TEKTELIC COMMUNICATIONS INC.

TEKTELIC IOT NETWORK SERVER

USER'S GUIDE

Name:	User's Guide
Revision:	1.3
Issue Date:	23/05/2019
Status:	*DRAFT*

Document Revision

Revision	Issue Date	Author	Comments
1.0	Nov. 9, 2017	G. Lungu	User's Guide (for NS release 1.0)
1.1	Mar. 7. 2019	G. Lungu	Updated swagger "live-demo" broken link
1.2	April 30, 2019	J Peterson	Added more detailed information on Swagger UI
1.3	May 23, 2019	A.Panchal	Added information on GPS Position feature, Real-time packets tab, GW Logging

Table of Contents

1 Main entities overview

1.1 Device

Device - represents a sensor or other end-device (in terms of LoRaWAN). Device can be LoRaWAN A, B and C classes.

1.2 Gateway

Gateway - receives data from physical devices and forwards it to the network server. Gateway is always registered on the NS and belongs to only one Provider.

1.3 Application

Application - is a logical grouping of devices. Application may store general settings that impact provisioning of devices. Each device may belong to only one Application.

1.4 Credentials

Both Gateways and Applications use credentials to access NS APIs. These credentials are designed to be customizable. Current NS version supports basic credentials (username and password).

1.5 Data Converter

Data converter is a NS extension that allows to convert incoming binary payloads to JSON payloads for upstream messages and vise-versa for downstream messages.

1.6 Application Integration

Integrations allow to push upstream messages from devices that belong to particular application to external systems. Example of integrations are: HTTP integration to publish upstream messages to external systems via HTTP request, or ThingsBoard integration that communicates with ThingsBoard via MQTT. Each application Integration uses exactly one Data Converter.

1.7 Users

User is an entity that has credentials to access NS Web UI/ REST API. Several roles are supported: System Administrator/User, Provider Administrator/User, Customer Administrator/User, Sub-Customer Administrator/User. The difference between Administrator and a User is that Administrator is able to do reads/writes and deletes while User is read-only.

Single user can have roles only on one level: System, Provider, Customer, Sub-Customer. This means that single user is not able to be Provider Administrator and Sub-Customer read-only user.

1.8 Sub-customer

Sub-customer is able to

1) Manage own devices.
2) Create and manage applications.

Applications and devices are isolated and not visible to other sub-customers.

1.9 Customer

Customer is able to

1) Manage own devices.
2) Create and manage applications.
3) Create and manage sub-customers.
4) Assign Applications or individual devices to a Sub-customer

Applications and devices are isolated and not visible to other customers or sub-customers.

1.10 Provider

Provider is able to

1) Manage own gateways.
2) Manage own customers.
3) Read statistics and control own Gateways.

Gateways, customers and other data is isolated and not visible to other providers.

1.11 System Administrator

System Administrator is able to:

1) Manage providers
2) Read total statistics from Providers

2 Available APIs

2.1 REST API - Overview

NS configuration and management is available through REST APIs. This APIs are available only for logged in users.

NS REST API can be explored using Swagger UI. To explore these REST APIs of the server visit the Swagger UI link (live-demo).

When accessing the assigned NS server, the UI can be opened using the following URL:

http(s)://YOUR_HOST:PORT/swagger-ui.html

2.1.1 REST API authentication

NS uses JWT for request auth the “X-Authorization” header needs to be populated, using “Authorize” button in the top-right corner of the Swagger UI.

In order to get the JWT token, the following request needs to be executed in a Linux machine. Please make sure your Linux Machine has curl packages installed.

```
curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' -d '{"username":"$username", "password":"$password"}' 'http://NS_URL/api/auth/login'
```

Where:

$username is your user name
$password is your user password
NS_URL is your Network Server base URL

- For US Server please use https://lorawan-ns-na.tektelic.com/swagger-ui.html#/
- For EU Server please use https://lorawan-ns-eu.tektelic.com/swagger-ui.html#/

After entering the above curl command, it would give you a token similar to the below

{"token":"eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJjdXN0b21lckB0ZWt0ZWxpYy5jb20iLCJzY29wZXMiOlsiQ1VTVE9NRVJfQURNSU4iXSwidXNlcklkIjoiZTEzMzc3ZTAtMzA1OS0xMWU5LThiMGQtM2Y0NTQ5OGExNTlhIiwiZW5hYmxlZCI6dHJ1ZSwicHJvdmlkZXJJZCI6ImUwZjcwYjIwLTMwNTktMTFlOS04YjBkLTNmNDU0OThhMTU5YSIsImN1c3RvbWVySWQiOiJlMTI5OGNkMC0zMDU5LTExZTktOGIwZC0zZjQ1NDk4YTE1OWEiLCJpc3MiOiJ0ZWt0ZWxpYy5jb20iLCJpYXQiOjE1NTYxMzQ2OTAsImV4cCI6MTU1NjE0MTg5MH0.CkrT6b8u5vR2fEXc56DgtxJyaD4yQOQVz-tdEvg2rpGr93Zv_rSb_jWkp2Tr5ES3JjqU1AoC2Qq0SvakL7cA","refreshToken":"eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJjdXN0b21lckB0ZWt0ZWxpYy5jb20iLCJzY29wZXMiOlsiUkVGUkVTSF9UT0tFTiJdLCJ1c2VySWQiOiJlMTMzNzdlMC0zMDU5LTExZTktOGIwZC0zZjQ1NDk4YTE1OWEiLCJpc3MiOiJ0ZWt0ZWxpYy5jb20iLCJqdGkiOiIwNjBhMDQyNC1kN2Y4LTQ1YzAtYmY5Ni1iMDZhNDAxNzIwYjkiLCJpYXQiOjE1NTYxMzQ2OTAsImV4cCI6MTU1NjIyMTA5MH0.uvqcm6x18zNI_VRLJIt9y5mUENy1PvlUN4ijm7kK13xEPj0xd6hNnO_X4wUvZHCf2b-VxVOWFymaKUSL10m8sg"}

Copy the first part of the token (as highlighted above) This is the JWT token

Go to swagger page for the Network Server: (Below Screen shots use US instance)

