

Demonstration of ST25DV64KC LoRa® provisioning

Introduction

This demonstration is a joint development between STMicroelectronics and the ISCA Laboratory, Hellenic Mediterranean University (refer to [1] for more information).

LoRa® is a long-range, low data rate, and lower-power wireless communication system used in industrial IoT applications including, for example, smart metering, sensor monitoring, alarm, etc.

Each device requires provisioning with cryptographic keys and also registration on a LoRaWAN® network.

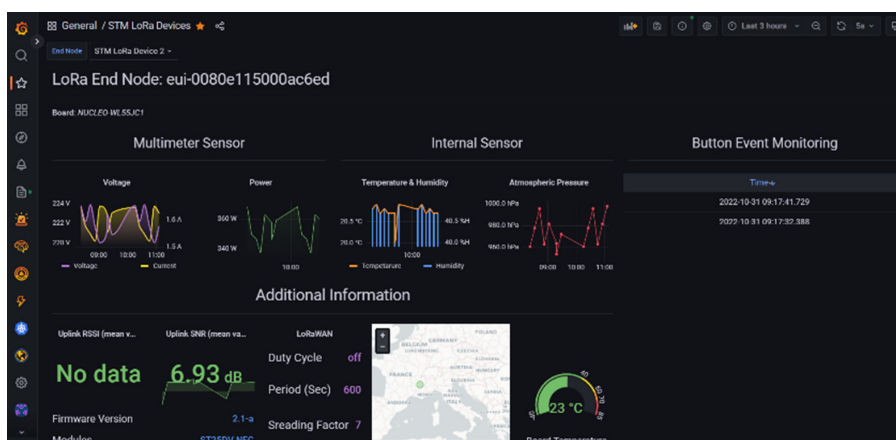
The current demonstration shows how a **ST25DV64KC** NFC tag is used to:

- Facilitate the provisioning of a LoRa® device with keys
- Register the device on a LoRaWAN® network

This demonstration uses The Things Network LoRaWAN® infrastructure.

When the registration is complete, it is possible to subscribe to receive MQTT notifications containing the data sent by the end node. This notification can be displayed on a PC (dashboard in a web browser) or on a smartphone with the Android™ application ST25 MQTT client.

Figure 1. Example of LoRa® PC dashboard display



The following packages are available on <http://www.st.com> for this demonstration:

- STSW-ST25DV010 STM32WL55 firmware
- STSW-ST25010 Android™ application
- STSW-ST25011 ST25 MQTT client for ST25DV64KC LoRa® provisioning demonstration

1 General information

For additional information, refer to the following websites.

Table 1. References

Reference	URL
[1]	ISCA Laboratory web site: https://isca.hmu.gr/
[2]	https://developer.android.com/studio
[3]	https://console.cloud.thethings.network
[4]	https://play.google.com/store/apps/details?id=com.st.st25loraprovisioning
[5]	https://developer.android.com/training/app-links
[6]	https://eu1.cloud.thethings.network/oauth/authorize
[7]	https://www.thethingsnetwork.org/docs/lorawan/frequencies-by-country
[8]	www.influxdata.com/time-series-platform/telegraf/
[9]	www.influxdata.com
[10]	www.grafana.com
[11]	https://jsonformatter.curiousconcept.com/
[12]	https://play.google.com/store/apps/details?id=com.st.st25mqttclient
[13]	https://www.thethingsindustries.com/docs/integrations/mqtt

Note: *The information provided is from third-party URL addresses, active at the time of document publication. However, STMicroelectronics shall not be liable for any change, move, or inactivation of the URL or the referenced material mentioned above.*

2 Glossary

Table 2. Acronyms and abbreviations

Acronyms	Definition
AppEUI	In LoRaWAN® specifications before 1.1, JoinEUI was called AppEUI. See JoinEUI.
AppKey	A device specific encryption key used during OTAA to derive both the Network Session Key and Application Session Key.
Application Session Key (AppSKey)	Key is used to encrypt the messages to ensure confidentiality.
Device Address (DevAddr)	A 32 bit non-unique identifier, assigned by the network server during device activation
Device EUI (DevEUI)	A 64 bit unique identifier for your device, set by the manufacturer.
End Device (Device) (Node)	A device with LoRaWAN® low power communication module
FTM Mailbox	ST25DV's RAM mailbox used to accelerate the communication speed during the host MCU and the NFC reader.
Gateway	Gateways form the bridge between end devices and network servers. Devices use low power networks like LoRa® to connect to the Gateway, while the Gateway uses high bandwidth networks like WiFi, Ethernet or cellular to connect to a network server.
Grafana	Online dashboard.
InfluxDB	InfluxDB is a high-speed read and write database used to store the LoRa® data.
IoT	Internet of Things
JoinEUI	The JoinEUI (formerly called AppEUI) is a 64 bit extended unique identifier used to identify the join server during activation.
Join Server	A join server is a component of the LoRaWAN® server defined in the LoRaWAN® back-end interfaces that is used to store root keys, generate session keys, and to send those securely to the network server and application server.
LoRa®	A modulation technology for long range wireless communication https://en.wikipedia.org/wiki/LoRa .
LoRaWAN®	LoRaWAN® is a media access control (MAC) protocol for wide area networks. It is designed to allow low-powered devices to communicate with internet connected applications over long range wireless connections.
MQTT	MQTT (Message Queuing Telemetry Transport) is a lightweight, publish-subscribe, machine to machine network protocol for message queuing service.
Network Session Key (NwkSKey)	Key used to ensure data integrity and authenticity.
Over-the-air Activation (OTAA)	This is the preferred and most secure way to connect a device to a LoRa® network. Devices perform a join-procedure with the network, during which a dynamic Device Address is assigned and security keys are negotiated with the device.
Telegraf	Open source server agent used to copy the LoRa® data from The Things Network to an InfluxDB database.
The Things Network (TTN)	The Things Network is a global collaborative Internet of Things ecosystem that creates networks, devices, and solutions using LoRaWAN®

3 Hardware requirements

The following hardware is needed to run this demonstration:

- A NUCLEO-WL55JC board. There are two versions of this board. Depending on your location, you need to choose the board suitable for the LoRaWAN® frequencies used in your region and configure the firmware accordingly (see [Section 7.1 Firmware personalization](#)):
 - NUCLEO-WL55JC1: high-frequency band, tuned for frequency between 865 MHz and 930 MHz. This is the board used in Europe, North America, India, and Australia.
 - NUCLEO-WL55JC2: low-frequency band, tuned for frequency between 470 MHz and 520 MHz. This is the board used in China.

Figure 2. NUCLEO-WL55JC2



- X-NUCLEO-NFC07A1 Nucleo shield. This Nucleo shield contains a ST25DV64KC NFC tag. This dynamic NFC tag is connected to the STM32WL55 MCU to provide it NFC bidirectional connectivity.

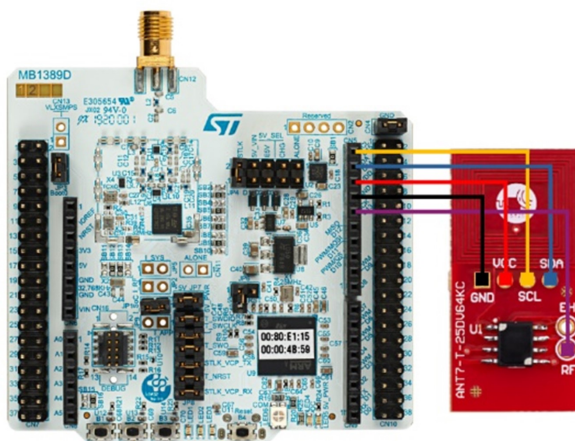
Figure 3. X-NUCLEO-NFC07A1 Nucleo shield



The ST1 jumper must be placed on the 3.3 V side.

Note: Alternatively, it is also possible to use a board ANT7-T-25DV64KC that has a smaller foot print and NFC antenna, but some soldering is needed to connect it to the Nucleo_WL55 board.

