TEMPLATE=eu868/gateway/{{ .GatewayID }}/event/{{ .Even
#      - INTEGRATION__MQTT__STATE_TOPIC_TEMPLATE=eu868/gateway/{{ .GatewayID }}/state/{{ .Stat
#      - INTEGRATION__MQTT__COMMAND_TOPIC_TEMPLATE=eu868/gateway/{{ .GatewayID }}/command/#
#    depends_on:
#      - mosquitto

  chirpstack-gateway-bridge-basicstation:
    image: chirpstack/chirpstack-gateway-bridge:4
    restart: unless-stopped
    command: -c /etc/chirpstack-gateway-bridge/chirpstack-gateway-bridge-basicstation-eu868.to
    ports:
      - "3001:3001"
    volumes:
      - ./configuration/chirpstack-gateway-bridge:/etc/chirpstack-gateway-bridge
    depends_on:
      - mosquitto

  chirpstack-rest-api:
    image: chirpstack/chirpstack-rest-api:4
    restart: unless-stopped
    command: --server chirpstack:8080 --bind 0.0.0.0:8090 --insecure
    ports:
      - "8090:8090"
    depends_on:
```

```
      - chirpstack

  postgres:
    image: postgres:14-alpine
    restart: unless-stopped
    volumes:
      - ./configuration/postgresql/initdb:/docker-entrypoint-initdb.d
      - postgresqldata:/var/lib/postgresql/data
    environment:
      - POSTGRES_USER=chirpstack
      - POSTGRES_PASSWORD=chirpstack
      - POSTGRES_DB=chirpstack

  redis:
    image: redis:7-alpine
    restart: unless-stopped
    command: redis-server --save 300 1 --save 60 100 --appendonly no
    volumes:
      - redisdata:/data

  mosquitto:
    image: eclipse-mosquitto:2
    restart: unless-stopped
    ports:
      - "1883:1883"
    volumes:
      - ./configuration/mosquitto/config/:/mosquitto/config/

volumes:
  postgresqldata:
  redisdata:
```

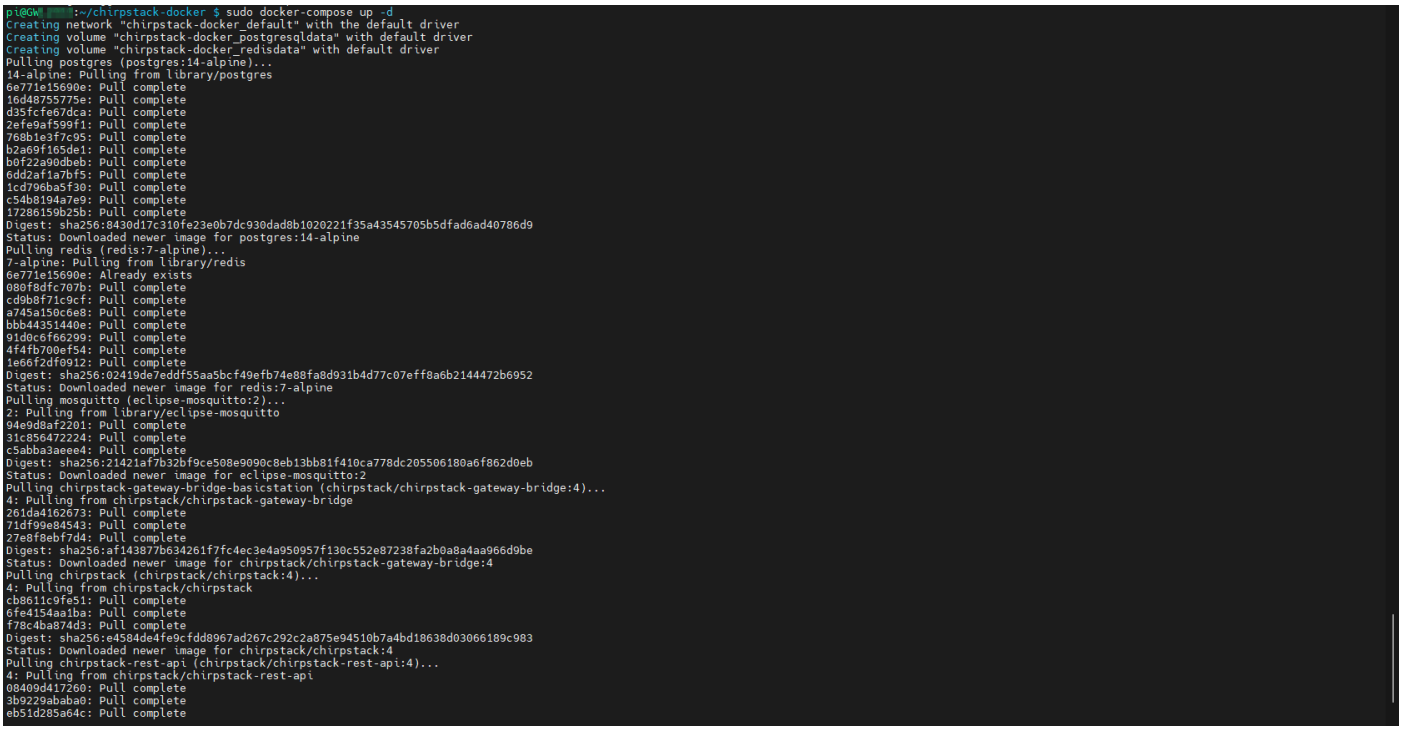
> 提示
>
> 用户可通过修改 `mosquitto` 部分配置mqtt服务。



3.10.4 启动chirpstack服务端

执行如下命令，在部署chirpstack服务端的设备上启动chirpstack服务端。

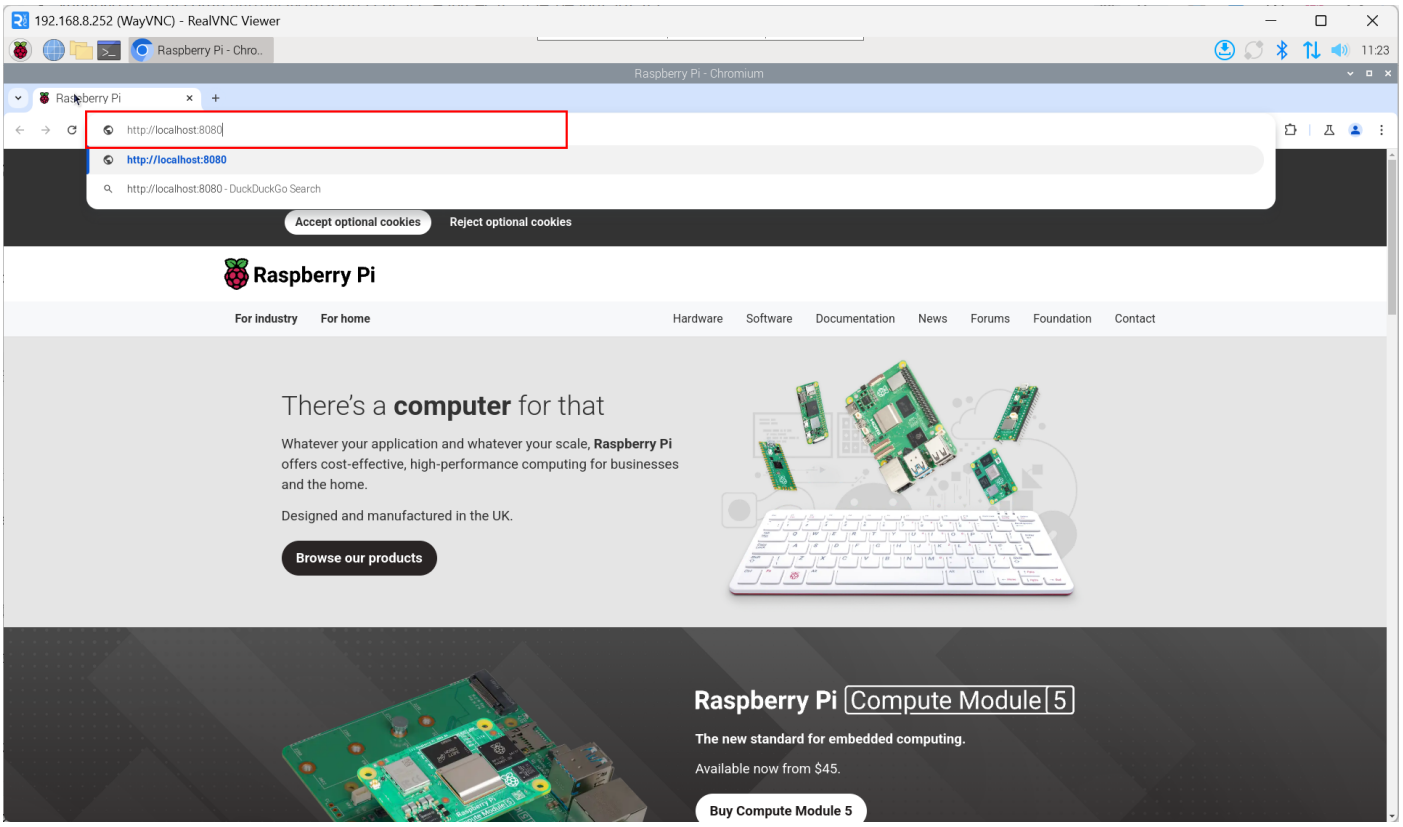
```sh
sudo docker-compose up -d
```



3.10.5 登录chirpstack服务管理界面

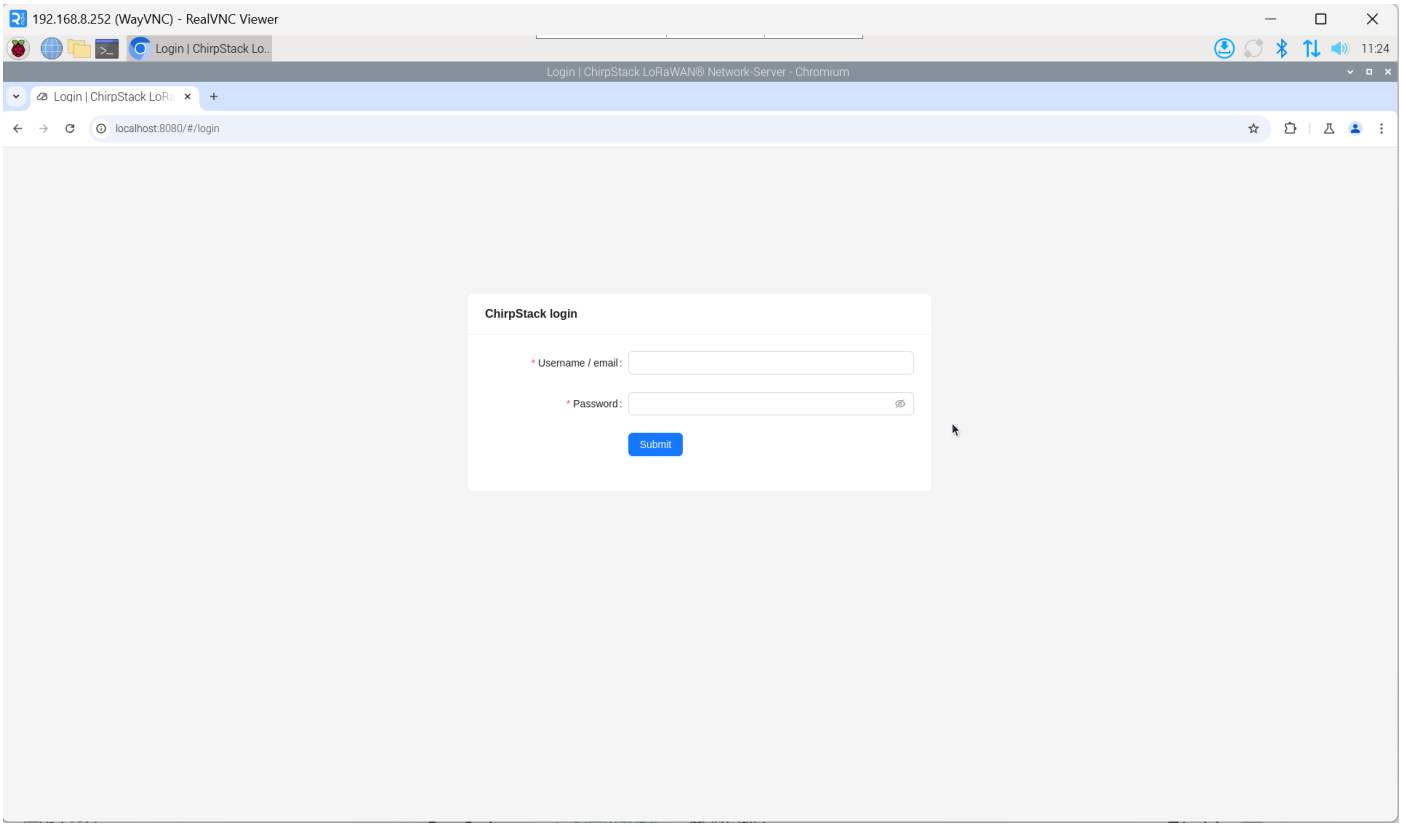
3.10.5.1 浏览器登录

1. 在PC端浏览器输入服务器的IP地址和端口号：http://localhost:8080，



2. 在登录界面输入用户名和密码。

> 提示
>
> 默认的用户名为 `admin` ，默认密码为 `admin` 。



3.10.5.2 获取LoRa网关ID

1. 执行如下命令，查看对应的LoRa文件。