- US Server: https://lorawan-ns-na.tektelic.com/swagger-ui.html#/

- EU Server https://lorawan-ns-eu.tektelic.com/swagger-ui.html#/

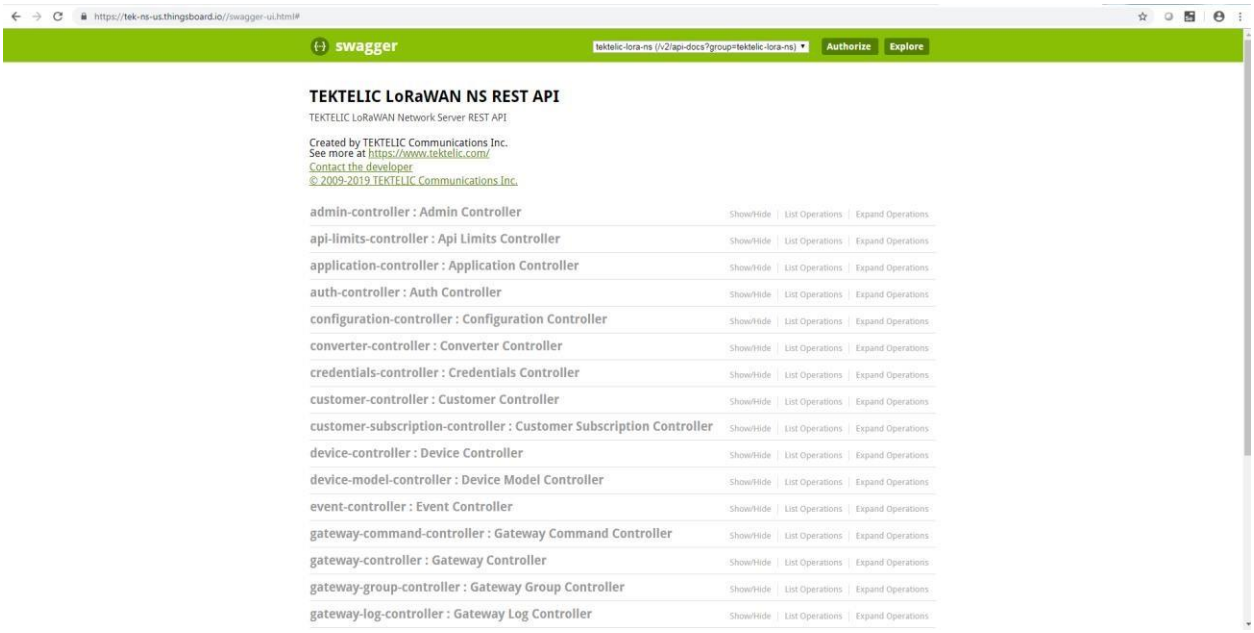


Figure 1 Tektelic NS REST API

Click Authorize button at the , And type: “Bearer” then a space then paste the JWT token that you copied before. Select Authorize.

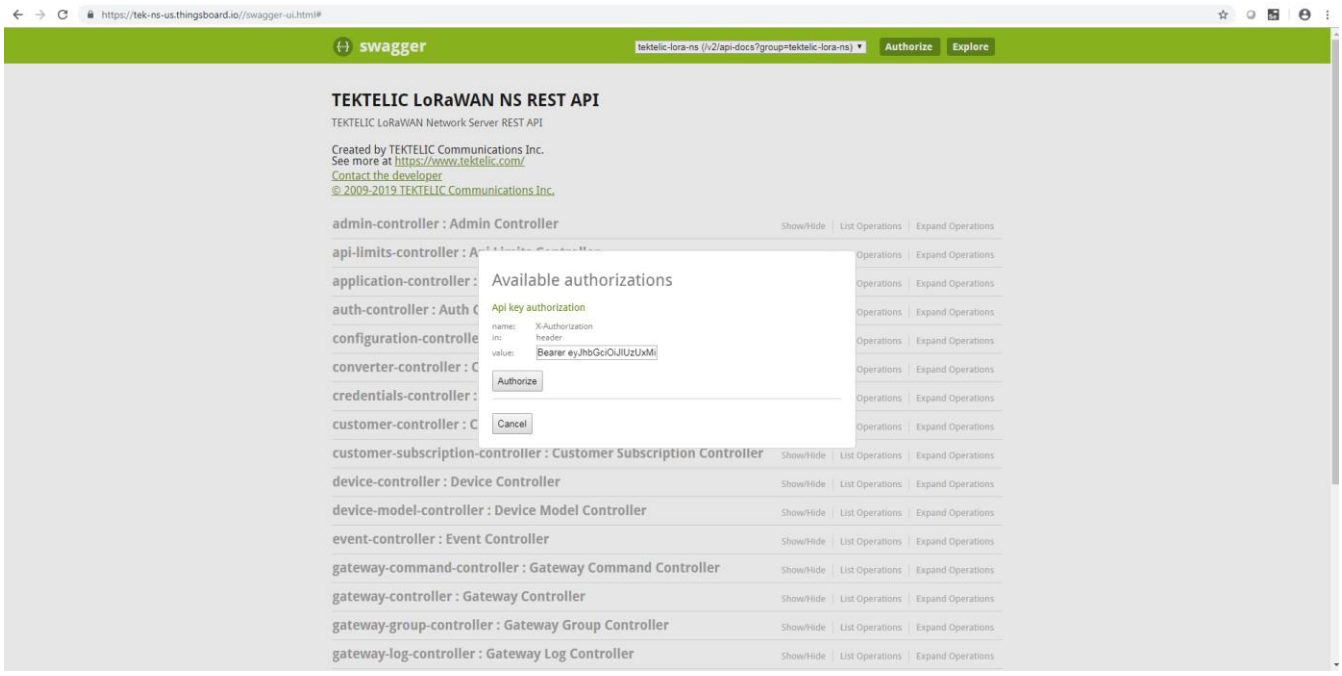


Figure 2 Available Authorizations

Select the API that you want to get information from NS. In this example we selected Devicecontroller then selected "GET /api/device/eui/{eui}"