Figure 4. Nucleo_WL55 board



Use either the board X-NUCLEO-NFC07A1 or the board ANT7-T-25DV64KC, both are not necessary.

Table 3. Ant7 pin

Name	Ant7 pin	Arduino CN5 pin (name)
I2C clock	SCL	10 (SCL/D15)
I2C data	SDA	9 (SDA/D14)
Interrupt	RF	6 (SCK/D13)
Voltage	VCC	8 (AVDD)
Ground	GND	7 (GND)

- An Android smartphone (with Android 6.0 or later) with internet connection and NFC capability.
- If there is no public LoRaWAN® Gateway registered to The Things Network community edition close to your location, it may be necessary to use your own LoRa® gateway with an internet connection. Choose a LoRaWAN® gateway adapted to the frequencies used.

Figure 5. Example of LoRaWAN® gateway provided by The Things Industry



4 Software requirements

In order to use this demonstration, the following software is required:

- [STM32CubeIDE](#)
- [STM32CubeProgrammer](#)
- Optional: To rebuild the Android application, refer to the Android Studio website ([\[2\]](#)).

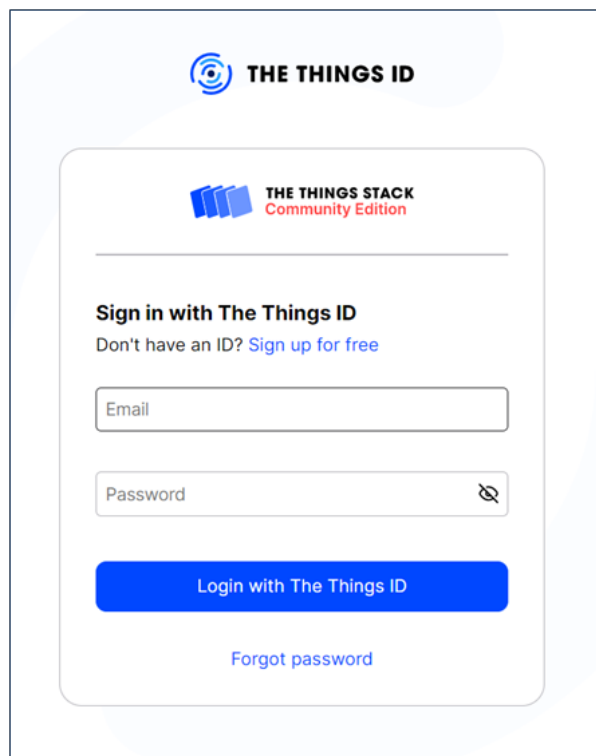
5 Prerequisites

The current demo relies on a LoRaWAN® infrastructure provided by The Things Network. This chapter explains how to create an account and create an application to The Things Network, which is necessary before using it. This registration is free and does not require a subscription.

5.1 Creation of a user account on The Things Network console

- Connect to The Things network (refer to [3]) and select a region close to your location.
- Click the **Sign up for free** button and follow the instructions.

Figure 6. The Things Network - Sign in

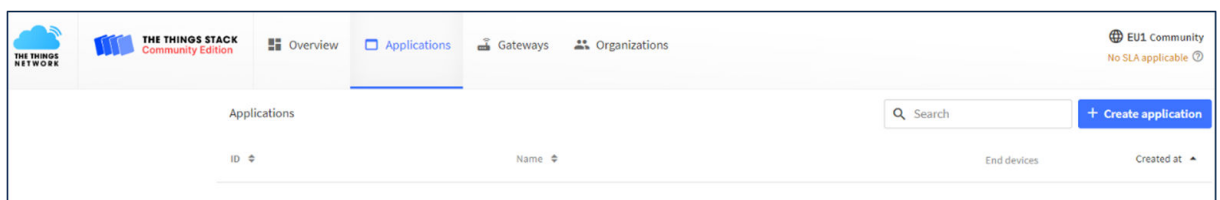


The screenshot shows the 'THE THINGS ID' sign-in page. At the top is the 'THE THINGS ID' logo. Below it is a box for 'THE THINGS STACK Community Edition'. The main heading is 'Sign in with The Things ID'. Below this is a link: 'Don't have an ID? [Sign up for free](#)'. There are two input fields: 'Email' and 'Password'. Below the password field is a blue button labeled 'Login with The Things ID'. At the bottom is a link: '[Forgot password](#)'.

5.2 Creation of an application on The Things Network

Once logged in, create an application on The Things Network. Click the Applications tab and then the **Create application** button.

Figure 7. The Things Network - Create application.



The screenshot shows the 'Applications' page in the The Things Network console. The top navigation bar includes 'Overview', 'Applications' (selected), 'Gateways', and 'Organizations'. On the right, it says 'EU1 Community' and 'No SLA applicable'. Below the navigation bar, there is a search bar and a '+ Create application' button. The main area shows a table with columns: 'ID', 'Name', 'End devices', and 'Created at'.

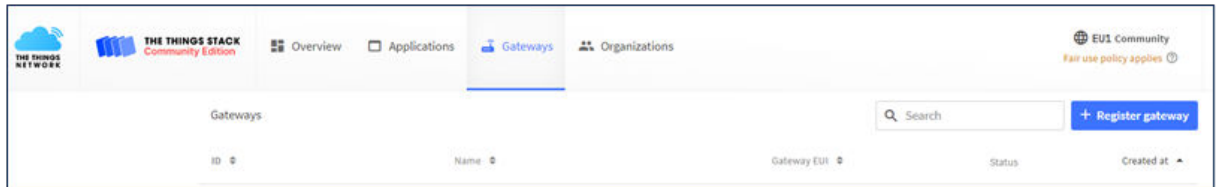
In this demonstration, an application with the name *ST25 LoRA Demo* is used with the id *st25lorademo*. This demonstration shows how to add a LoRa® device to this application and how to see the data coming from this device.

Note: A LoRa® device can belong to only one application at a time.

5.3 Registration of a gateway on The Things Network

If using your own LoRa® gateway, register it on The Things Network by clicking the Gateways tab and then the **Register gateway** button:

Figure 8. The Things Network – Register gateway



6 Android application installation

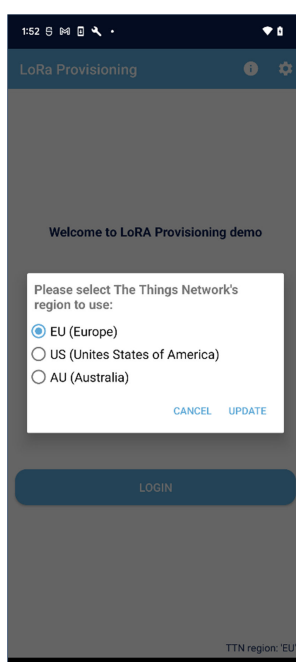
Install the Android application from Google Play (refer to [4]).

Android APK and source code can be downloaded from [STSW-ST25010](#). To open and rebuild the source code, use Android Studio (refer to [2]).

6.1 Personalization of the Android application

For the Android application, no change is needed in the source code. The personalization is done at runtime by clicking the gear in the top right corner of the home page, then selecting The Things Network's region. There is a choice between Europe, US, and Australia. Select the region closest to your location:

Figure 9. The Things Network – Region selection



The selected The Things Network region is then indicated at the bottom right of the main page.