```sh
systemctl status ed-lora.service
```

```
pi@GWL2010:~ $ systemctl status ed-lora.service
● ed-lora.service - EDATec LoRa Packet Forwarder Service
     Loaded: loaded (/lib/systemd/system/ed-lora.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-03-04 11:12:18 CST; 13min ago
   Main PID: 2447 (start.sh)
      Tasks: 5 (limit: 8731)
        CPU: 35.715s
     CGroup: /system.slice/ed-lora.service
             ├─2447 /bin/bash /opt/ed-lora/start.sh
             └─2452 /opt/ed-lora/lora_pkt_fwd -c /opt/ed-lora/conf/global_conf.json.US915

Mar 04 11:24:21 GWL2010 start.sh[2452]: ### Concentrator temperature: 30 C ###
Mar 04 11:24:21 GWL2010 start.sh[2452]: ##### END #####
Mar 04 11:24:21 GWL2010 start.sh[2452]: JSON up: {"stat":{"time":"2025-03-04 03:23:51 GMT","rxnb":0,"rxok":0,"rxfw":0,"ackr":0.0,"dwnb":0,"txnb":0,"temp":30.0}}
Mar 04 11:24:21 GWL2010 start.sh[2452]: ##### 2025-03-04 03:24:21 GMT #####
Mar 04 11:24:21 GWL2010 start.sh[2452]: ### [UPSTREAM] ###
Mar 04 11:24:21 GWL2010 start.sh[2452]: # RF packets received by concentrator: 0
Mar 04 11:24:21 GWL2010 start.sh[2452]: # CRC_OK: 0.00%, CRC_FAIL: 0.00%, NO_CRC: 0.00%
Mar 04 11:24:21 GWL2010 start.sh[2452]: # RF packets forwarded: 0 (0 bytes)
Mar 04 11:24:21 GWL2010 start.sh[2452]: # PUSH_DATA datagrams sent: 1 (123 bytes)
Mar 04 11:24:21 GWL2010 start.sh[2452]: # PUSH_DATA acknowledged: 0.00%
pi@GWL2010:~ $
```

2. 执行如下命令，查看LoRa网关的ID。