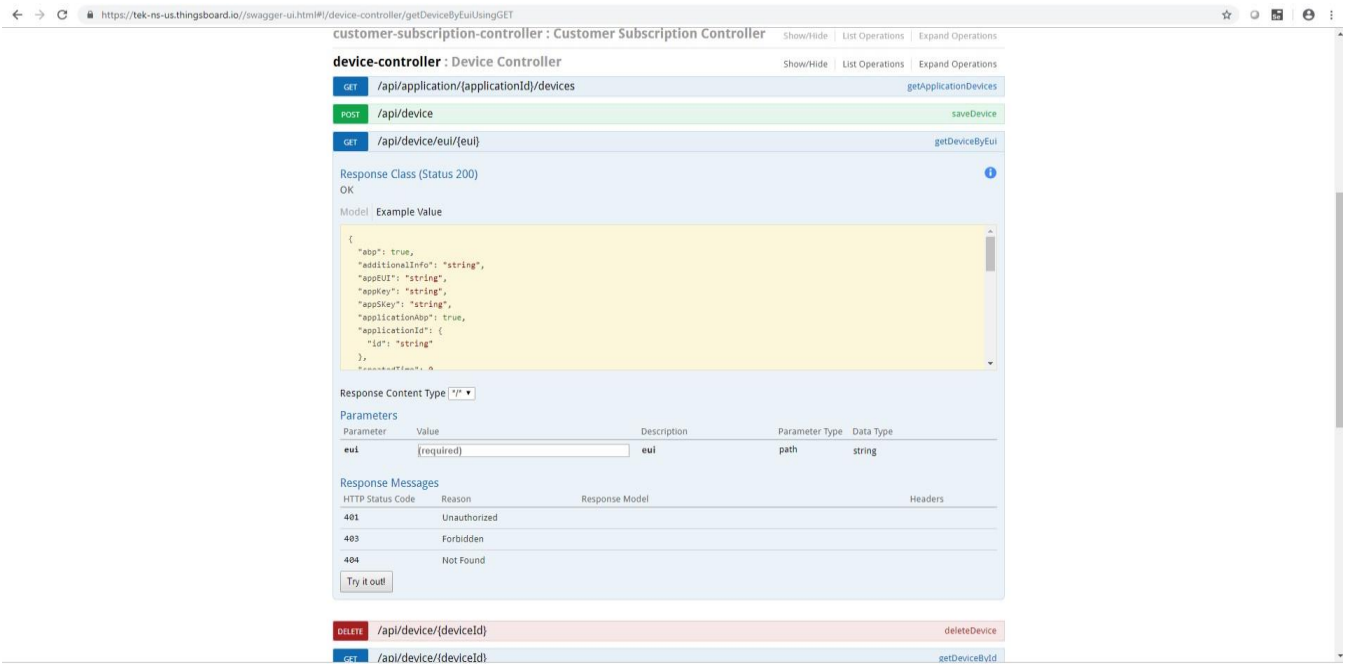


Figure 3 API Selection

Enter the EUI for your device in the "value" field, then select "Try it out!". You should receive device information in the response body from the NS.

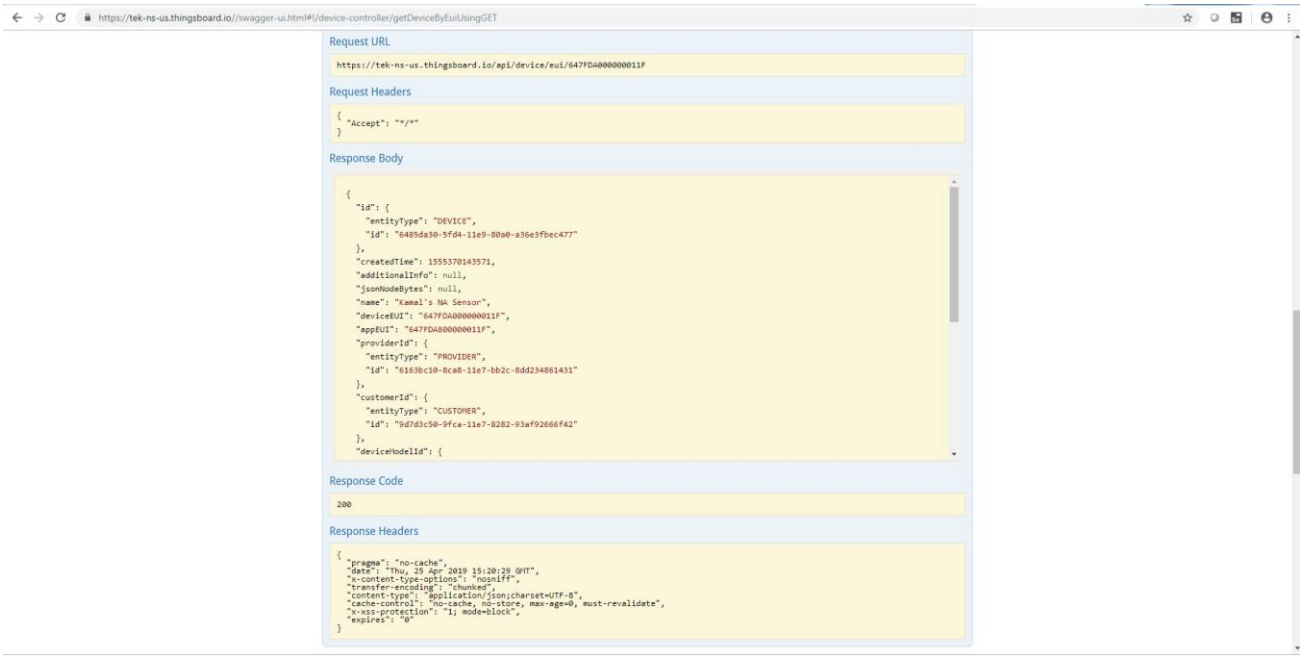


Figure 4 Device Information

2.2 MQTT API overview

NS acts as a limited MQTT broker. This means that NS provides MQTT compatible APIs that allow gateways and server side applications to interact with the NS.

2.2.1 Gateway Bridge MQTT APIs

Gateway Bridge MQTT APIs are designed for compatibility with legacy UDP packet forwarder protocol. By design, these APIs do not support authentication of particular gateways. However, if gateway is not present in the NS database, it will not be able to push data to the NS. This is controlled by the NS application logic and not by the transport security layer.

It is possible to disable Gateway Bridge MQTT APIs. See configuration guide for more details.

Topic for received packets from gateway: gateway/[mac]/rx Example

payload:

```
{
    "phyPayload": "AAEBAQEBAQEBAgICAgICAgJpNbxrAh8=",  // base64 encoded LoRaWAN packet
    "rxInfo": {
        "channel": 1,
        "codeRate": "4/5",
        "crcStatus": 1,
```

```
        "dataRate": {
            "bandwidth": 125,
            "modulation": "LORA",
            "spreadFactor": 7
        },
        "frequency": 868300000,
        "loRaSNR": 7,
        "mac": "1dee08d0b691d149",
        "rfChain": 1,
        "rssi": -57,
        "size": 23,
        "time": "0001-01-01T00:00:00Z",
        "timestamp": 2074240683                          // gateway internal timestamp
(32 bit) with microsecond precision
    }
}
```


Topic for transmitted packets to gateway: gateway/[mac]/tx Example

payload:


```
{
    "phyPayload": "IKu70cumKom7BREUFrxlHtM=",
    "txInfo": {
        "codeRate": "4/5",
        "dataRate": {
            "bandwidth": 125,
            "modulation": "LORA",
            "spreadFactor": 7
        },
        "frequency": 868300000,
        "immediately": false,
        "mac": "1dee08d0b691d149",
        "power": 14,
        "timestamp": 2079240683
    }
}
```


2.2.2 Gateway MQTT APIs

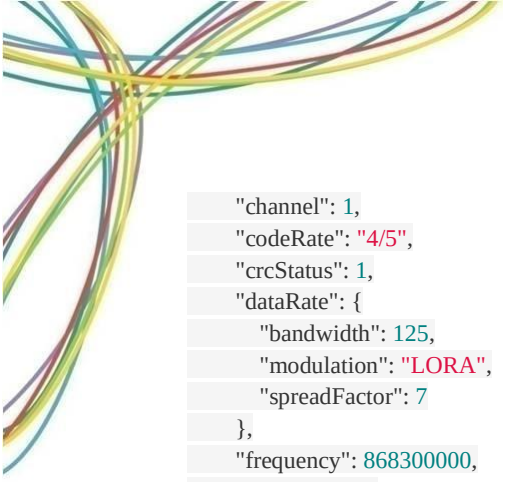
Gateway MQTT APIs are designed to be similar to Gateway Bridge MQTT APIs. The difference is that they enable credentials check on the transport level during connection procedure of Gateway MQTT client.

