# LoRaWAN Network Setup Manual - Ubuntu 22.04

## Overview

This manual provides step-by-step instructions for setting up a complete LoRaWAN network infrastructure on Ubuntu 22.04. The setup includes ChirpStack (LoRaWAN Network Server), EMQX (MQTT Broker), InfluxDB2 (Time Series Database), and Grafana (Data Visualization).

## Architecture Components

- **ChirpStack**: Open-source LoRaWAN Network Server
- **ChirpStack Gateway Bridge**: Connects LoRaWAN gateways to the network server
- **EMQX**: High-performance MQTT broker for IoT messaging
- **PostgreSQL**: Database backend for ChirpStack
- **Redis**: In-memory data structure store for caching
- **InfluxDB2**: Time-series database for sensor data storage
- **Grafana**: Analytics and monitoring platform

## Prerequisites

- Ubuntu 22.04 LTS server
- Root or sudo privileges
- Stable internet connection
- At least 4GB RAM and 20GB disk space

---

## Step 1: System Preparation

Update your system packages to ensure you have the latest security patches and dependencies.

```
sudo apt update
sudo apt upgrade -y
```

---

## Step 2: ChirpStack Installation

### 2.1 Install Required Dependencies

Install the core services that ChirpStack depends on:

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

**Components installed:**

- **Mosquitto**: MQTT broker (will be replaced by EMQX later)
- **Redis**: Caching and session storage
- **PostgreSQL**: Primary database for ChirpStack

### 2.2 Configure PostgreSQL Database

Set up the database for ChirpStack with proper user permissions:

```
sudo -u postgres psql
```

Execute the following SQL commands in the PostgreSQL prompt:

```
create role chirpstack with login password 'chirpstack';
create database chirpstack with owner chirpstack;
\c chirpstack
create extension pg_trgm;
\q
```

**What this does:**

- Creates a dedicated user `chirpstack` with login privileges
- Creates a database owned by the chirpstack user
- Enables the `pg_trgm` extension for text search capabilities

## 2.3 Add ChirpStack Repository

Install repository management tools and add the ChirpStack package repository:

```
sudo apt install apt-transport-https dirmngr
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 1CE2AFD
36DBCCA00
sudo echo "deb https://artifacts.chirpstack.io/packages/4.x/deb stable mai
n" | sudo tee /etc/apt/sources.list.d/chirpstack.list
sudo apt update
```

## 2.4 Install ChirpStack Gateway Bridge

Install and configure the gateway bridge component:

```
sudo apt install chirpstack-gateway-bridge
```

## 2.5 Configure Gateway Bridge

Edit the configuration file to match your region:

```
sudo nano /etc/chirpstack-gateway-bridge/chirpstack-gateway-bridge.tom
l
```

**For IN865 region (India), update the `[integration.mqtt]` section:**

```
[integration.mqtt]
event_topic_template="in865/gateway/{{ .GatewayID }}/event/{{ .EventTyp
e }}"
command_topic_template="in865/gateway/{{ .GatewayID }}/command/#"
```

**Note**: Replace `in865` with your appropriate region code:

- `eu868` for Europe

- `us915` for North America
- `as923` for Asia-Pacific
- `au915` for Australia

## 2.6 Start Gateway Bridge Service

Enable and start the ChirpStack Gateway Bridge:

```
# Start the service
sudo systemctl start chirpstack-gateway-bridge

# Enable automatic startup on boot
sudo systemctl enable chirpstack-gateway-bridge
```

## 2.7 Install ChirpStack Network Server

Install the main ChirpStack application:

```
sudo apt install chirpstack
```

## 2.8 Configure ChirpStack

The configuration files are located at `/etc/chirpstack/`. You'll find:

- `chirpstack.toml`: Global configuration
- Various region-specific configuration files

For basic setup, the default configuration should work. For advanced configuration, edit:

```
sudo nano /etc/chirpstack/chirpstack.toml
```

## 2.9 Start ChirpStack Service

Enable and start the ChirpStack Network Server:

```
# Start the service
sudo systemctl start chirpstack
```

```
# Enable automatic startup on boot
sudo systemctl enable chirpstack
```

### 2.10 Monitor ChirpStack Logs

Check the service status and logs:

```
# View real-time logs
sudo journalctl -f -n 100 -u chirpstack

# Check service status
sudo systemctl status chirpstack
```

---

## Step 3: EMQX MQTT Broker Installation

### 3.1 Download and Install EMQX

Download the EMQX Enterprise package and install it:

```
wget https://www.emqx.com/en/downloads/enterprise/5.9.0/emqx-enterprise-5.9.0-ubuntu22.04-amd64.deb
sudo apt install ./emqx-enterprise-5.9.0-ubuntu22.04-amd64.deb
```

### 3.2 Switch from Mosquitto to EMQX

Stop the current services and start EMQX:

```
# Stop existing services
sudo systemctl stop mosquitto
sudo systemctl stop chirpstack
sudo systemctl stop chirpstack-gateway-bridge

# Start EMQX and enable it
sudo systemctl start emqx
sudo systemctl enable emqx

# Restart ChirpStack services with EMQX
sudo systemctl start chirpstack-gateway-bridge
sudo systemctl enable chirpstack-gateway-bridge
sudo systemctl start chirpstack
sudo systemctl enable chirpstack
```