```sh
cat /opt/ed-lora/conf/global_conf.json.US915 | grep gateway
```

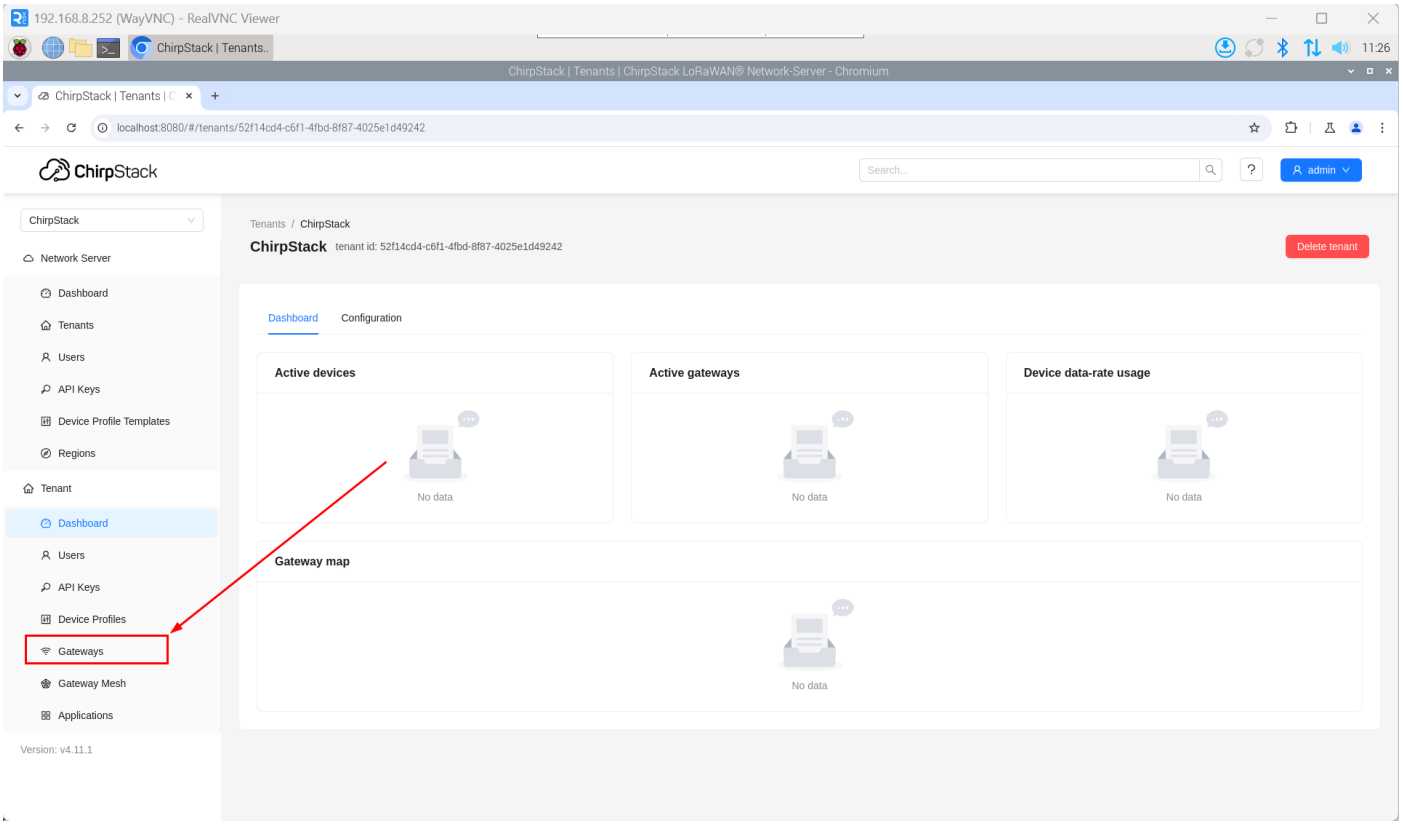
```
pi@GWL2010:~ $ cat /opt/ed-lora/conf/global_conf.json.US915 | grep gateway
    "gateway_conf": {
        "gateway_ID": "AA555A0000000000",
```

> 提示

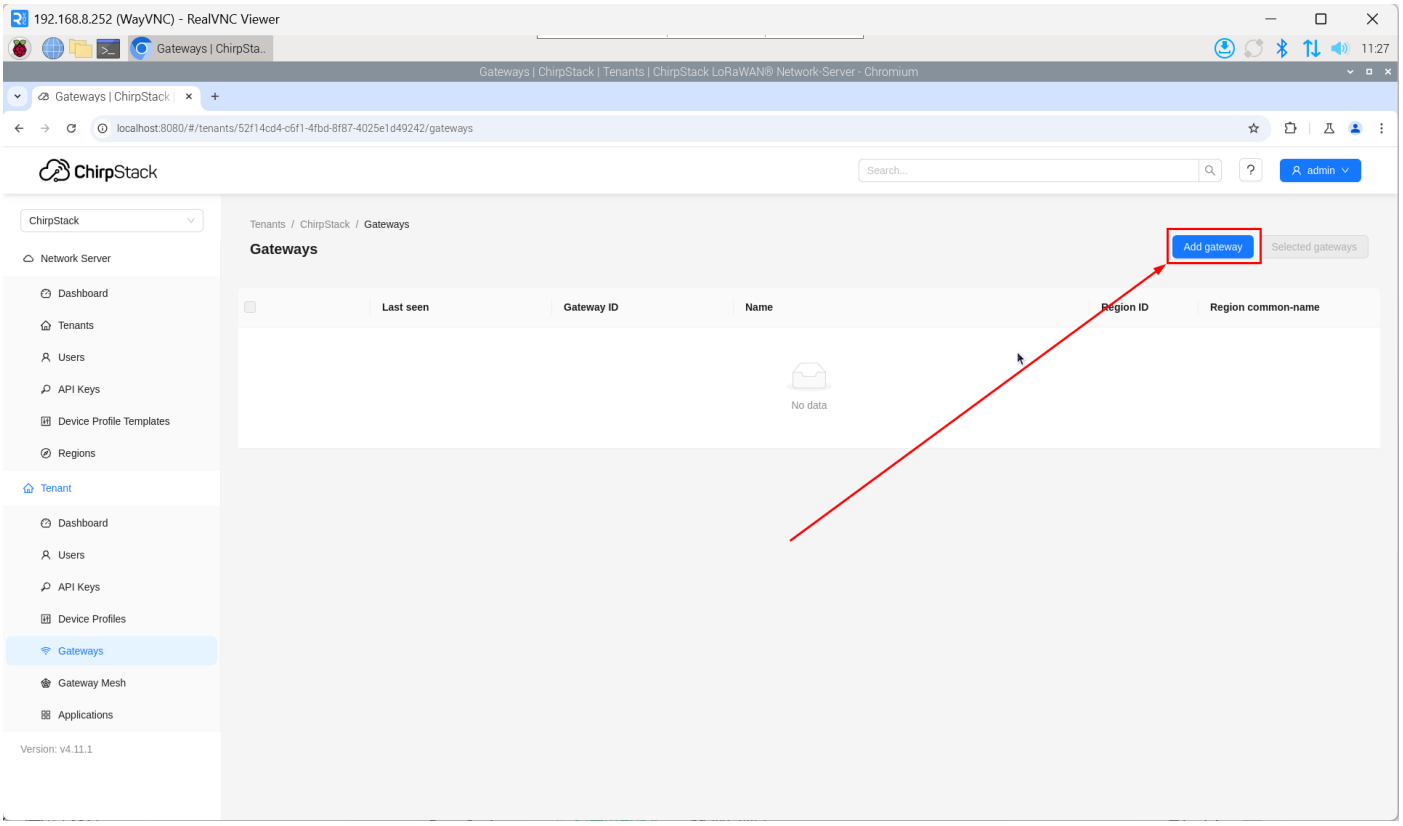
> 在chirpstack服务端添加LoRa网关时，需要填加对应的网关ID。

3.10.5.3 添加LoRa网关

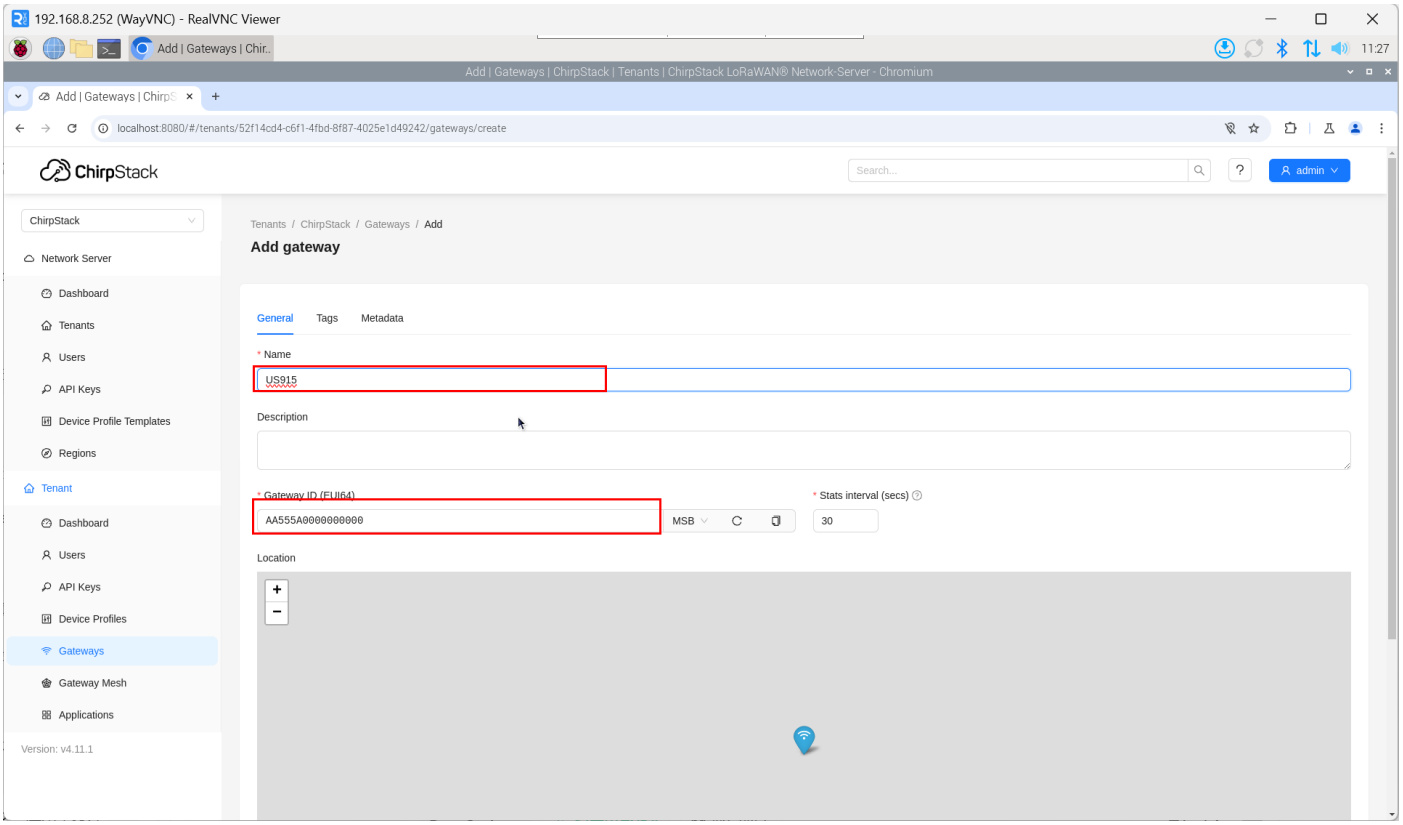
1. 在PC端浏览器打开chirpstack管理界面，单击“Gateway” -> “Add gateway”。

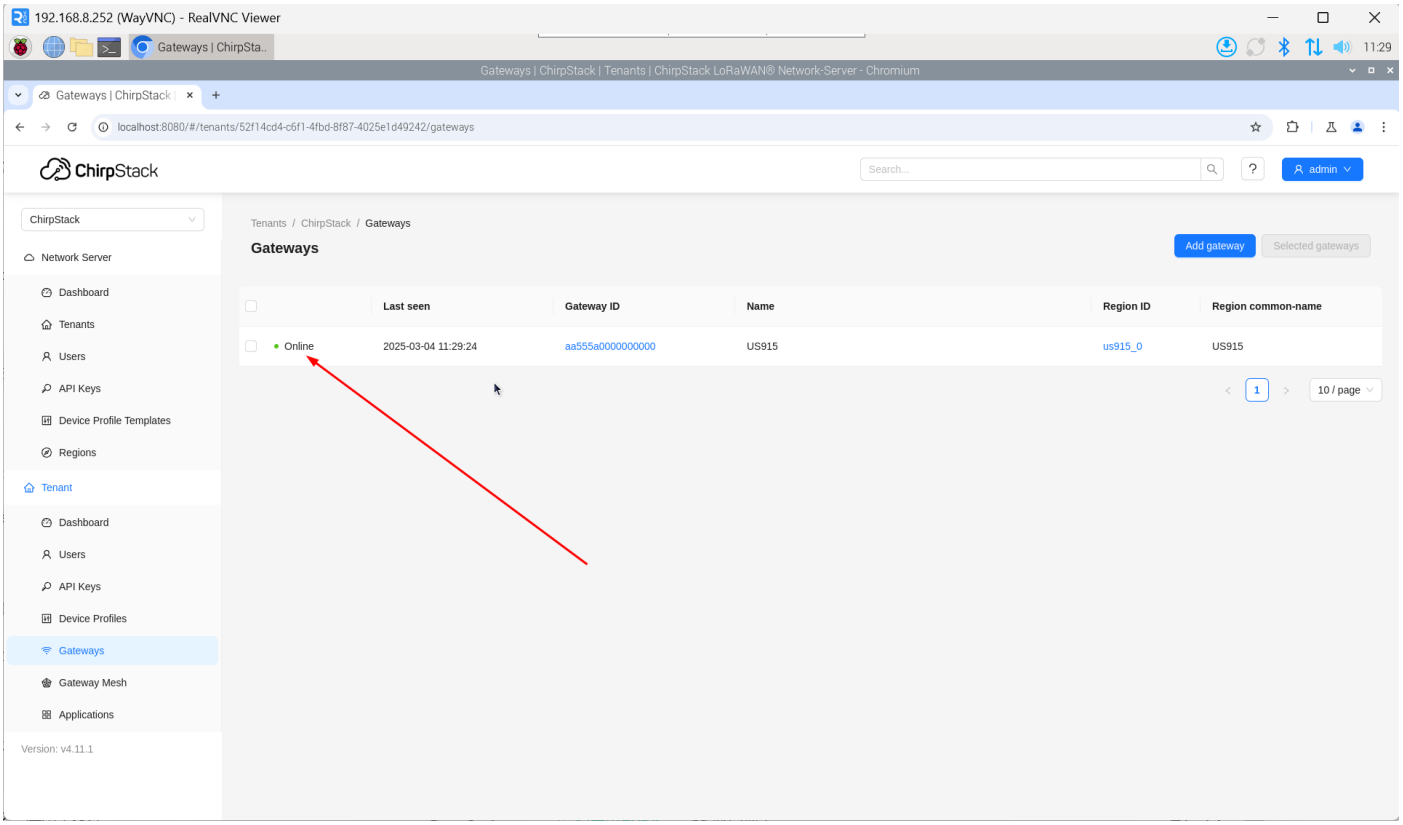


2. 单击“add”。



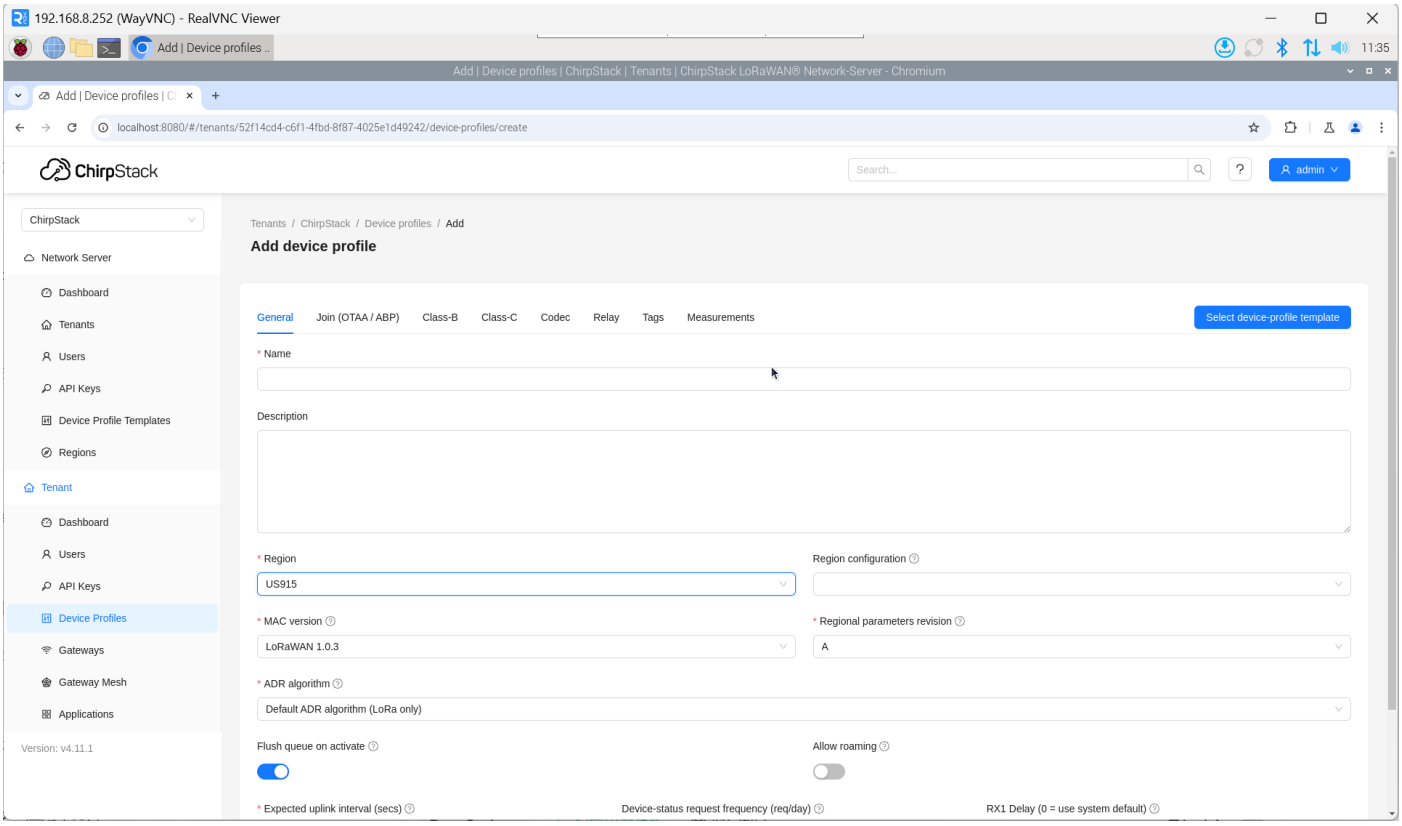
3. 填入设备对应的Gateway ID，并设置Name，然后单击“Submit”。如果网络连接正确，稍等片刻即可看到添加的网关变为Online状态。





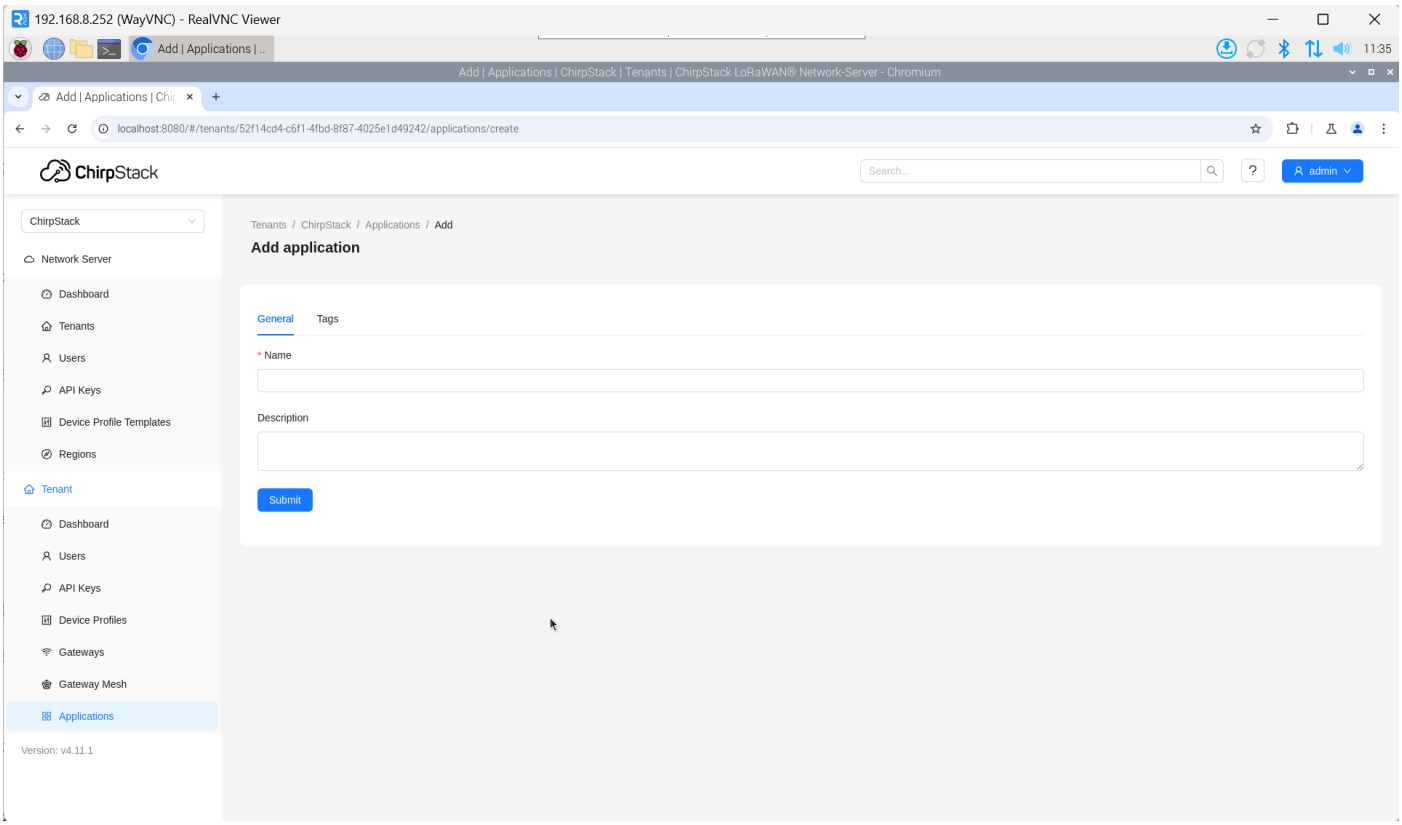
3.10.5.4 添加设备配置

单击“Device Profile” -> “Add device profile”，补充设备信息。



3.10.4.5 添加应用

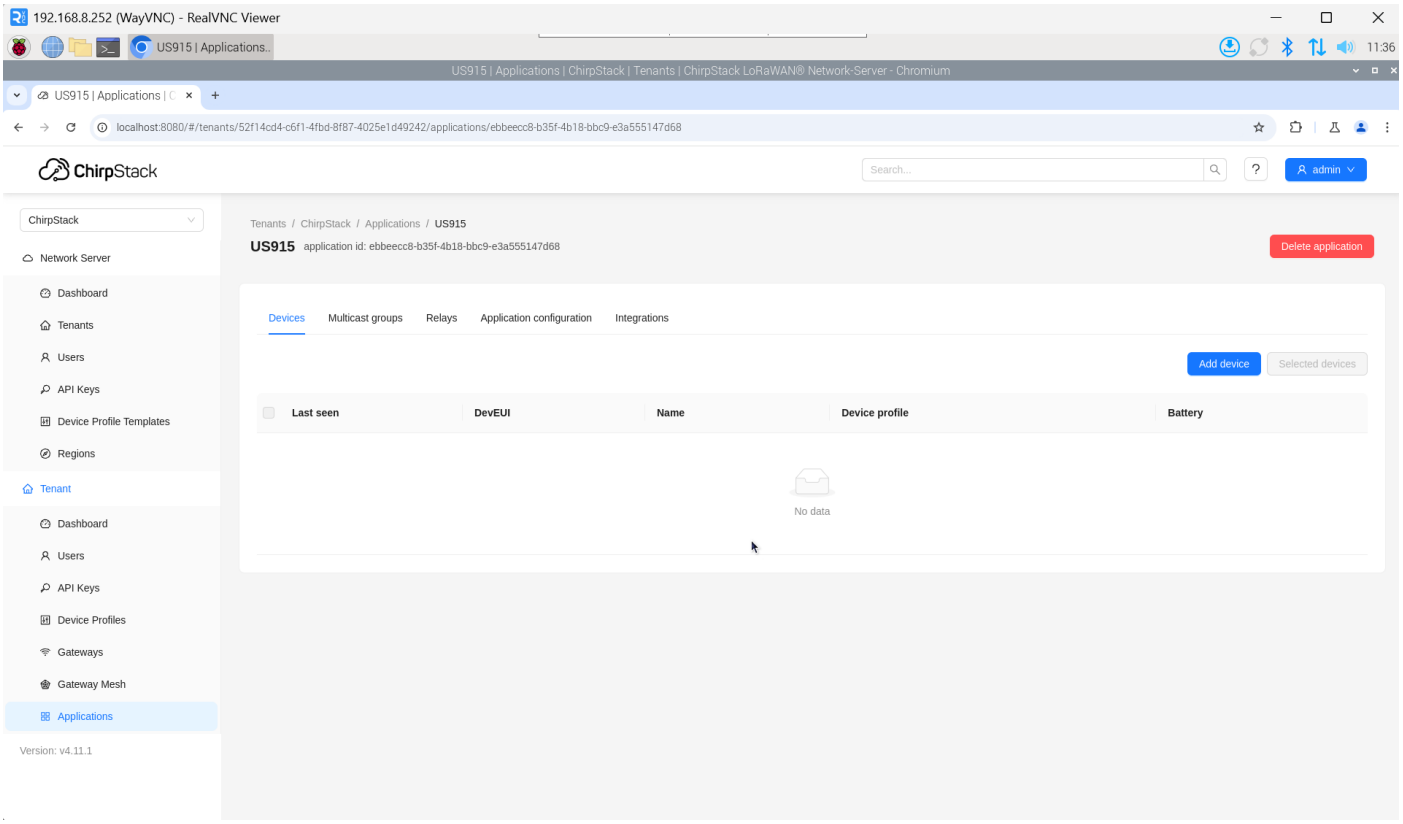
单击“Applications” -> “Add application”。

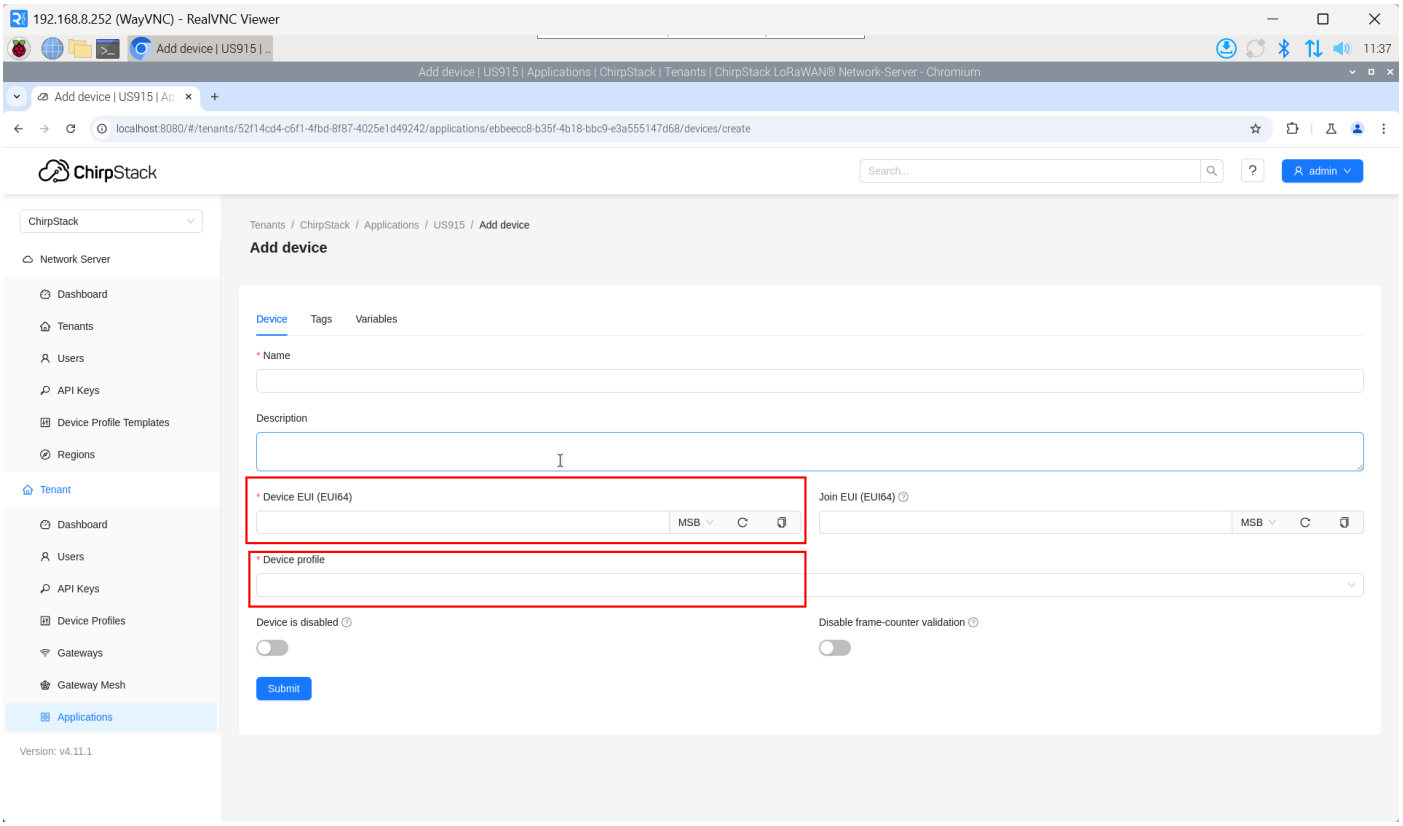