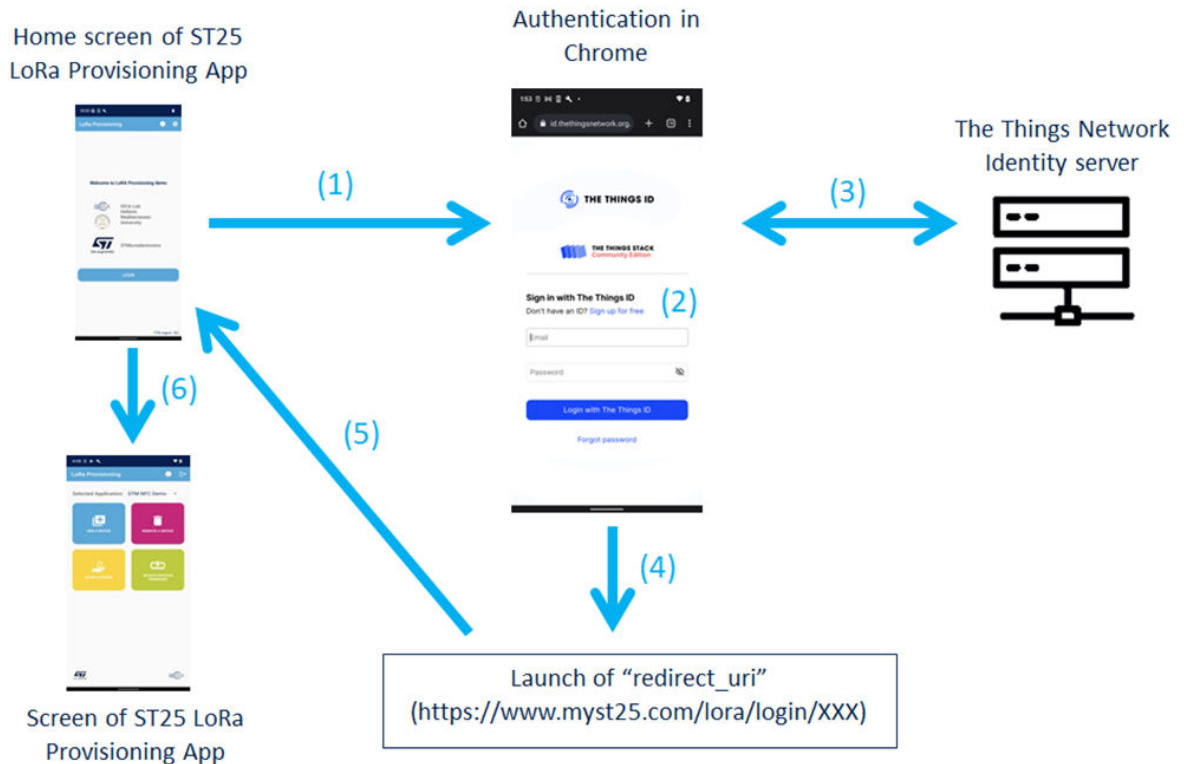
The user identification is always done through The Things Network servers located in Europe, independently of the region selected for using the application. The Android application automatically manages the user identification.

6.2 Authentication in the Android application

To authenticate the account on The Things Network servers (with the login and password created for The Things Network user account), use the Login button of the Android application. This authentication is based on [OAuth 2.0 protocol](#) and Android App links (refer to [5]).

Clicking the Login button launches the Android application URL to start a login session on The Things Network server. This URL opens in the default web browser used by the smartphone. The use of Chrome is recommended for this operation.

Figure 10. OAuth 2.0 authentication steps



1. Click the Login button. The Android application opens a URL to start a login session on The Things Network server.

Note:

Example URL:

`https://eu1.cloud.thethings.network/oauth/authorize?client_id=st25-lora-provisioning&redirect_uri=https://www.myst25.com/lora/login/&state=0FEF55D9FD8AE3F492EED15E4B498F71&response_type=code`

For The Things Network URL for user authentication, refer to [6].

The following parameters are passed to this URL:

- A `client_id`: (st25-lora-provisioning)
 - A `redirect_uri`: (`https://www.myst25.com/lora/login/`). This URI is called if the login, done in the web browser, is successful.
 - A state value: This is a random value changed every time.
 - A `response_type`: Signifies to the authorization server that the application is initiating the authorization code flow.
2. Enter the login and password (with the login and password created for The Things Network user account).
 3. The browser communicates with The Things Network identity server to check the user identity.
 4. If the authentication is successful, the "redirect_uri" passed at step 1 is called. This URL starts with `https://www.myst25.com/lora/login/`. If the authentication is not successful, an error message is displayed in Chrome.
 5. A mechanism called Android App links (see [5]) is used: When the `redirect_uri` (starting with `https://www.myst25.com/lora/login`) is called, the ST25 LoRa provisioning demo is automatically called. The AppLink configuration is done in the manifest file of the Android application. A notification indicating a successful login session is displayed when the Android application is viewed in the foreground. In the parameter of the URL used, the application finds a JSON message containing an `access_token`. The `access_token` is an ApiKey that is used to communicate with The Things Network server during this session. It is used to authenticate the user and to grant the access to the features allowed to the user.
 6. The main screen of the LoRa[®] application is launched. Once the user is authenticated, the user has access to The Things Network server to add or remove a LoRa[®] device. It is done from HTTPS requests using the `access_token` as an API key.

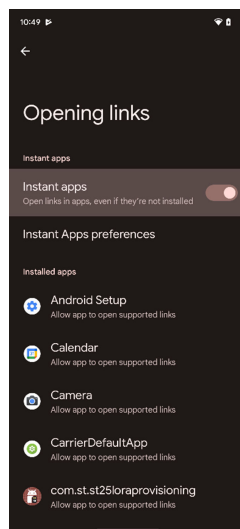
6.3 Troubleshooting

Depending on the smartphone and the browser used, the Android application link may not work as expected. For example, the `redirect_uri` (<https://www.myst25.com/lora/login/>) may open in the browser without opening the Android application.

The following example demonstrates how to resolve the problem on a Google Pixel 6:

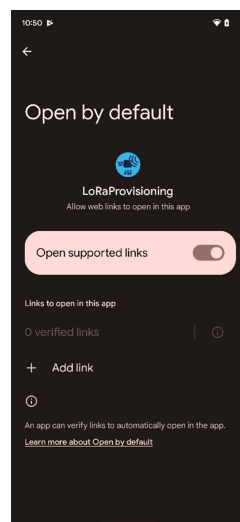
1. Go to the **Settings** menu and search for **Opening links**.

Figure 11. Opening links



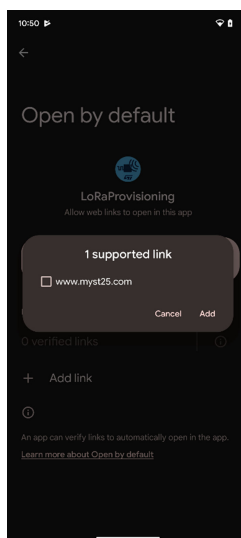
2. Click `com.st25loraprovisioning` to see the links associated to this application:

Figure 12. Viewing associated links



3. Click the **Add link** button and tick the check box corresponding to www.myst25.com:

Figure 13. Adding links



The links pointing to www.myst25.com open in the LoRa® provisioning application. However, each smartphone has its own organization for the Settings menu, so the wording and the location of the settings may be different.

7 STM32WL firmware installation

Download the STM32WL firmware from [STSW-ST25DV010](#).

This package contains the bootloader (source code and binary) and the firmware (source code and binaries for the European region).

- For the bootloader, it is not necessary to recompile it, use directly as provided.
- For the STM32WL firmware: For the European region, the binaries are provided in the STSW-ST25DV010 package and can be used directly. For other regions, the firmware must be personalized and recompiled.

7.1 Firmware personalization

The current demonstration is by default configured for:

- European region (868 MHz frequency plan)
- ant7-t-25dv64kc board

Before compiling the firmware, customize the frequency plan and the hardware board used.

7.1.1 Change of frequency plan

Consult The Things Network web page for the frequency plans per region (see [7]).

The table below lists commonly used frequency plans.