Connect message assumes NS credentials to be present as a username and password.

Topic for received packets from gateway: gateway/rx Example

payload:


```
{
  "phyPayload": "AAEBAQEBAQEBAgICAgICAgJpNbxrAh8=",  // base64 encoded LoRaWAN packet
  "rxInfo": {
```



```
      "channel": 1,
      "codeRate": "4/5",
      "crcStatus": 1,
      "dataRate": {
         "bandwidth": 125,
         "modulation": "LORA",
         "spreadFactor": 7
      },
      "frequency": 868300000,
      "loRaSNR": 7,
      "mac": "1dee08d0b691d149",
      "rfChain": 1,
      "rssi": -57,
      "size": 23,
      "time": "0001-01-01T00:00:00Z",
      "timestamp": 2074240683                    // gateway internal timestamp (32 bit) with microsecond precision
   }
}
```

Topic for received packets from gateway: gateway/tx Example

payload:

```
{
   "phyPayload": "IKu70cumKom7BREUFrxlHtM=",
   "txInfo": {
      "codeRate": "4/5",
      "dataRate": {
         "bandwidth": 125,
         "modulation": "LORA",
         "spreadFactor": 7
      },
      "frequency": 868300000,
      "immediately": false,
      "mac": "1dee08d0b691d149",
      "power": 14,
      "timestamp": 2079240683
   }
}
```

2.2.3 Application MQTT APIs

Application MQTT APIs are designed to receive uplink message and join notification and push downlink messages. This APIs are enabled by default and is the basic integration APIs for server side applications. Other integration APIs are enabled separately using Application Integrations UI or REST API.

Connect message assumes NS credentials to be present as a username and password.

Uplink API

You can find example of subscription command below:

mosquitto_sub -h "NS_HOST" -t "app/#" -v -u "APP_CREDENTIALS_KEY" -P "APP_CREDENTIALS_VALUE"

Where

NS_HOST is a NS host name

APP_CREDENTIALS_KEY is a NS Application credentials key.

APP_CREDENTIALS_VALUE is a NS Application credentials value.

The following is an example of device JOIN notification. This notification arrives to **app/join** topic.


```
{
   "applicationMetaData":{
      "id":{
         "entityType":"APPLICATION",
         "id":"617b6350-8353-11e7-8437-5fb048d81198"
      },
      "customerId":{
         "entityType":"CUSTOMER",
         "id":"61261760-8353-11e7-8437-5fb048d81198"
      },
      "subCustomerId":null,
      "name":"Demo Application"
   },
   "gatewayMetaDataList":[
      {
         "id":{
            "entityType":"GATEWAY",
            "id":"616e43f0-8353-11e7-8437-5fb048d81198"
         },
         "name":"Demo Gateway",
         "mac":"647FDAFFFE0041CA",
         "latitude":null,
         "longitude":null,
```

```
            "altitude":null,
            "rxInfo":{
               "channel":0,
               "codeRate":"4/5",
               "crcStatus":1,
               "dataRate":{
                  "modulation":"LORA",
                  "spreadFactor":12,
                  "bandwidth":125
               },
               "frequency":868100000,
               "loRaSNR":9,
               "mac":"647fdafffe0041ca",
               "rfChain":1,
               "rssi":-59,
               "size":23,
               "time":"2017-08-17T15:08:27Z",
               "timestamp":254825812
            }
         }
      ],
      "deviceMetaData":{
         "id":{
            "entityType":"DEVICE",
            "id":"7ee420d0-8358-11e7-9b14-5fb048d81198"
         },
         "name":"1723D0012",
         "type":null,
         "deviceEUI":"647FDA0000000148",
         "appEUI":"647FDA8000000148"
      }
   }
```

The following is an example of device UPLINK notification. This notification arrives to **app/rx** topic.


```
{
   "payloadMetaData":{
      "applicationMetaData":{
         "id":{
            "entityType":"APPLICATION",
            "id":"617b6350-8353-11e7-8437-5fb048d81198"
         },
         "customerId":{
            "entityType":"CUSTOMER",
            "id":"61261760-8353-11e7-8437-5fb048d81198"
         },
         "subCustomerId":null,
         "name":"Demo Application"
      },
      "gatewayMetaDataList":[
         {
            "id":{
               "entityType":"GATEWAY",
               "id":"616e43f0-8353-11e7-8437-5fb048d81198"
            },
            "name":"Demo Gateway",
            "mac":"647FDAFFFE0041CA",
            "latitude":null,
            "longitude":null,
            "altitude":null,
            "rxInfo":{
               "channel":2,
               "codeRate":"4/5",
               "crcStatus":1,
               "dataRate":{
```

```
                "modulation":"LORA",
                "spreadFactor":12,
                "bandwidth":125
            },
            "frequency":868500000,
            "loRaSNR":8,
            "mac":"647fdafffe0041ca",
            "rfChain":1,
            "rssi":-43,
            "size":20,
            "time":"2017-08-17T15:16:53Z",
            "timestamp":760310156
        }
    }
  ],
  "deviceMetaData":{
    "id":{
      "entityType":"DEVICE",
      "id":"7ee420d0-8358-11e7-9b14-5fb048d81198"
    },
    "name":"1723D0012",
    "type":null,
    "deviceEUI":"647FDA0000000148",
    "appEUI":"647FDA8000000148"
  },
  "fcount":1,
  "fport":10
 },
 "payload":"BQD/CAQAAA=="
}
```

Downlink API

The following is an example of publish command:

mosquitto_pub -h "NS_HOST" -t "app/tx" -u "APP_CREDENTIALS_KEY" -P "APP_CREDENTIALS_VALUE" -f “FILE”

Where

NS_HOST is a NS host name

APP_CREDENTIALS_KEY is a NS Application credentials key.

APP_CREDENTIALS_VALUE is a NS Application credentials value.