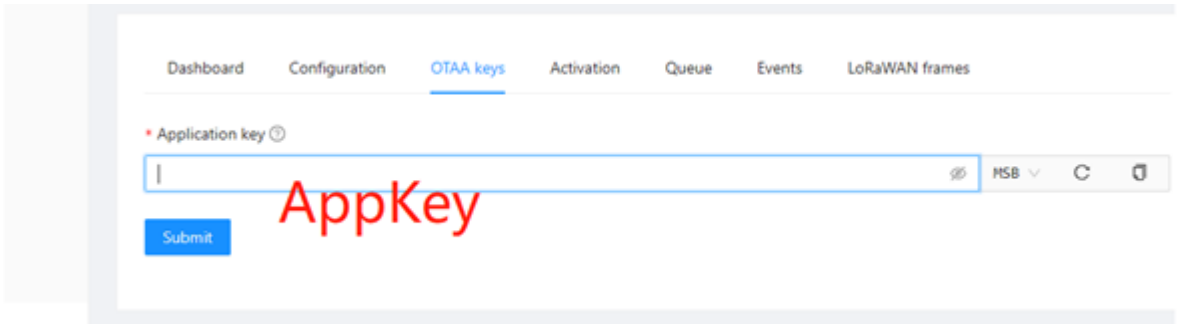
3.10.4.6 添加设备

LoRa终端产品的`DevEUI`和`AppKey`均由LoRa终端设备制造商提供。

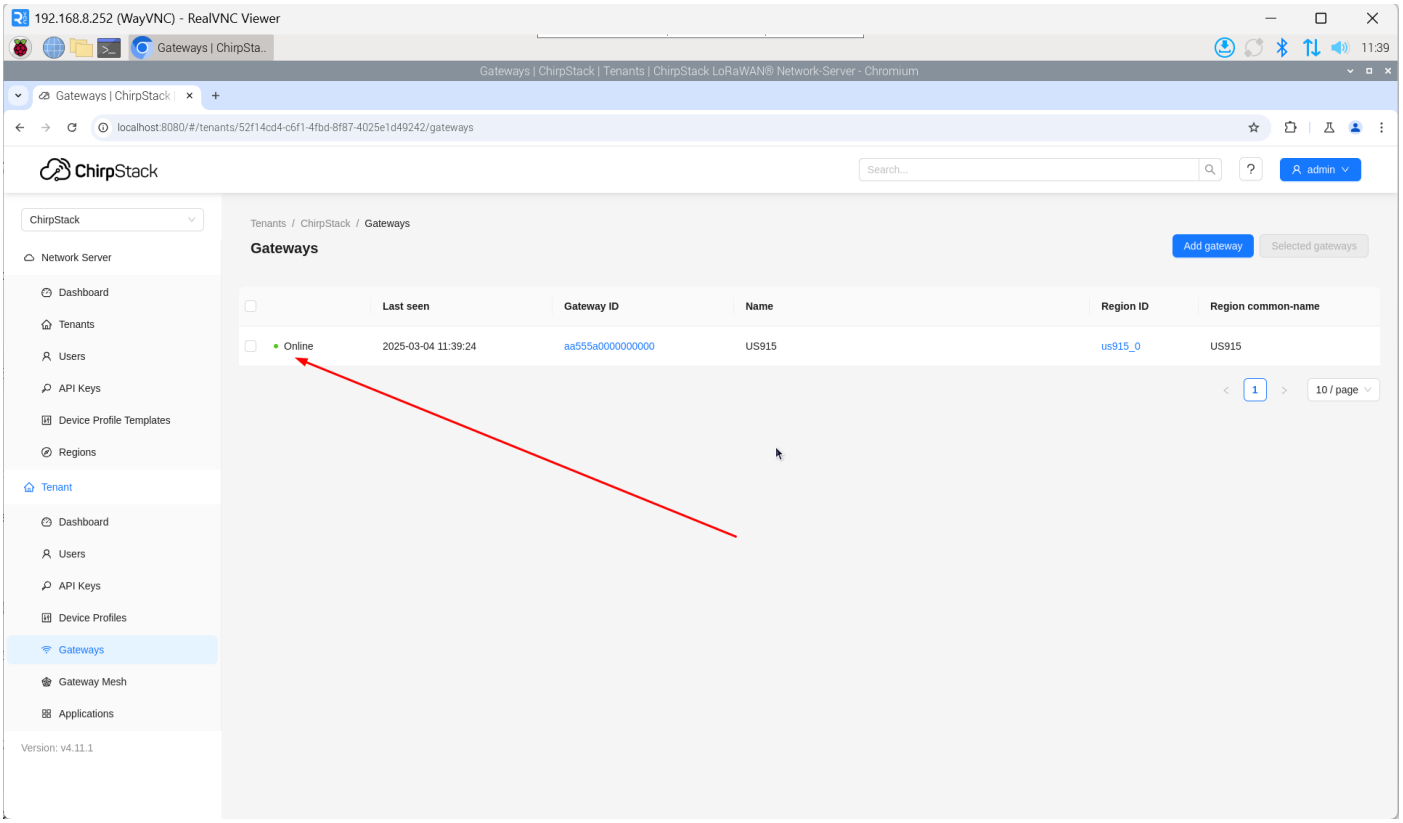
1. 单击“Application” -> “your application” -> “Add device”，添加LoRa终端设备。


192.168.8.252 (WayVNC) - RealVNC Viewer
US915 | Applications | ChirpStack | Tenants | ChirpStack LoRaWAN® Network-Server - Chromium
localhost:8080/#/tenants/52f14cd4-c6f1-4fbd-8f87-4025e1d49242/applications/ebbeecc8-b35f-4b18-bbc9-e3a555147d68
ChirpStack
admin
Tenants / ChirpStack / Applications / US915
US915 application id: ebbeecc8-b35f-4b18-bbc9-e3a555147d68
Delete application
Devices
Multicast groups
Relays
Application configuration
Integrations
Add device
Selected devices
Last seen
DevEUI
Name
Device profile
Battery
No data
Network Server
Dashboard
Tenants
Users
API Keys
Device Profile Templates
Regions
Tenant
Device Profiles
Gateways
Gateway Mesh
Applications
Version: v4.11.1


192.168.8.252 (WayVNC) - RealVNC Viewer
Add device | US915 | Applications | ChirpStack | Tenants | ChirpStack LoRaWAN® Network-Server - Chromium
localhost:8080/#/tenants/52f14cd4-c6f1-4fbd-8f87-4025e1d49242/applications/ebbeecc8-b35f-4b18-bbc9-e3a555147d68/devices/create
ChirpStack
admin
Tenants / ChirpStack / Applications / US915 / Add device
Add device
Device
Tags
Variables
* Name
Description
* Device EUI (EUI64)
MSB
Join EUI (EUI64)
* Device profile
Device is disabled
Disable frame-counter validation
Submit
Network Server
Dashboard
Tenants
Users
API Keys
Device Profile Templates
Regions
Tenant
Device Profiles
Gateways
Gateway Mesh
Applications
Version: v4.11.1


Dashboard
Configuration
OTAA keys
Activation
Queue
Events
LoRaWAN frames
* Application key
MSB
AppKey
Submit

2. 等待几分钟后，可看到设备变成online状态。



3.11 加密芯片

ED-GWL2110板载ATECC608加密芯片，它连接到i2c-1总线，器件默认地址为0x60。

atecc: https://github.com/wirenboard/atecc-util