Table 4. Example frequency plans

Frequency plan	Base frequency	Description	Nucleo board requirement
EU_863_870	868	Default frequency plan for Europe	NUCLEO-WL55JC1
US_902_928_FSB_2	915	TTN Community Network frequency plan for the United States and Canada, using sub-band 2	NUCLEO-WL55JC1
AU_915_928_FSB_2	915	TTN Community Network frequency plan for Australia, using sub-band 2	NUCLEO-WL55JC1
CN_470_510_FSB_11	470	TTN Community Network frequency plan for China, using sub-band 11	NUCLEO-WL55JC2

1. Click the file `sw\WL55JC1\ .project` to open the project in STM32CubeIDE.
2. Open the file `lorawan_conf.h`.
3. To change the region, comment the line containing `REGION_EU868` and uncomment another region:

Figure 14. Region selection

```

89 /* Region -----*/
90 /* the region listed here will be linked in the MW code */
91 /* the application (on sys_conf.h) shall just configure one region at the time */
92 /*#define REGION_AS923*/
93 /*#define REGION_AU915*/
94 /*#define REGION_CN470*/
95 /*#define REGION_CN779*/
96 /*#define REGION_EU433*/
97 #define REGION_EU868
98 /*#define REGION_KR920*/
99 /*#define REGION_IN865*/
100 /*#define REGION_US915*/
101 /*#define REGION_RU864*/

```

4. Open the file `lora_app.h`.

5. Change define of ACTIVE_REGION:

Figure 15. Define the ACTIVE_REGION

```

39  /* Exported constants ----- */
40
41  /* LoraWAN application configuration (Mw is configured by lorawan_conf.h)
42     Select one of the regions defined in the enum LoRaMacRegion_t
43     (LORAMAC_REGION_EU868, LORAMAC_REGION_CN470, LORAMAC_REGION_US915...) */
44  #define ACTIVE_REGION                LORAMAC_REGION_EU868
45

```

The possible values are listed in the enum `LoRaMacRegion_t` of the file `Middlewares/Third_Party/LoRaWAN/Mac/LoRaMacInterfaces.h`:

```

/*!
 * LoRaMAC region enumeration
 */
typedef enum eLoRaMacRegion
{
    /*!
     * AS band on 923MHz
     */
    LORAMAC_REGION_AS923,
    /*!
     * Australian band on 915MHz
     */
    LORAMAC_REGION_AU915,
    /*!
     * Chinese band on 470MHz
     */
    LORAMAC_REGION_CN470,
    /*!
     * Chinese band on 779MHz
     */
    LORAMAC_REGION_CN779,
    /*!
     * European band on 433MHz
     */
    LORAMAC_REGION_EU433,
    /*!
     * European band on 868MHz
     */
    LORAMAC_REGION_EU868,
    /*!
     * South korean band on 920MHz
     */
    LORAMAC_REGION_KR920,
    /*!
     * India band on 865MHz
     */
    LORAMAC_REGION_IN865,
    /*!
     * North american band on 915MHz
     */
    LORAMAC_REGION_US915,
    /*!
     * Russia band on 864MHz
     */
    LORAMAC_REGION_RU864,
} LoRaMacRegion_t;

```

7.1.2 Hardware board selection

Configure the firmware to correspond to the hardware used.
 This is done by opening the file `isys_configuration.h`.
 Edit the line containing the `#define NFC_TYPE`:

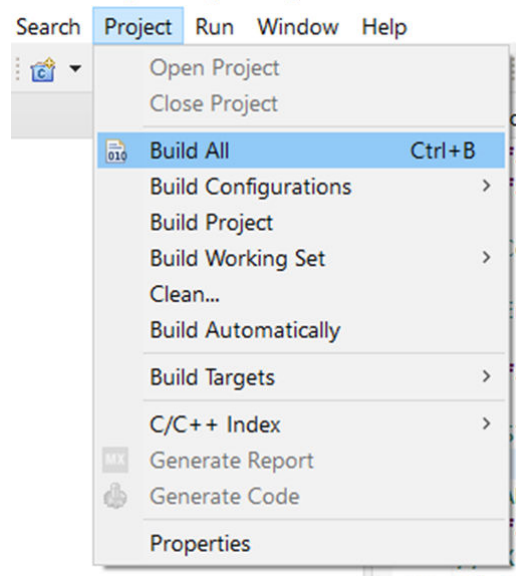
Figure 16. #define NFC_TYPE

```
73
74 /* Select ST25DV64KC PCB type {NFC_ANT7, NFC_NUCLEO_SHIELD}
75 */
76 #define NFC_TYPE NFC_ANT7
```

7.2 Build of firmware binaries

The firmware is built with the STM32CubeIDE using the **Build All** entry in the project menu.

Figure 17. Build firmware binaries



7.3 Flashing

Use STM32CubeProgrammer to flash the bootloader and the firmware:

- Bootloader must be flashed to the address 0x08000000
- Firmware must be flashed to the address 0x08005800

Some binaries are available in the STSW-ST25DV010 package in \binaries directory. It contains:

- the bootloader binary
- the firmware for the board ANT7-T-25DV64KC and European region (EU868)
- the firmware for the board X-NUCLEO-NFC07A1 and European region (EU868)
- binaries for Firmware Upgrade demo (one binary for the board with ANT7-T-25DV64KC and one for the board with X-NUCLEO-NFC07A1)

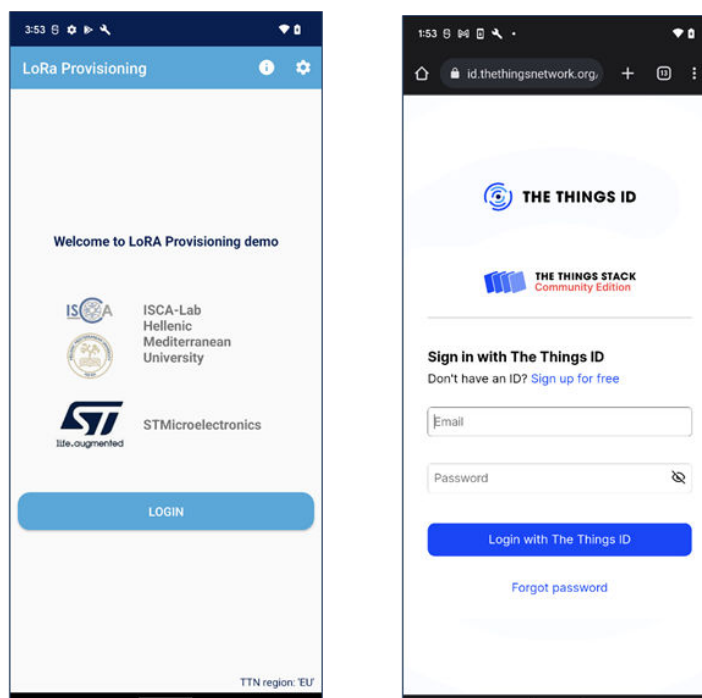
Other declinations of firmware (according to board and region) must be recompiled as shown in [Section 7.1 Firmware personalization](#).

8 How to run the demonstration

8.1 Login on The Things Network

Launch the Android application and click the **Login** button. This opens a web browser to perform an OAuth authentication. The page requires you to provide to log in and password of your account on The Things Network:

Figure 18. OAuth authentication login and sign in

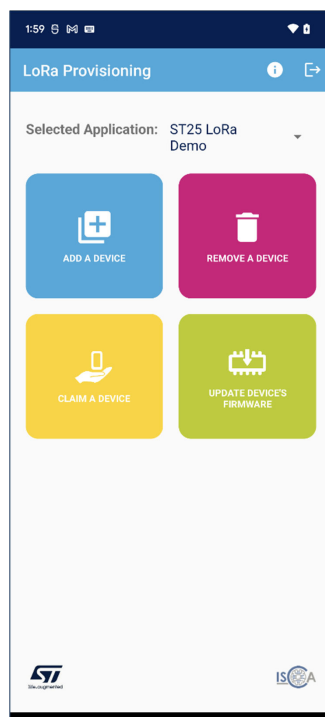


DT72447

Note: *It is recommended to use Chrome for the authentication phase.*

Once the login is complete, the Android application opens and the main screen is available where it is possible to add or remove a LoRa® device.