FILE is a path to your downlink JSON. The structure of JSON file is listed below:

{"msgId":"1", "devEUI":"647FDA0000000148", "port":3, "confirmed": false, "data": "AQ=="}

Where **msgId** is a unique id of the message within

corresponding device **devEUI** is a device identifier

Confirmed is a boolean value that identifies is this a confirmed message or not.

port is a LoRaWAN **fPort data** is a Base64 string with the binary data for

device.

3 ThingsBoard Integration

It is possible to setup data stream from Tek NS to ThingsBoard platform. One needs to create corresponding integration and specify data converter for the application. The important things to notice is that Token in Integration configuration should correspond to TB Gateway Device Access Token.

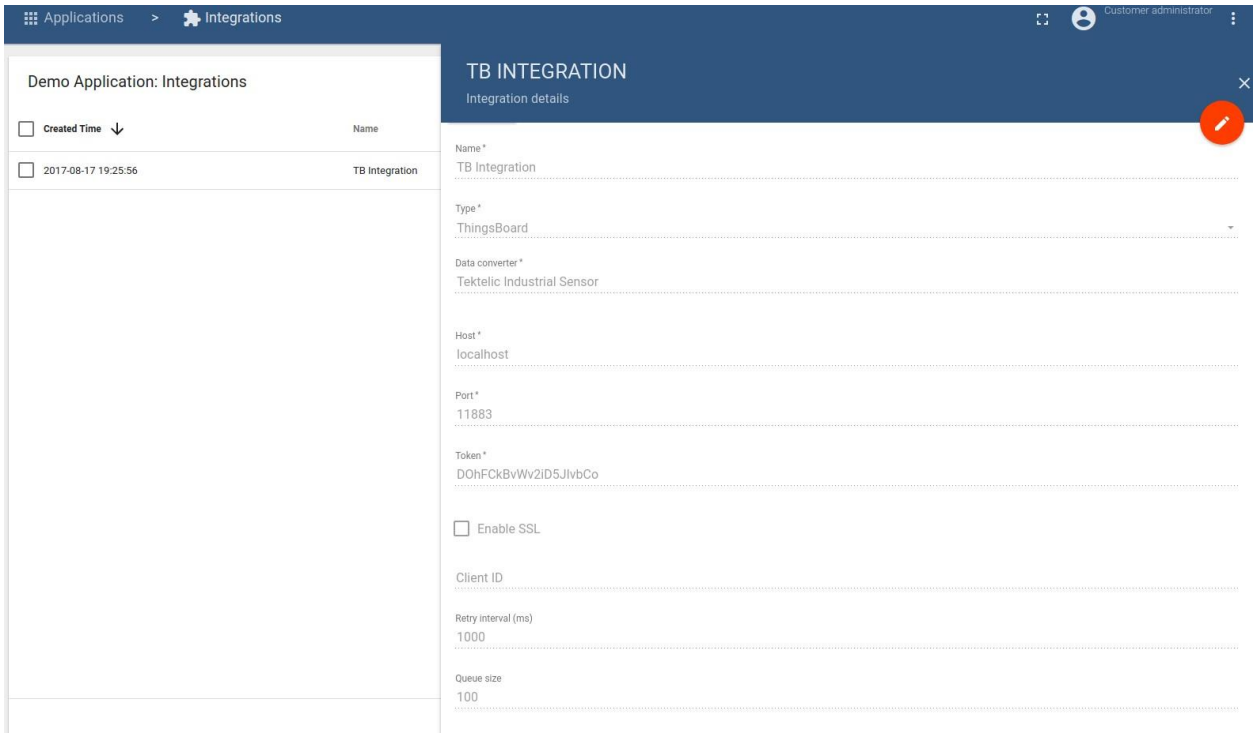


Figure 5 ThingsBoard Integration Platform

Once integration is created, browse TB and find your devices. There will be at least two devices created. One corresponds to your sensor and one to the LoRaWAN Gateway device. Both devices will have attributes and telemetry values. The devices will be visible in TB after corresponding integration provisioned once one of the devices will execute any of the Uplink commands.

Integration Steps:

1) Login to ThingsBoard as a tenant administrator. If you don't have tenant administrator account - create one using system administrator account. If you don't have system administrator account - contact your server administrator.
2) Go to Devices and/or Device Groups depending on your TB version: community or professional edition.
3) Create new Device and don't forget to mark it as a gateway
4) Copy the access token of the new gateway device
5) Login as a Customer to your NS instance
6) Navigate to applications

7) Select your application and open application details
8) Click on “Integrations”
9) Create new integration. Choose ThingsBoard and use 9883 port and access token from step 4.

4 GPS Position

Gateway location information, for gateways that do not have a GPS receiver, can be added manually. Click on the pencil icon to edit the fields.