```sh
atecc -b 1 -c 'serial'
```

4 安装操作系统（可选）

设备出厂时，默认带有操作系统。如果在使用过程中操作系统被损坏或者用户需要更换操作系统，则需要重新下载合适的系统镜像并进行烧录。我司支持通过先安装标准Raspberry Pi OS，再安装Firmware包，来实现操作系统的安装。

下文介绍镜像下载、镜像烧录和安装Firmware包的具体操作。

4.1 镜像下载

可根据实际的需要下载对应的Raspberry Pi官方系统镜像，下载路径如下表：

OS	下载路径
Raspberry Pi OS(Desktop) 64-bit-bookworm (Debian 12)	https://downloads.raspberrypi.com/raspios_arm64/images/raspios_arm64-2024-07-04/2024-07-04-raspios-bookworm-arm64.img.xz (https://downloads.raspberrypi.com/raspios_arm64/images/raspios_arm64-2024-07-04/2024-07-04-raspios-bookworm-arm64.img.xz)
Raspberry Pi OS(Lite) 64-bit-bookworm (Debian 12)	https://downloads.raspberrypi.com/raspios_lite_arm64/images/raspios_lite_arm64-2024-07-04/2024-07-04-raspios-bookworm-arm64-lite.img.xz (https://downloads.raspberrypi.com/raspios_lite_arm64/images/raspios_lite_arm64-2024-07-04/2024-07-04-raspios-bookworm-arm64-lite.img.xz)
Raspberry Pi OS(Desktop) 32-bit-bookworm (Debian 12)	https://downloads.raspberrypi.com/raspios_armhf/images/raspios_armhf-2024-07-04/2024-07-04-raspios-bookworm-armhf.img.xz (https://downloads.raspberrypi.com/raspios_armhf/images/raspios_armhf-2024-07-04/2024-07-04-raspios-bookworm-armhf.img.xz)
Raspberry Pi OS(Lite) 32-bit-bookworm (Debian 12)	https://downloads.raspberrypi.com/raspios_lite_armhf/images/raspios_lite_armhf-2024-07-04/2024-07-04-raspios-bookworm-armhf-lite.img.xz (https://downloads.raspberrypi.com/raspios_lite_armhf/images/raspios_lite_armhf-2024-07-04/2024-07-04-raspios-bookworm-armhf-lite.img.xz)

4.2 镜像烧录

ED-GWL2110支持从SD卡启动系统，可根据实际应用参考下文进行烧录。

4.2.1 SD卡烧录

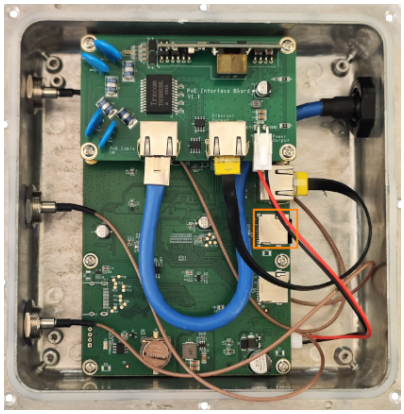
建议使用Raspberry Pi官方烧录工具，下载路径如下：

- Raspberry Pi Imager : https://downloads.raspberrypi.org/imager/imager_latest.exe (https://downloads.raspberrypi.org/imager/imager_latest.exe)
- SD Card Formatter : https://www.sdcardformatter.com/download/ (https://www.sdcardformatter.com/download/)
- Rpiboot : https://github.com/raspberrypi/usbboot/raw/master/win32/rpiboot_setup.exe (https://github.com/raspberrypi/usbboot/raw/master/win32/rpiboot_setup.exe)

前提条件：

- 已完成烧录工具的下载，并安装至电脑。
- 已打开设备外壳并拔出Micro SD卡。
  1. 使用螺丝刀逆时针拧下ED-GWL2110底部的9颗螺钉，如下图所示。

  2. 在如下图标记位置拔出Micro SD卡。

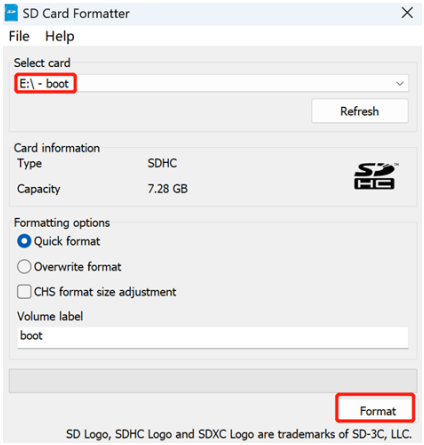


- 已获取待烧录的镜像文件。
- 已准备一个SD卡读卡器。
- 已断开电源。

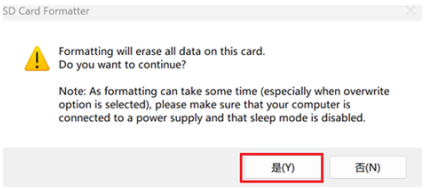
操作步骤：

操作步骤以Windows系统为例进行说明。

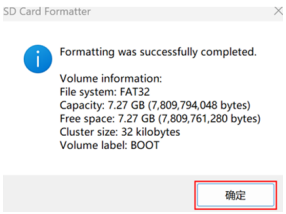
1. 将Micro SD卡插入读卡器,然后插入电脑的USB接口。
2. 打开SD Card Formatter，选择被格式化的盘符，单击右下方“Format”进行格式化。



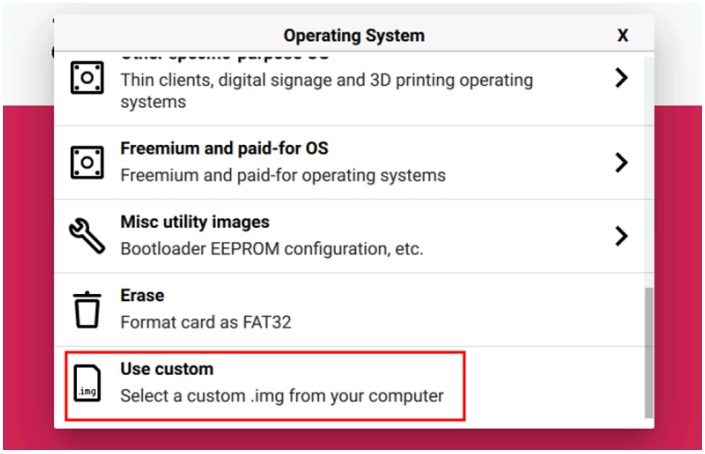
3. 在弹出的提示框中，单击“是”。



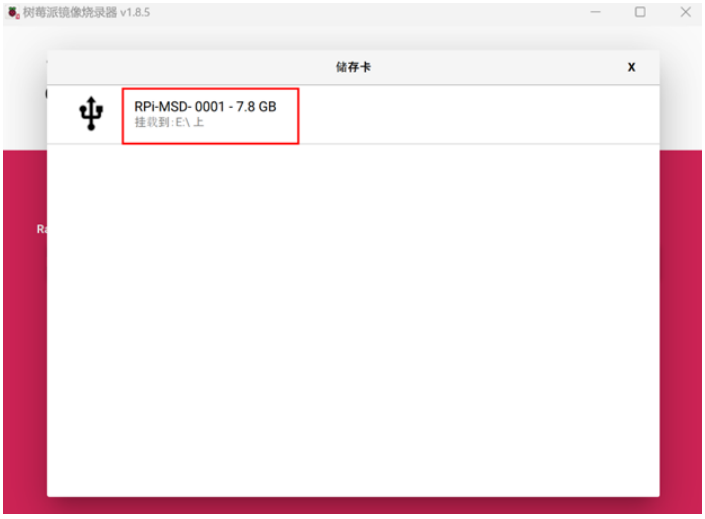
4. 格式化完成后，在提示框中单击 “确定”。



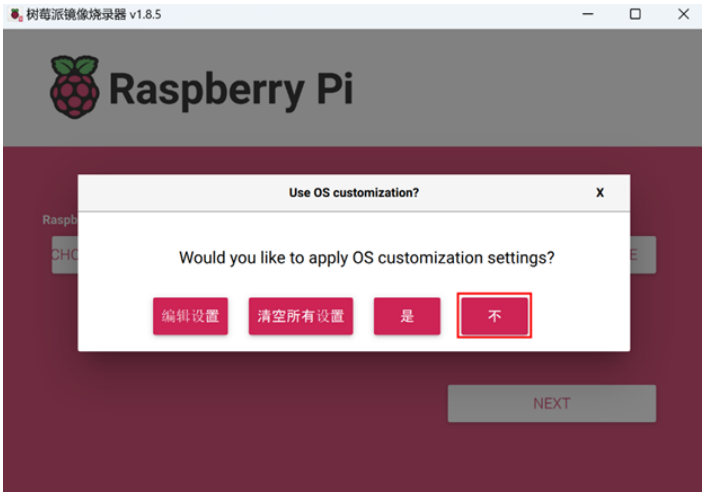
5. 关闭SD Card Formatter。
6. 打开Raspberry Pi Imager，单击“选择操作系统”，在弹出的窗格中选择“Use custom”。



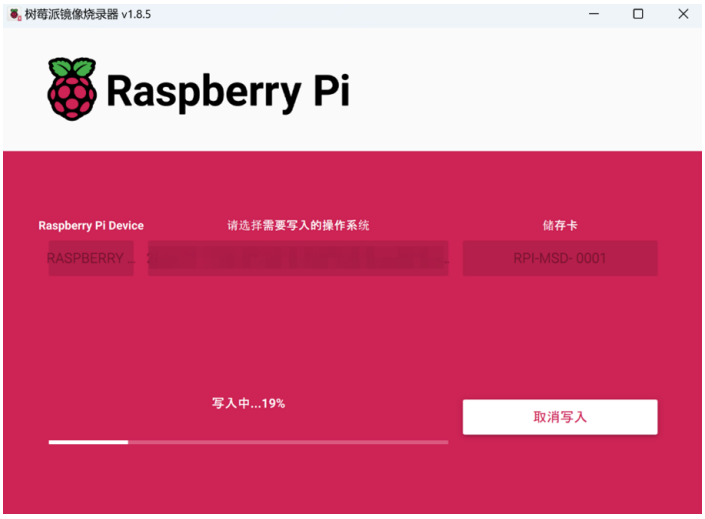
7. 根据提示，在自定义路径下选择已获取的镜像文件，并返回至烧录主界面。
8. 单击“选择SD卡”，在“存储卡”界面选择默认的SD卡，并返回至烧录主界面。



9. 单击“NEXT”，在弹出的“Use OS customization？”提示框中选择“不”，开始写入镜像。



10. 在弹出的“警告”提示框中选择‘”是“，开始写入镜像。



11. 待镜像写入完成后，会进行文件的验证。

12. 验证完成后，弹出“烧录成功”提示框，单击“继续”完成烧录。
13. 关闭Raspberry Pi Imager，取下读卡器和SD卡，将SD卡重新插入设备中。

4.2 安装Firmware包

在ED-GWL2110系列上烧录标准的Raspberry Pi OS后，需要通过添加edatec apt源和安装firmware包来配置系统，使系统能够正常使用。
针对ED-GWL2110配备有三种不同的LoRa WAN协议频段，470MHz(CN470)，868MHz(EU868)，915MHz(US915)安装的firmware的名称不同，对应如下表：

型号	firmware名称
470MHz(CN470)	gwl2110_470
868MHz(EU868)	gwl2110_868
915MHz(US915)	gwl2110_915

前提条件：

- 已完成Raspberry Pi标准的bookworm镜像的烧录。
- 设备已正常启动，且已完成相关的启动配置。

操作步骤：

1. 设备正常启动后，在命令窗格依次执行如下命令，添加edatec apt源和安装Firmware包。

- 470MHz(CN470)