Figure 19. LoRa® main screen

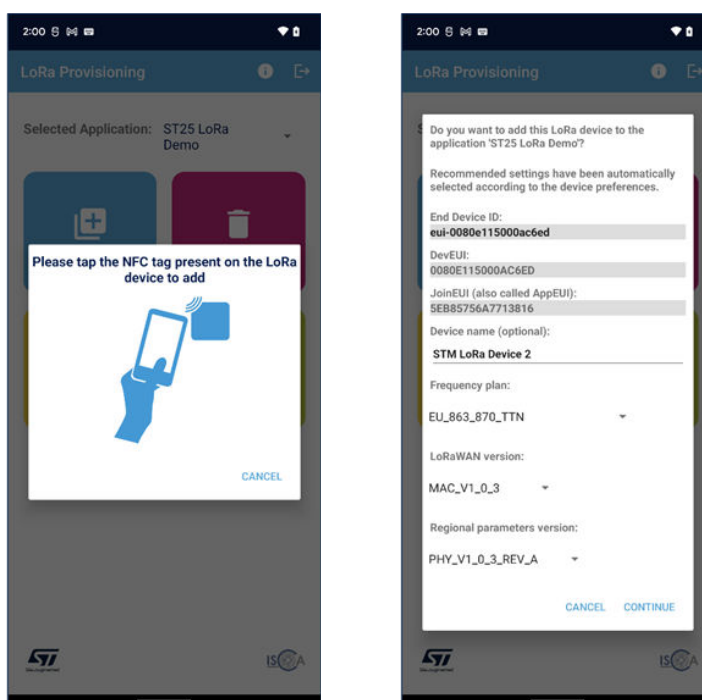


At the top of the screen, select the application with the name chosen in [Section 5.2 Creation of an application on The Things Network](#). In this case, the application is named ST25 LoRa Demo.

8.2 Add a LoRa® device to The Things Network

Click the **Add a device** button and follow the instructions requesting to tap the NFC tag present on the LoRa® device.

Figure 20. Add a LoRa® device



DT72448

An NFC communication takes place between the smartphone and the LoRa® device. This NFC communication uses the FTM mailbox of the ST25DV64KC to collect some information about the device.

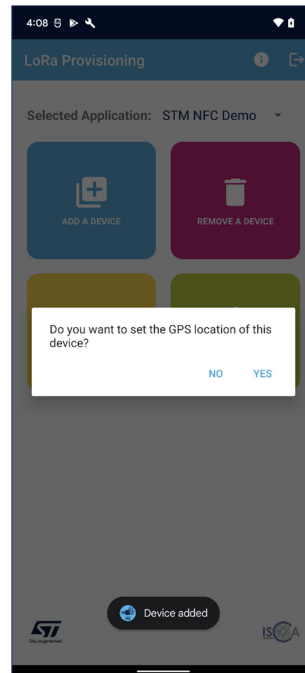
A popup opens to confirm adding this device to the application on The Things Network. The popup displays some information about the device, for example, its unique DevEUI. Here, provide a name to this device to facilitate its recognition.

By default, the fields about the frequency plan, the LoRaWAN® version, and the regional parameters are filled with the values recommended by the device. It is possible to modify them, but if the firmware has been properly configured for the selected region (see [Section 7.1 Firmware personalization](#)), this should not be necessary.

Click continue. If the device add operation is successful, a **Device added** message is displayed at the bottom of the screen. During this phase, the device is provisioned with some cryptographic keys that are used to obtain access to a LoRa® gateway during an *Over The Air Activation*. It is important to keep the smartphone on the device during the phases showing a **Please wait** message.

The application requests setting the GPS location of the device.

Figure 21. Set the device GPS location



The GPS location is saved in the information associated to this device on The Things Network. LoRa® devices are frequently used in isolated places, therefore knowing the exact GPS location is practical and may help to locate the device as necessary.

A few seconds after the provisioning, the LoRa® device reboots and tries to contact an accessible LoRa® gateway to perform an Over The Air Activation (OTAA). While the device is not connected, LED3 of X-NUCLEO-NFC07A1 blinks orange.

If this activation is successful, LED2 of the Nucleo shield turns blue during 1 second and the LED3 is switched OFF. The device is then connected to The Things Network.

If LED3 keeps blinking, it means that the activation failed.

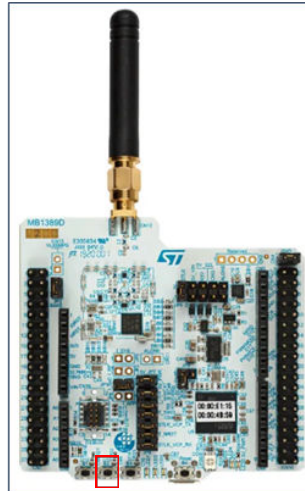
After a reset, the activation usually takes about 12 seconds.

8.3 Getting data from theLoRa® device

Once connected to The Things Network, the LoRa® device regularly sends data. Every 10 minutes the device sends some fictive data (voltage, power, temperature, etc.) to The Things Network servers.

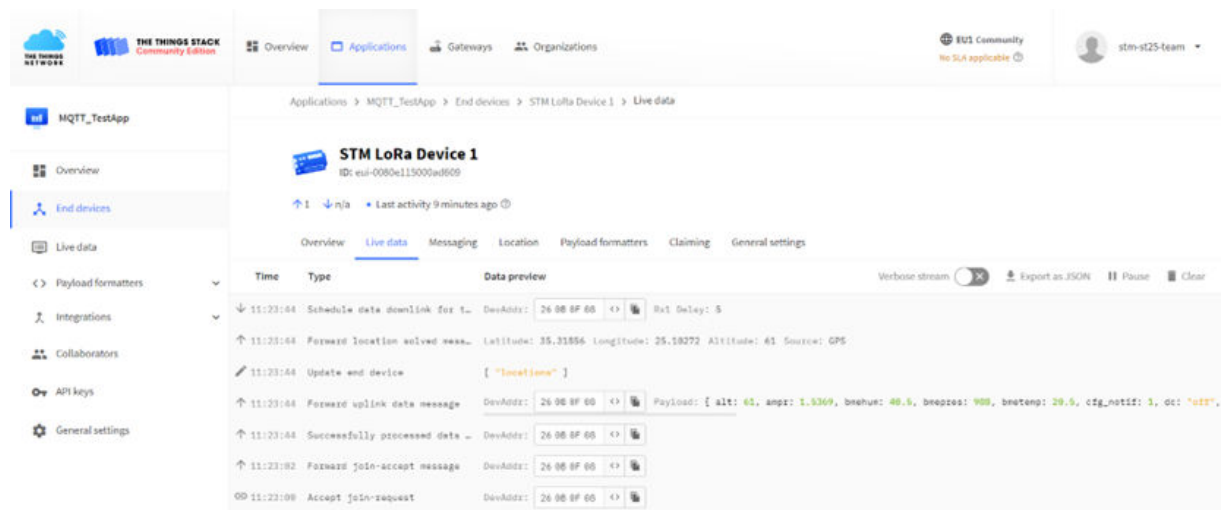
On the Nucleo board, pressing the second button (SW2), triggers a data transmission (user event).

Figure 22. User button SW2 on Nucleo board



On The Things Network console, click the application and then select the end device. Use the **Live Data** tab to see all the data (join data, uplink, and downlink data) exchanged with this device. The uplink data message contains the data sent by the device ("alt", "ampr", "bmehum", etc.).

Figure 23. Selection of the end device

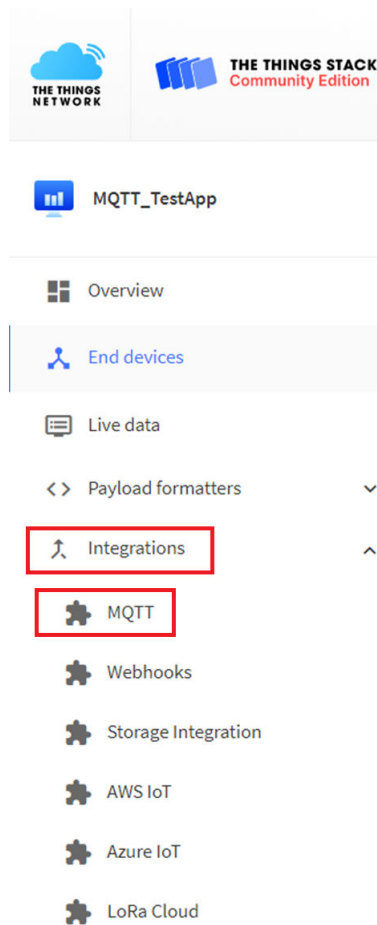


Each time the SW2 button is pressed, new live data appears. It indicates that the data is properly received by TTN's LoRaWAN® server.