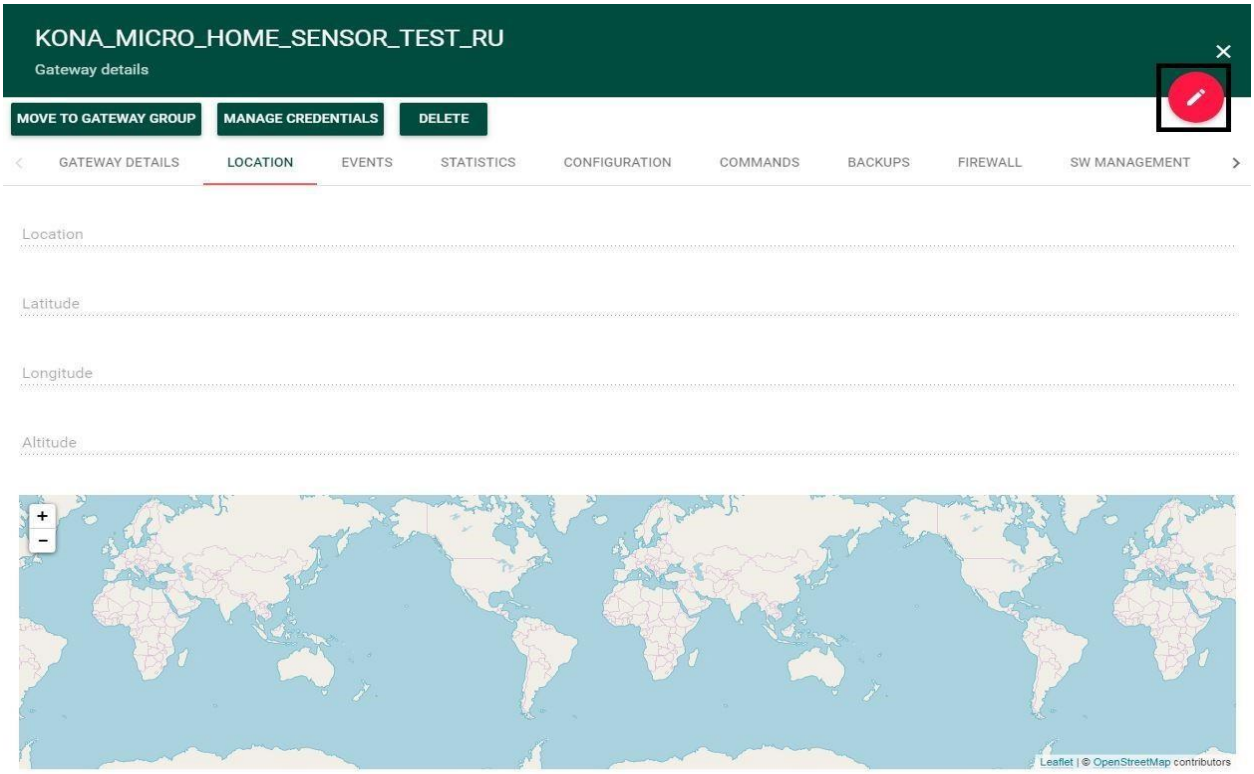


Figure 6 GPS Information through Location tab on Tektelic NS

Post adding the information, click on Save button (shown in Figure-7) to save these details related to location of the GW.

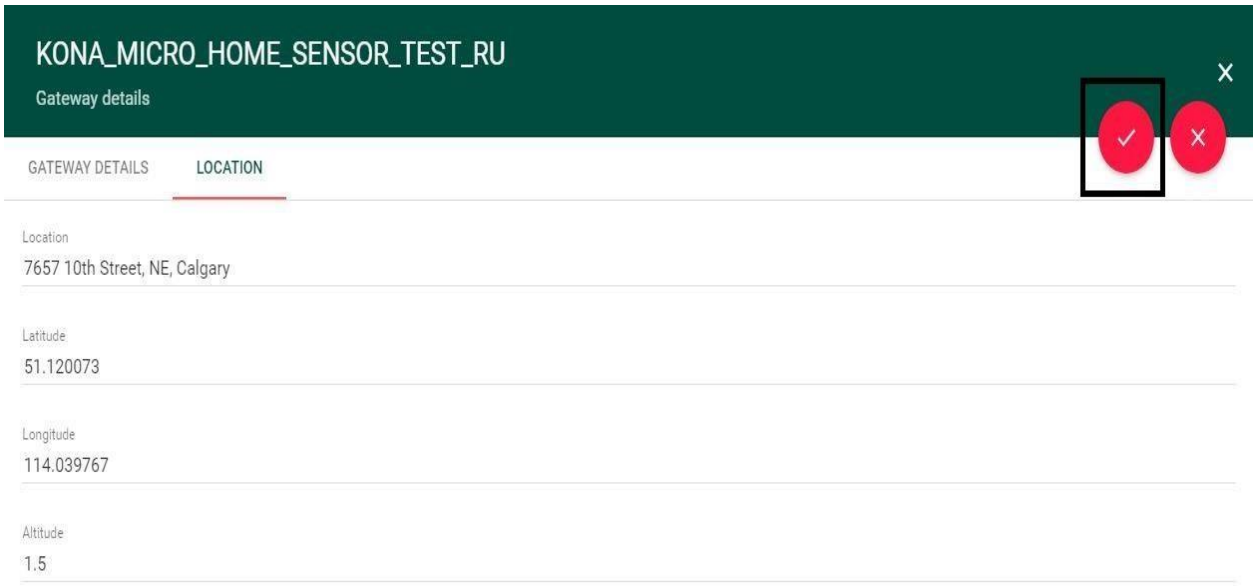


Figure 7 Saving Location information of Kona Micro GW

Figure-8 illustrates the location information of a GW after adding and saving the details.