**Why EMQX?**

- Higher performance and scalability
- Better monitoring and management features
- Enhanced security features
- Built-in clustering support

---

## Step 4: InfluxDB2 Installation

### 4.1 Add InfluxDB Repository

Download and verify the repository key:

```
curl --silent --location -O \
   https://repos.influxdata.com/influxdata-archive.key

echo "943666881a1b8d9b849b74caebf02d3465d6beb716510d86a39f6c8
e8dac7515  influxdata-archive.key" \
   | sha256sum --check - && cat influxdata-archive.key \
   | gpg --dearmor \
   | sudo tee /etc/apt/trusted.gpg.d/influxdata-archive.gpg > /dev/null \
   && echo 'deb [signed-by=/etc/apt/trusted.gpg.d/influxdata-archive.gpg]
https://repos.influxdata.com/debian stable main' \
   | sudo tee /etc/apt/sources.list.d/influxdata.list
```

### 4.2 Install InfluxDB2

Update package list and install InfluxDB2:

```
sudo apt-get update && sudo apt-get install influxdb2
```

### 4.3 Start InfluxDB Service

Enable and start the InfluxDB service:

```
# Start the service
sudo service influxdb start

# Enable automatic startup
sudo systemctl enable influxdb.service

# Check service status
sudo service influxdb status
```

**Access InfluxDB2 Web UI:**

- URL: `http://your-server-ip:8086`
- Complete the initial setup through the web interface

---

## Step 5: Grafana Installation

### 5.1 Install Prerequisites

Install required packages for Grafana:

```
sudo apt-get install -y apt-transport-https software-properties-common w
get
```

### 5.2 Add Grafana Repository

Create keyring directory and add Grafana GPG key:

```
sudo mkdir -p /etc/apt/keyrings/
wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /e
tc/apt/keyrings/grafana.gpg > /dev/null
```

Add Grafana repository:

```
echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.c
om stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.c
om beta main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
```

### 5.3 Install Grafana

Update package list and install Grafana:

```
# Update package list
sudo apt-get update

# Install Grafana OSS
sudo apt-get install grafana
```

### 5.4 Start Grafana Service

Enable and start Grafana:

```
# Start Grafana
sudo systemctl start grafana-server

# Enable automatic startup
sudo systemctl enable grafana-server

# Check status
sudo systemctl status grafana-server
```

**Access Grafana Web UI:**

- URL: `http://your-server-ip:3000`
- Default credentials: `admin/admin`

---

## Step 6: Verification and Access

### 6.1 Service Status Check

Verify all services are running:

```
sudo systemctl status chirpstack
sudo systemctl status chirpstack-gateway-bridge
sudo systemctl status emqx
sudo systemctl status influxdb
sudo systemctl status grafana-server
sudo systemctl status redis-server
sudo systemctl status postgresql
```

### 6.2 Web Interfaces

Access the following web interfaces to complete setup:

1. **ChirpStack**: `http://your-server-ip:8080`
   - Default credentials: `admin/admin`
2. **EMQX Dashboard**: `http://your-server-ip:18083`
   - Default credentials: `admin/public`
3. **InfluxDB2**: `http://your-server-ip:8086`
   - Complete initial setup wizard
4. **Grafana**: `http://your-server-ip:3000`
   - Default credentials: `admin/admin`

### 6.3 Firewall Configuration

If using UFW firewall, open required ports:

```
sudo ufw allow 8080  # ChirpStack
sudo ufw allow 18083 # EMQX Dashboard
sudo ufw allow 8086  # InfluxDB2
sudo ufw allow 3000  # Grafana
sudo ufw allow 1883  # MQTT
sudo ufw allow 1700/udp  # LoRaWAN Gateway
```

---

## Step 7: Initial Configuration

### 7.1 ChirpStack Setup

1. Access ChirpStack web interface
2. Create device profiles for your LoRaWAN devices
3. Add gateways to the network
4. Register devices and applications

### 7.2 Data Integration

1. Configure InfluxDB2 bucket for sensor data
2. Set up Grafana data source pointing to InfluxDB2
3. Create dashboards for monitoring LoRaWAN network and device data

---

## Troubleshooting

### Common Issues

1. **Service fails to start**: Check logs with `sudo journalctl -u service-name`
2. **Port conflicts**: Ensure no other services are using the same ports
3. **Database connection issues**: Verify PostgreSQL is running and credentials are correct
4. **Gateway connection problems**: Check firewall settings and network configuration

### Log Locations

- ChirpStack: `sudo journalctl -u chirpstack`
- Gateway Bridge: `sudo journalctl -u chirpstack-gateway-bridge`
- EMQX: `/var/log/emqx/`
- InfluxDB2: `sudo journalctl -u influxdb`
- Grafana: `/var/log/grafana/`

---

## Security Recommendations

1. Change default passwords for all services
2. Configure SSL/TLS certificates for web interfaces
3. Set up proper firewall rules
4. Regular security updates
5. Database backup procedures
6. Monitor system logs regularly

---

## Conclusion

Your LoRaWAN network infrastructure is now ready. You can begin connecting LoRaWAN gateways and devices to start collecting and visualizing IoT data. Remember to properly secure your installation and keep all components updated.