```sh
curl -s https://apt.edatec.cn/bsp/ed-install.sh | sudo bash -s gwl2110_470
```

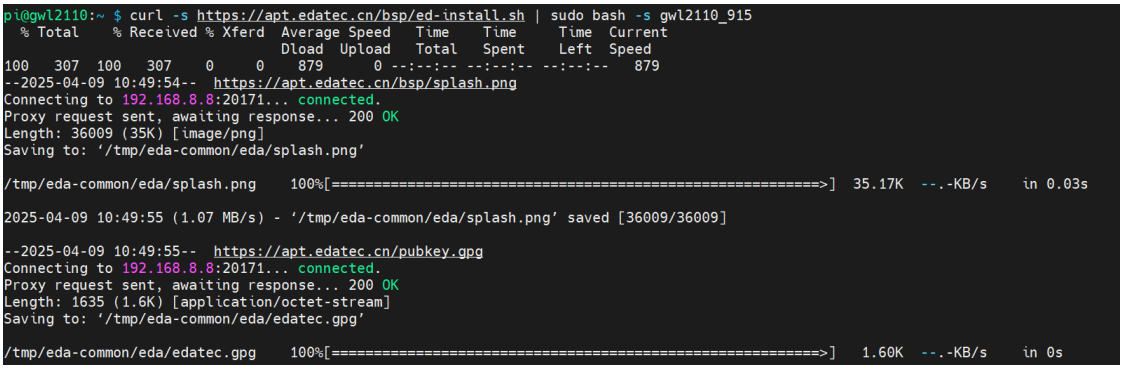
- 868MHz(EU868)