8.4 Display of LoRa® data

The data collected on The Things Network's LoRaWAN® server is used in multiple ways. In the left pane of the TTN console, click Integrations to see all the possibilities to use the data coming from the TTN servers.

Figure 24. LoRa® dashboard



In this demonstration, the data collected by TTN's LoRaWAN® server is exported through MQTT and displayed on a smartphone dashboard or in a web browser.

TTN server acts as an MQTT broker sending notifications every time LoRa® data is available.

It is possible for MQTT clients to subscribe to receive the notifications. Each MQTT notification is independent, allowing multiple clients to subscribe to the same notifications. On the TTN console, go to the MQTT tab to see the URL and the credentials to receive the MQTT notifications.

Figure 25. MQTT URL and credentials

MQTT

MQTT is a publish/subscribe messaging protocol designed for IoT. Every application on TTS automatically exposes an MQTT endpoint. In order to connect to the MQTT server you need to create a new API key, which will function as connection password. You can also use an existing API key, as long as it has the necessary rights granted.

Further resources

[MQTT server](#) | [Official MQTT website](#)

Connection information

MQTT server host

Public address

Public TLS address

Connection credentials

Username

Password [Go to API keys](#)

When using the MQTT for the first time, click the Generate new API key button.

Note: Copy the key and save it in a secure place because it is not possible to access it again.
 The API key is used as a password when configuring an MQTT client to receive MQTT notifications.

8.4.1 Example of uplink data

The following is an example of uplink data notified by MQTT (in JSON format):

```
{
  "end_device_ids": {
    "device_id": "eui-0080e115000ac6ed",
    "application_ids": {
      "application_id": "my-mqtt-test-app",
      "dev_eui": "0080E115000AC6ED",
      "join_eui": "5EB85756A7713816",
      "dev_addr": "260B8E6D"
    },
    "correlation_ids": [
      "as:up:01GRB7X7HPZCEPR45MQ74901PY",
      "gs:conn:01GRB3WA7DGB1T87VR2RDK07Z2",
      "gs:up:host:01GRB3WA7PHRMD5CN1PJCGK2NW",
      "gs:uplink:01GRB7X7B6RCQA8A07C3BV75JB",
      "ns:uplink:01GRB7X7B7N1XZGW2ACAVHJDW6",
      "rpc:/ttn.lorawan.v3.GsNs/HandleUplink:01GRB7X7B6RKZEY6TDB7ZM1TX",
      "rpc:/ttn.lorawan.v3.NsAs/HandleUplink:01GRB7X7HN9WA1524QXE01SGKC"
    ],
    "received_at": "2023-02-03T08:57:22.229736325Z",
    "uplink_message": {
      "session_key_id": "AYYWFV3anqaW2UbmYuF/hg==",
      "f_port": 2,
      "f_cnt": 2,
      "frm_payload": "BAAAVtkAADyZAACGuhUH5AGSJwY=",
      "decoded_payload": {
        "ampr": 1.5513,
        "bmehum": 40.2,
        "bmepres": 999,
        "bmetemp": 20.2,
        "cfg_notif": 0,
        "event_type": 1,
        "temp": 21,
        "volt": 222.33,
        "watt": 344.9
      },
      "rx_metadata": [
        {
          "gateway_ids": {
            "gateway_id": "eui-58a0cbffffe8049c2",
            "eui": "58A0CBFFFE8049C2"
          },
          "time": "2023-02-03T08:57:21.867705106Z",
          "timestamp": 4222790348,
          "rssi": -33,
          "channel_rssi": -33,
          "snr": 7.5,
          "location": {
            "latitude": 45.15531774774593,
            "longitude": 5.760384968059596,
            "source": "SOURCE_REGISTRY"
          },
          "uplink_token": "CiIKIAoUZXXVpLTU4YTBjYmZmZmU4MDQ5YzISCfigy//+gEnCEMzVyt0PGgsI8pjzngYQ//S+CiDg2ayP83o=",
          "received_at": "2023-02-03T08:57:21.868256649Z"
        }
      ],
      "settings": {
        "data_rate": {
          "lorawan": {
            "bandwidth": 125000,
            "spreading_factor": 7,
            "coding_rate": "4/5"
          }
        },
        "frequency": "867500000",
        "timestamp": 4222790348,
        "time": "2023-02-03T08:57:21.867705106Z",
        "received_at": "2023-02-03T08:57:22.023110775Z",
        "consumed_airtime": "0.082176s",
        "locations": {
          "frm_payload": {
            "latitude": 35.31856,
            "longitude": 25.10272,
            "altitude": 61,
            "source": "SOURCE_GPS",
            "user": {
              "latitude": 45.1554092,
              "longitude": 5.7604965,
              "altitude": 283,
              "source": "SOURCE_REGISTRY"
            }
          },
          "net_work_ids": {
            "net_id": "000013",
            "tenant_id": "ttn",
            "cluster_id": "eu1",
            "cluster_address": "eu1.cloud.thethings.network"
          }
        }
      }
    }
  }
}
```

Reading JSON data is difficult because the data is not indented. It is possible to format the data with an online formatter (for an example, see [11]). The structure is easier to read, as shown below.

Figure 26. Data reformatted from JSON format

```
{
  "end_device_ids": {
    "device_id": "eui-0080e115000ac6ed",
    "application_ids": {
      "application_id": "my-mqtt-test-app"
    },
    "dev_eui": "0080E115000AC6ED",
    "join_eui": "5EB85756A7713816",
    "dev_addr": "260B8E6D"
  },
  "correlation_ids": [
    "as:up:01GRB7X7HPZCEPR45MQ74901PY",
    "gs:conn:01GRB3WA7DG81T87VR2RDK07Z2",
    "gs:up:host:01GRB3WA7PHRMD5CN1PJCGK2NW",
    "gs:uplink:01GRB7X7B6RCQA8A07C3BV75JB",
    "ns:uplink:01GRB7X7B7N1XZGW2ACAVHJDW6",
    "rpc:/ttn.lorawan.v3.GsNs/HandleUplink:01GRB7X7B6RKZEY6TDB7ZM1TXX",
    "rpc:/ttn.lorawan.v3.NsAs/HandleUplink:01GRB7X7HN9WA1524QXE01SGKC"
  ],
  "received_at": "2023-02-03T08:57:22.229736325Z",
  "uplink_message": {
    "session_key_id": "AYYWfV3anqaW2UbmYuF/hg==",
    "f_port": 2,
    "f_cnt": 2,
    "frm_payload": "BAAAVtkAADyZAACGuhUH5AGSJwY=",
    "decoded_payload": {
      "ampr": 1.5513,
      "bmehum": 40.2,
      "bmepres": 999,
      "bmetemp": 20.2,
      "cfg_notif": 0,
      "event_type": 1,
      "temp": 21,
      "volt": 222.33,
      "watt": 344.9
    },
    "rx_metadata": [
      {
        "gateway_ids": {
          "gateway_id": "eui-58a0cbfffe8049c2",
          "eui": "58A0CBFFFE8049C2"
        }
      }
    ]
  }
}
```

The data sent by the LoRa® device is shown in the `decoded_payload` field (including current, humidity, pressure, temperature, voltage, power, etc.).