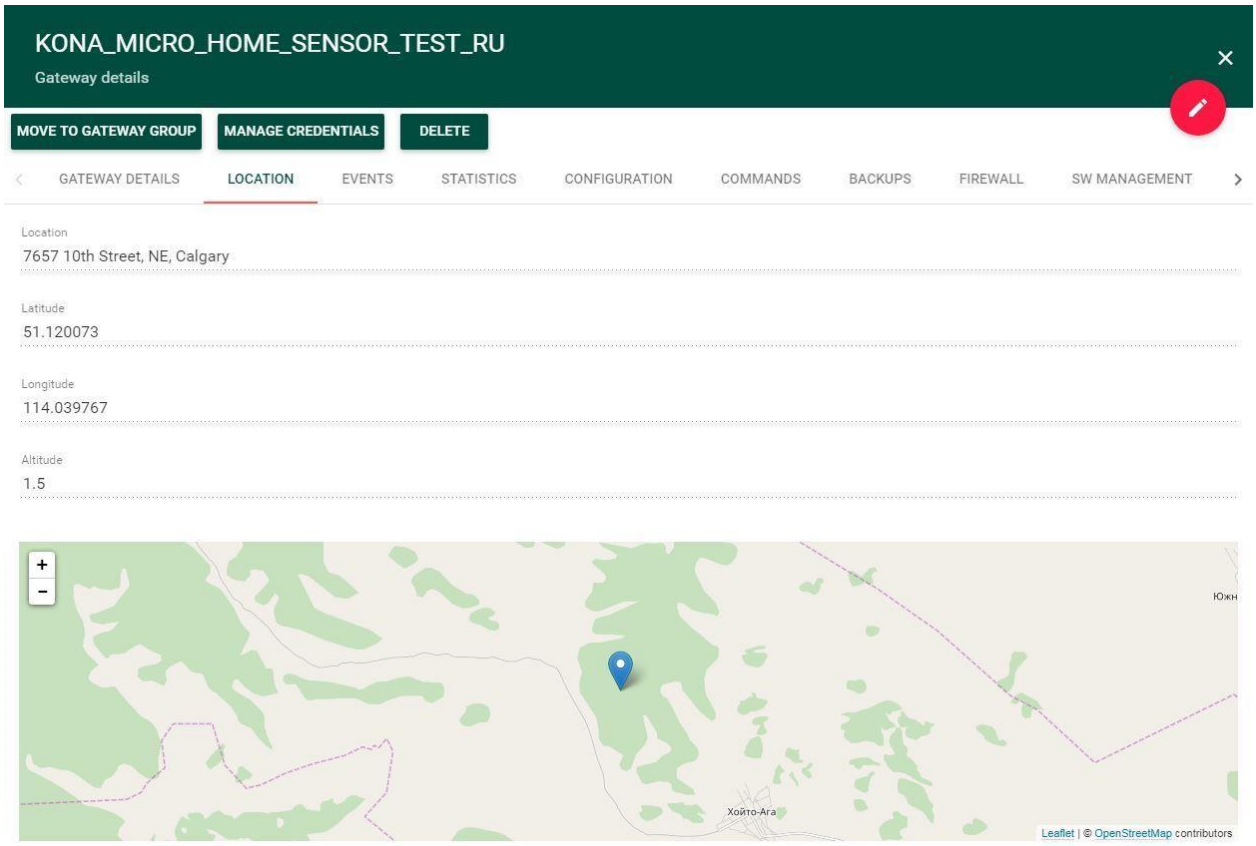


Figure 8 Location information for Kona Micro GW

5 Real Time Packets

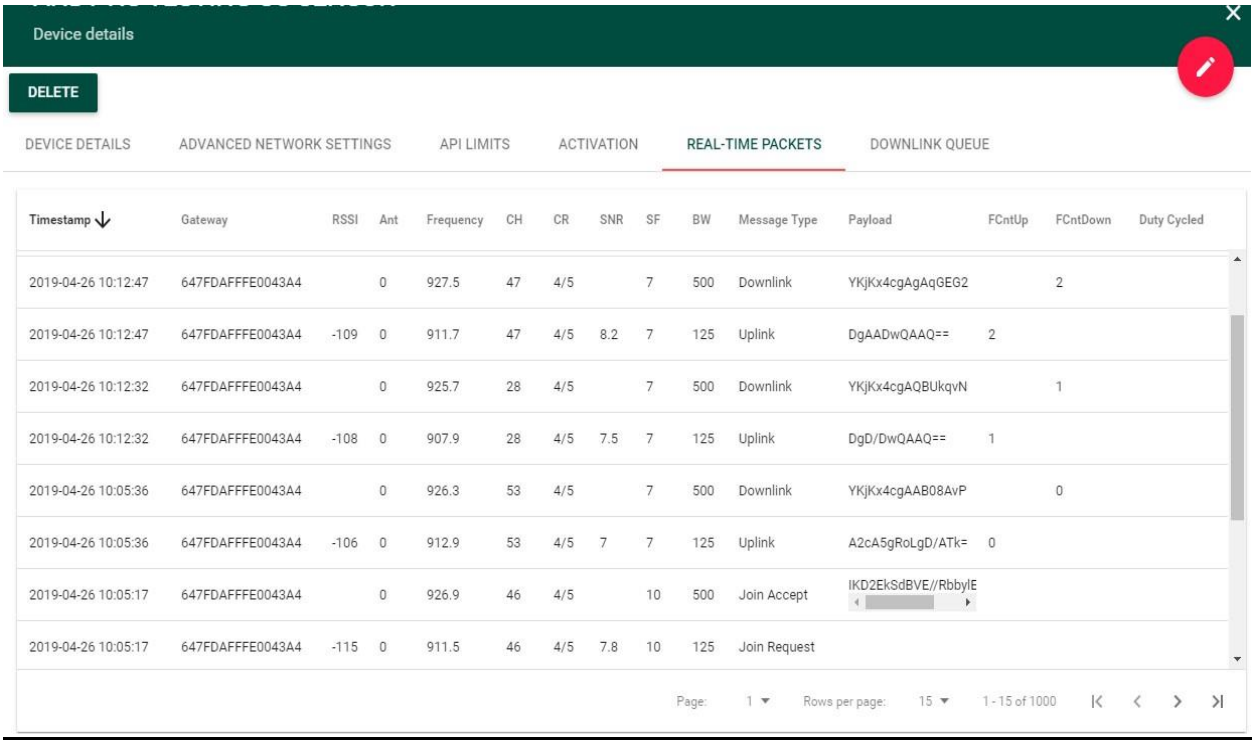


Figure 9 GUI displaying Real Time Packets

Gateway location information, for gateways that do not have a GPS receiver, can be added manually. Click on the pencil icon to edit the fields.

Real Time Packets tab in NS provides information related to packet traffic between Network Server (NS) and Sensor. Below table summarizes the description of each field available in Real Time Packets tab.

Field	Description
Timestamp	Time at which NS either • Receives an uplink or • Issues a downlink
Gateway	ID of the Gateway that receives the packet from Sensor
RSSI	Parameter indicating signal strength
Ant	Points out the RF Channel in use
Frequency	Radio Frequency of Uplink and Downlink

CH	Indicates the LoRa WAN channel for particular use NB Uplink: 0-63 WB Uplink: 64-71
	WB Downlink: 0-7 Note that the above channel plan applies only to the US902-928 region. Channel plans vary by region.
CR	Coding Rate for Forward Error Correction
SNR	Signal to Noise Ratio
SF	LoRa WAN data-rate (Spread Factor)
BW	Bandwidth of a specific packet
Message Type	Type of the message indicating communication process between device and GW in NW Uplink : Data sent from End Device (Sensor) to NS Downlink : Data sent from NS to End Device (Sensor) Join Request : Sensor attempts to join to the network Join Accept : NS approved the request of Sensor to join the network
Payload	Unencrypted Uplink or Encrypted Downlink
FCntUp	Frame Counter Up
FCntDown	Frame Counter Down

NOTE:

- Sensor uplink data can be received by multiple gateways that forward the data to the network server. All of these GWs can also forward the data to NS. NS employs an algorithm to sort them out based on the RSSI and SNR.

Information on Payload:

- The UL packets posted on the real-time packets page are unencrypted, as are those displayed in the Gateway Logs.

- The DL packets posted on the real-time packets page are encrypted, as are those displayed in the Gateway Logs.

- Whether the keys are provided or not, the raw packet data (i.e. PHYPayload) can always be decoded: For example,

```
PHYPayload = 80A97A04D30005006498AB89D188D0314C377428C337
```

- However, the FRMPayload portion will still be encrypted, which is where the application payload lives. As an example, the uplink above is the device answering a request for its DevEUI:

```
FRMPayload = 98AB89D188D0314C37 (from packet, encrypted)
           = 00647FDA0000000DBE (decrypted)
```

- The first byte is the register address of the DevEUI, the remaining is the DevEUI itself.

6 Gateway Logging

GW Logs window in Tektelic NS appears as shown in Figure-10. Details must be filled to extract the logs between certain duration.

1. GW ID is the 16 Digit ID of the GW.

2. DeviceEUI represents the device in the set-up with GW.

3. Type of message provides an option to list the GW logs in different categories such as GW statistics, GW alarm event, GW RPC command, GW RPC response, and etc.