```sh
curl -s https://apt.edatec.cn/bsp/ed-install.sh | sudo bash -s gwl2110_868
```

- 915MHz(US915)

```sh
curl -s https://apt.edatec.cn/bsp/ed-install.sh | sudo bash -s gwl2110_915
```

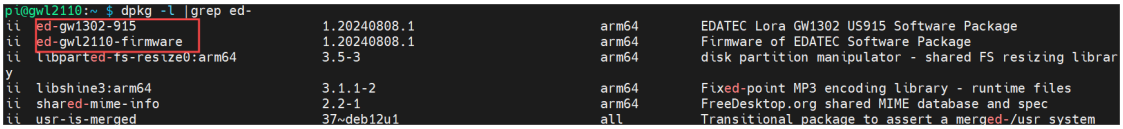
下图仅以915MHz(US915)为例。



2. 安装完成后，设备自动重启。
3. 执行如下命令，检查firmware包是否安装成功。

```sh
dpkg -l | grep ed-
```

下图中的结果表示firmware包已安装成功。



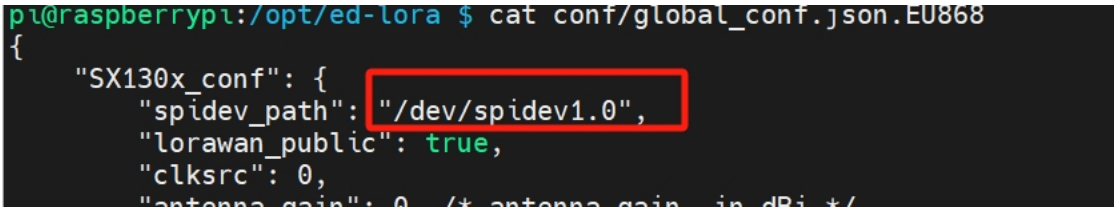
> 提示:
>
> 如果安装了错误的firmware包，可以执行 `sudo apt-get --purge remove package` 进行删除，其中package为包的名字。

4. 执行如下命令，使能i2c接口。

```sh
sudo raspi-config nonint do_i2c 0
```

5. 依次执行如下命令查看LoRa的SPI配置，并设置 `"the spidev_path"` 为 `"/dev/spidev1.0"` 。

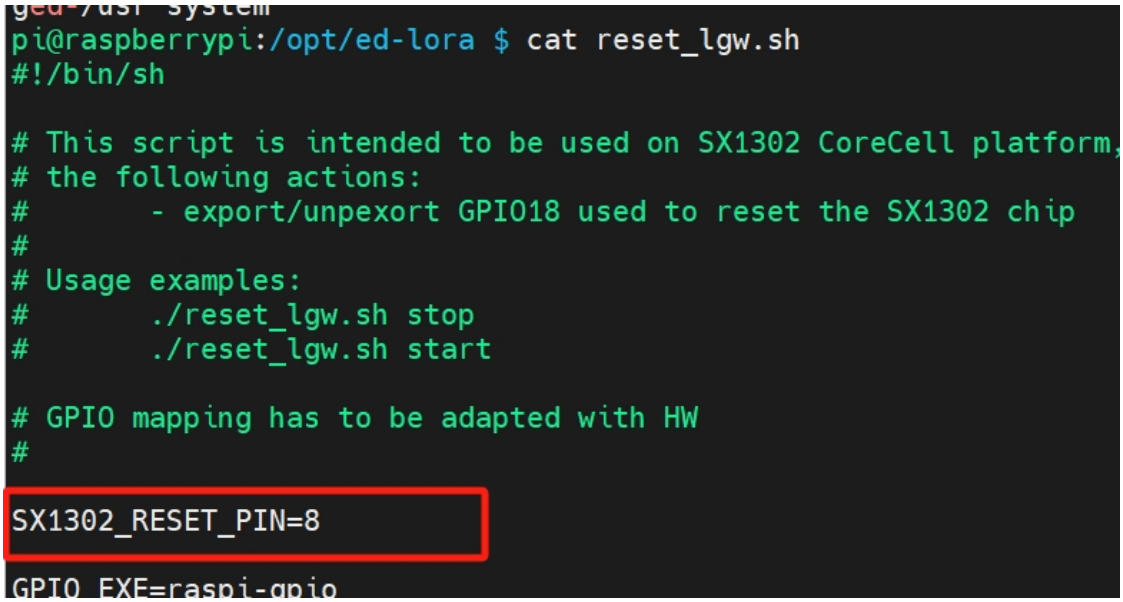
```sh
cd /opt/ed-lora
cat conf/global_conf.json.EU868
```



> 提示
>
> 如果使用的LoRa模块为US915或CN470，请将 `EU868` 改为 `US915` 或 `CN470` 。

6. 依次执行如下命令查看LoRa的复位引脚，并修改LoRa的复位引脚为8。

```sh
cd /opt/ed-lora
cat reset_lgw.sh
```

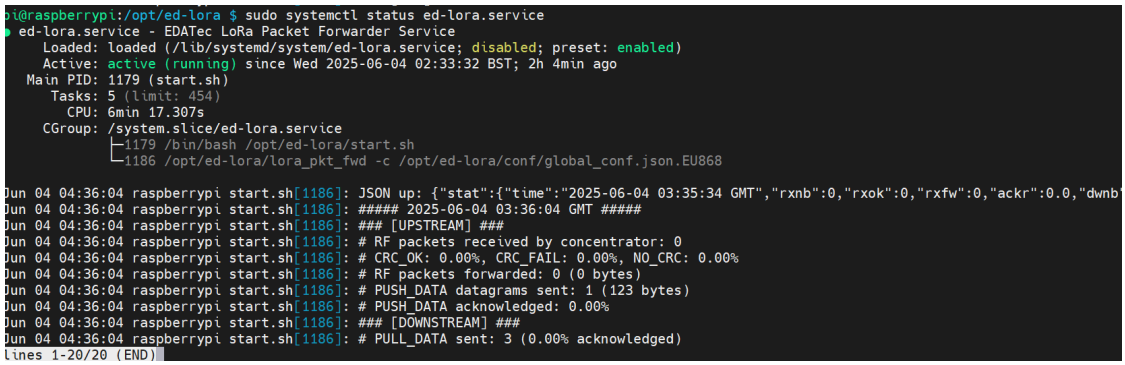


7. 依次执行如下命令，重启设备。

```sh
sudo reboot
```

8. 依次执行如下命令，查看LoRa服务的状态

```sh
sudo systemctl status ed-lora.service
```



- 如果LoRa服务已开启，则配置完成。
- 如果LoRa服务未开启，请通过执行 `sudo systemctl start ed-lora.service` 命令来手动开启LoRa服务。