In this demonstration, two ways of displaying data coming from the LoRa® device are shown:

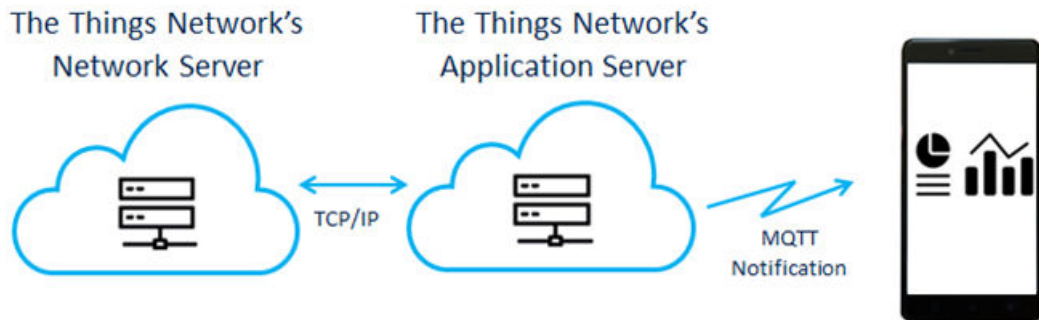
- Display on a smartphone
- Display on a web browser

8.4.2 Display on a smartphone

Install the ST25MQTTClient Android application from Google Play (refer to [12]).

This application serves as an MQTT client able to receive the MQTT notifications, parse the JSON data, and display certain data in a graph or table.

Figure 27. Data transfer to a smartphone



When opening the application, the MQTT credentials are provided. Shown below is a mapping between TTN credentials and the Android application:

Figure 28. MQTT credentials mapping



Once connected to the MQTT server, specify an MQTT topic.

The Things Network documentation (refer to [13]) indicates all the MQTT topics that are used to subscribe to receive certain TTN data. The case presented uses the topic containing the uplink data (referring to data going from the LoRa® device to the TTN LoRaWAN® server).

`v3/{application id}/{tenant id}/devices/{device id}/up`

The following is an example of a topic for the application st25lorademo and the device eui-0080e115000ac6ed:

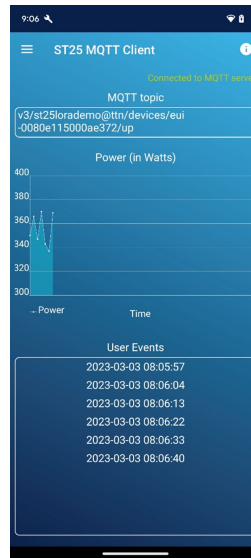
`v3/st25lorademo@ttn/devices/eui-0080e115000ac6ed/up`

The string starting with eui-XXXXXXXXXX is a unique ID that can be replaced by your device ID.

NFC can be used to facilitate the setup of the topic. It can be edited manually or tap the NFC tag present on the LoRa® device. The topic is automatically filled with the device EUI.

Once the topic set, the Android application receives MQTT notifications every time the LoRa® device sends data. The application parses the JSON data and displays it.

Figure 29. Example MQTT notification display on a smartphone



For example, the power data displayed in the graph, is sent by the LoRa® device every 10 minutes. The user events are sent when the SW2 button of the Nucleo board is pressed.

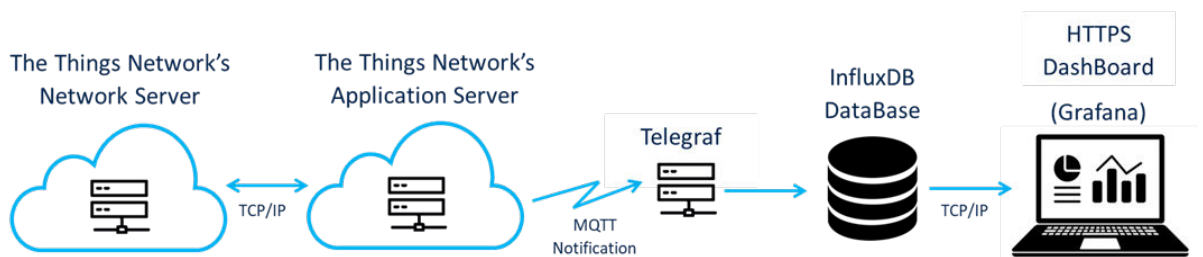
Note: It is necessary to wait for a minimum of 15 seconds between consecutive user events.

8.4.3 Display in a web browser

It is possible to display MQTT data in a web browser. In this demonstration, a server based agent, Telegraf (refer to [8]), is used to subscribe to receive the MQTT data. The data is then copied to the InfluxDB (cloud) database (refer to [9]).

A Grafana dashboard (refer to [10]) is used to display the data present in the InfluxDB database.

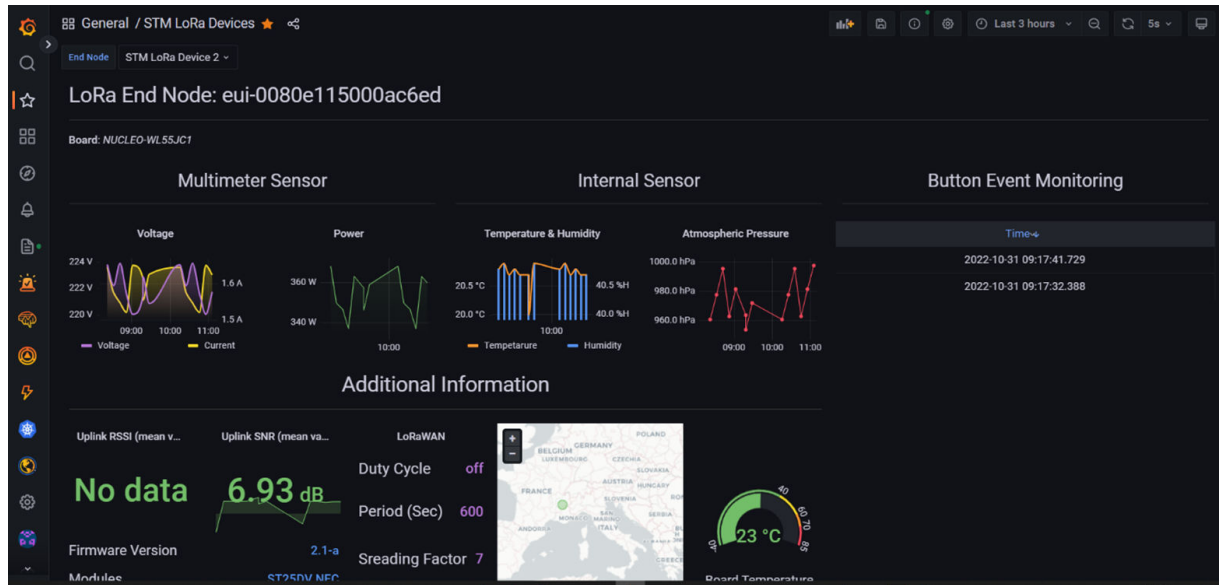
Figure 30. MQTT data transfer



The configuration of Telegraf, InfluxDB, and Grafana is beyond the scope of this demonstration and is not covered in the current document.

The information displayed on the Grafana dashboard is shown below.

Figure 31. Grafana dashboard display



Some graphs show the voltage, power, temperature, and pressure received from the LoRa® device every 10 minutes. It shows how some measurements are collected periodically from a low-power device.

On the right side of the figure, a table shows the button events raised when the SW2 button of the Nucleo board is pressed. It also shows how it is possible for the device to send an alarm at any time.

8.5 Remove a LoRa® device from The Things Network

It is possible to unregister a device from The Things Network by clicking the Remove button.

A device can be attached to only one application from The Things Network, so this option is useful to remove a device from an application and add it to another one.

When clicking this button, a message is received requesting to tap the NFC tag of the device again. The smartphone uses this to get the unique DevEUI of the device and then remove it from The Things Network.

8.6 Claiming a device

The Claiming feature is an alternative way of taking ownership of a device.