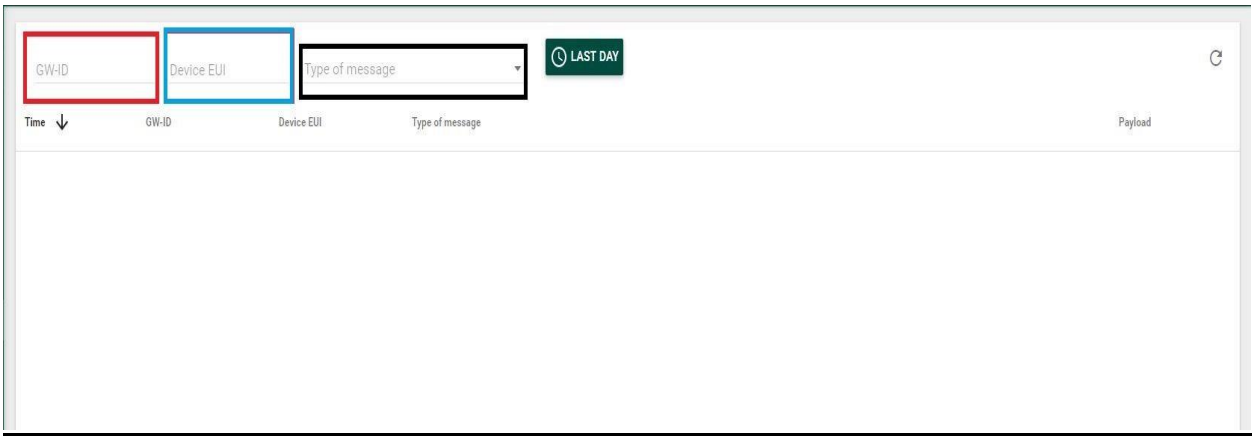


Figure 10 GUI displaying Real Time Packets

NOTE:

- To retrieve the GW logs for a particular GW, Logging Option for a particular GW from Gateway Groups tab must have been enabled first.

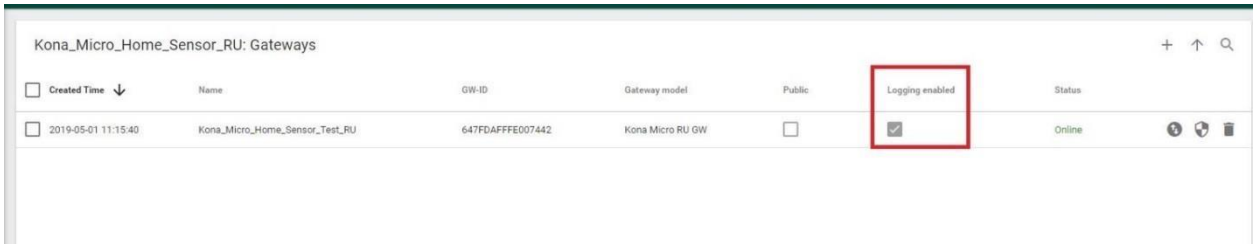


Figure 11 GW Logging Enabled

Figure-12 illustrates the GW logs window when GW logs are available.

Time ↓	GW-ID	Device EUI	Type of message	Payload
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••
2019-05-09 13:48:42	647FDAFFFE007442		Gateway event	•••

Figure 12 Gateway Logs

Below table summarizes the description of each field available in GW logs tab.

Field	Description
Time	Time at which an event takes place in NW
Gateway ID	ID of the Gateway that receives the packet from Sensor
DeviceEUI	EUI of the device (sensor)
Type of message	Indicates a specific type of messages from the list of different types of message such as • Gateway event • Gateway Statistics • Gateway Alarm Event • Gateway RPC command • Gateway RPC response • Gateway configuration update • Gateway configuration event • Join request • Join accept • Unconfirmed data up • Unconfirmed data down • Confirmed data up • Confirmed data down
Payload	Message comprising details of the packet traffic. Note: Payload details vary according to the type of messages.

After successfully integrating the sensor and GW with the application platform, downlink packets can be seen in the Downlink Queue tab on Tektelic NS. Figure-13 demonstrates the downlink data along with the port number.

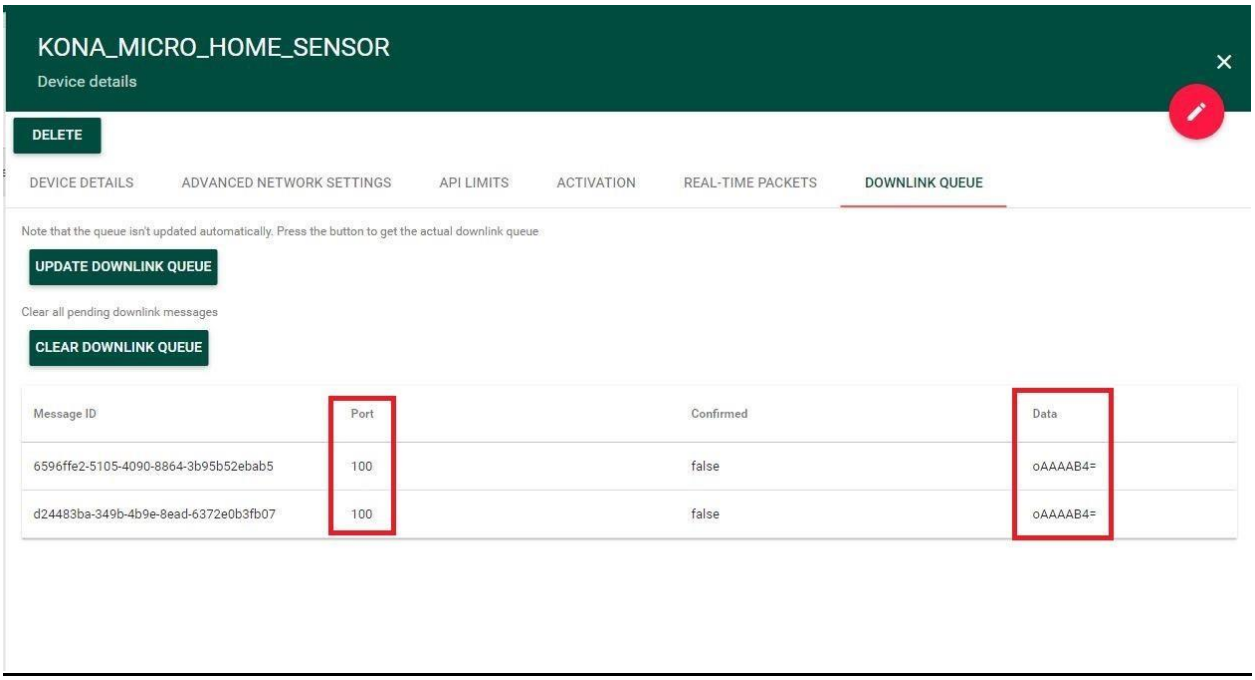


Figure 13 Downlink Queue