When using this feature, the device is provisioned with keys by the manufacturer directly at the factory. The device is also registered on the Join Server of a LoRa® network (for example The Things Network). In the device booklet, a QR code is added. The QR code to prove of the device is used by the device owner as proof of ownership. The ownership of the device is then transferred from the manufacturer to the owner. The Join Server referenced in the QR code (through the JoinEUI id) is used to propagate the keys corresponding to this device to the network server and application server. The device contacts a LoRa® gateway and starts its activation.

In the current demo, a claiming phase based on an NFC tag instead of a QR code is used. NFC tag content is identical to the QR code content and based on the TR005_LoRaWAN_Device_Identification_QR_Codes recommendation of the LoRa® alliance.

8.7 Firmware upgrade

The NFC tag present on the LoRa® device is used for the provisioning of the device registering it on a LoRa® network. This bidirectional interface is also used during the product life to update its firmware.

The ST25DV64KC contains a RAM mailbox of 256 bytes allowing the transfer of data both ways. The firmware of the STM32WL is updated this way. The data rate is about 40 seconds to transfer 100 Kbytes.

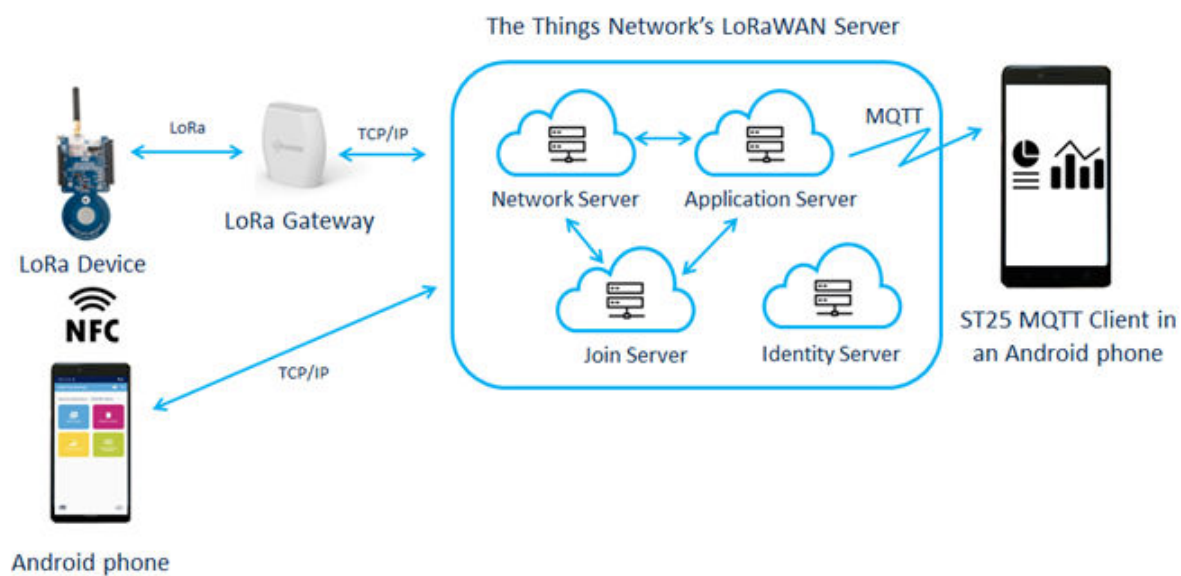
9 Comparison with existing provisioning solutions

Table 5. Comparison of provisioning solutions

Provisioning method	Comments
Provisioning with dynamic NFC tags	<p>This solution is presented in the current document. The bidirectional capability of the NFC tag is used to provision the device with keys.</p> <p>It is important to highlight that, with this solution, the key provisioning is not done at the factory but at runtime when the user or an installer sets up the product for the first time. As a result, it is not necessary to do any kind of personalization at the end of the production line.</p> <p>The NFC interface is used later on in product life to collect some data/logs, configure the product or update its firmware.</p>
Provisioning with QR code	<p>With this solution, the device is provisioned with keys at the factory. The manufacturer should set some keys in the device and store them as well on a join server in the Cloud. The manufacturer needs to power the device and use an interface to communicate with the product.</p> <p>A QR code is put on the product to identify the device and the join server to contact for the registration of this device.</p> <p>When the user installs the product, it is necessary to install a specific application on the smartphone used and to scan the QR code. The application contacts the join server and requests to propagate the keys to the network server and application server.</p>
Provisioning with Bluetooth® Low Energy	<p>This solution requires the implementation of a full Bluetooth® Low Energy stack.</p> <p>In this solution, as for the provisioning with NFC, the keys are set in the device at runtime when the user or installer sets up the product.</p>

Appendix A

Figure 32. Infrastructure of ST25DV64KC LoRa® provisioning demonstration



Revision history

Table 6. Document revision history

Date	Version	Changes
09-Mar-2023	1	Initial release.
12-Jun-2023	2	Updated: <ul style="list-style-type: none">• Section Introduction• Section 7.3 Flashing

Contents

1	General information	2
2	Glossary	3
3	Hardware requirements	4
4	Software requirements	6
5	Prerequisites	7
5.1	Creation of a user account on The Things Network console	7
5.2	Creation of an application on The Things Network	7
5.3	Registration of a gateway on The Things Network	8
6	Android application installation	9
6.1	Personalization of the Android application	9
6.2	Authentication in the Android application	9
6.3	Troubleshooting	11
7	STM32WL firmware installation	13
7.1	Firmware personalization	13
7.1.1	Change of frequency plan	13
7.1.2	Hardware board selection	14
7.2	Build of firmware binaries	15
7.3	Flashing	15
8	How to run the demonstration	16
8.1	Login on The Things Network	16
8.2	Add a LoRa® device to The Things Network	17
8.3	Getting data from the LoRa® device	19
8.4	Display of LoRa® data	20
8.4.1	Example of uplink data	22
8.4.2	Display on a smartphone	24
8.4.3	Display in a web browser	25
8.5	Remove a LoRa® device from The Things Network	26
8.6	Claiming a device	26
8.7	Firmware upgrade	26
9	Comparison with existing provisioning solutions	27
Appendix A		28
	Revision history	29

List of figures

Figure 1.	Example of LoRa® PC dashboard display	1
Figure 2.	NUCLEO-WL55JC2	4
Figure 3.	X-NUCLEO-NFC07A1 Nucleo shield	4
Figure 4.	Nucleo_WL55 board	5
Figure 5.	Example of LoRaWAN® gateway provided by The Things Industry	5
Figure 6.	The Things Network - Sign in	7
Figure 7.	The Things Network - Create application	7
Figure 8.	The Things Network - Register gateway	8
Figure 9.	The Things Network - Region selection	9
Figure 10.	OAuth 2.0 authentication steps	10
Figure 11.	Opening links	11
Figure 12.	Viewing associated links	11
Figure 13.	Adding links	12
Figure 14.	Region selection	13
Figure 15.	Define the ACTIVE_REGION	14
Figure 16.	#define NFC_TYPE	15
Figure 17.	Build firmware binaries	15
Figure 18.	OAuth authentication login and sign in	16
Figure 19.	LoRa® main screen	17
Figure 20.	Add a LoRa® device	18
Figure 21.	Set the device GPS location	19
Figure 22.	User button SW2 on Nucleo board	20
Figure 23.	Selection of the end device	20
Figure 24.	LoRa® dashboard	21
Figure 25.	MQTT URL and credentials	22
Figure 26.	Data reformatted from JSON format	23
Figure 27.	Data transfer to a smartphone	24
Figure 28.	MQTT credentials mapping	24
Figure 29.	Example MQTT notification display on a smartphone	25
Figure 30.	MQTT data transfer	25
Figure 31.	Grafana dashboard display	26
Figure 32.	Infrastructure of ST25DV64KC LoRa® provisioning demonstration	28

List of tables

Table 1.	References	2
Table 2.	Acronyms and abbreviations	3
Table 3.	Ant7 pin	5
Table 4.	Example frequency plans	13
Table 5.	Comparison of provisioning solutions	27
Table 6.	Document revision history	29

IMPORTANT NOTICE – READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgment.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2023 STMicroelectronics – All rights reserved