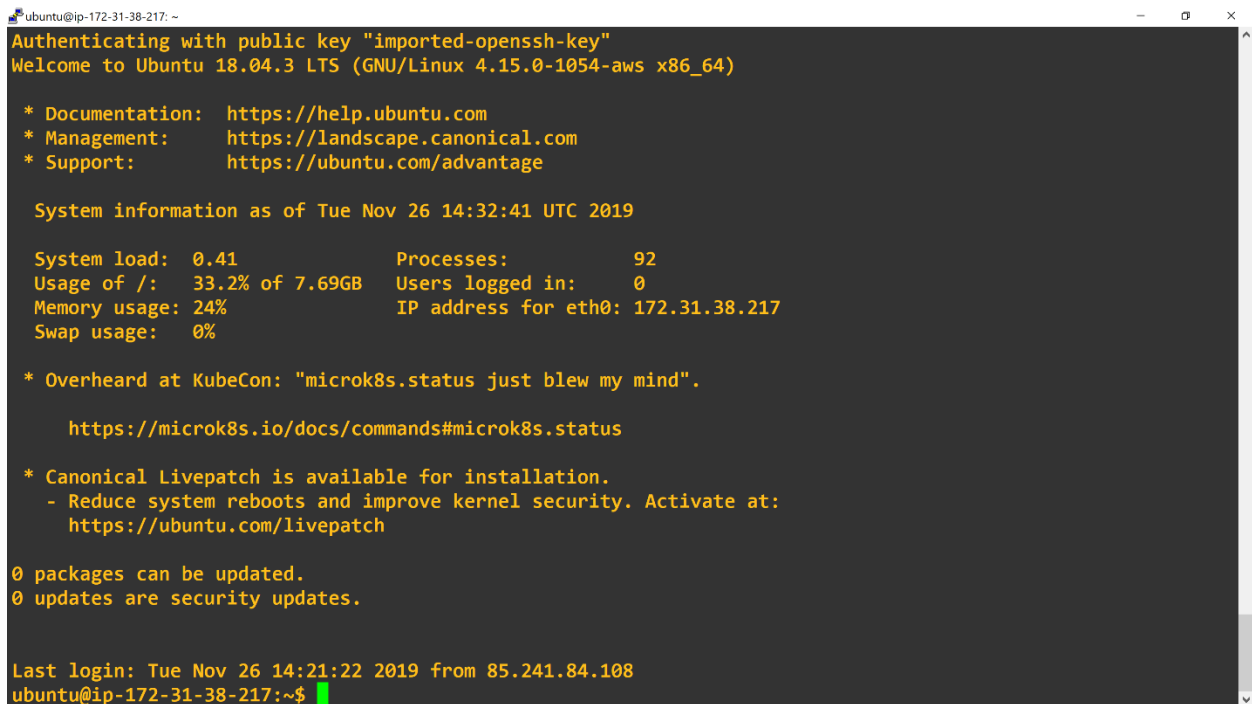


Node-RED

This document contains the steps required for the installation and configuration of Node-RED. Node-RED will be used as backend allowing to do data processing of the packets received by The Things Network (TTN), the purposed idea is having devices communicating through LoRa with a gateway. This gateway will forward the packet to TTN, TTN will then communicate by MQTT to Node-RED. Node-RED will communicate also by MQTT to the Carelink platform, the reverse path is also possible. Node-RED is also used to process the "statusWifiAPs" messages (that are published in the MQTT broker, with a field containing the Wi-Fi access points that are near the device), which will make use of 3 Wi-Fi location API's to determine an "assisted" location, complementary to the gps. This will enable the pycom devices to have one more alternative to the gps, in case it fails. Additionally, a third localization technique is also available when using LoRa protocol.

1. Installation

SSH connection:

A screenshot of a terminal window titled 'ubuntu@ip-172-31-38-217: ~'. The terminal displays the Ubuntu 18.04.3 LTS login screen. It starts with 'Authenticating with public key "imported-openssh-key"' and 'Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-1054-aws x86_64)'. Below this, it lists links for documentation, management, and support. Then, it shows system information as of Tue Nov 26 14:32:41 UTC 2019, including system load, processes, memory usage, and IP address. It also mentions 'Overheard at KubeCon' and 'Canonical Livepatch is available for installation'. At the bottom, it states '0 packages can be updated.' and '0 updates are security updates.' The last line shows the last login time and the prompt 'ubuntu@ip-172-31-38-217:~\$'.

```
ubuntu@ip-172-31-38-217: ~
Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-1054-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Nov 26 14:32:41 UTC 2019

System load:  0.41               Processes:           92
Usage of /:   33.2% of 7.69GB    Users logged in:    0
Memory usage: 24%               IP address for eth0: 172.31.38.217
Swap usage:   0%

 * Overheard at KubeCon: "microk8s.status just blew my mind".

   https://microk8s.io/docs/commands#microk8s.status

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.

Last login: Tue Nov 26 14:21:22 2019 from 85.241.84.108
ubuntu@ip-172-31-38-217:~$
```

Figure 1 Ubuntu initial screen

2. Setup and Install Node-RED in the Cloud machine

Install node.js and Node-Red in the machine by running the following commands:

```
curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -  
sudo apt-get install -y nodejs build-essential  
sudo npm install -g node-red
```

To test the instance run `node-red start` after this an editor will be available at <http://<instance-ip>:1880/>

replace instance-ip by the server ip.

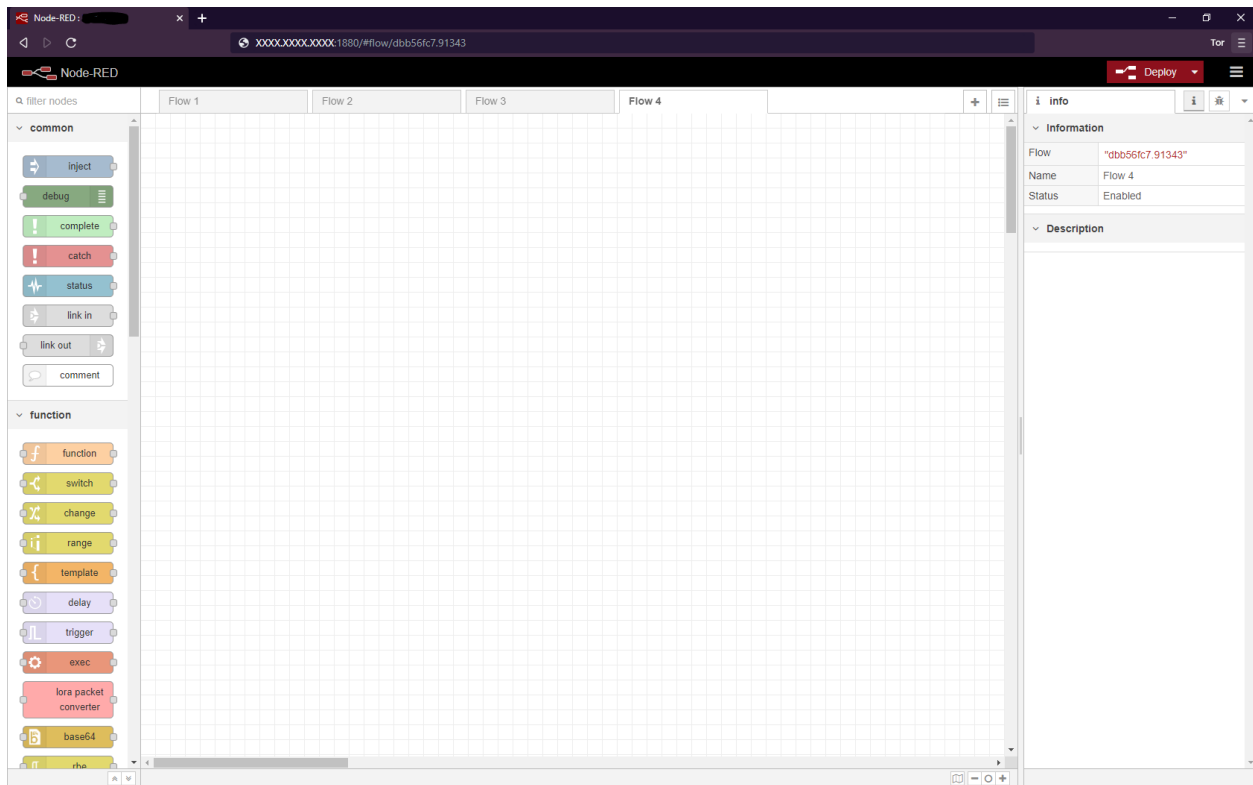


Figure 2 Node-RED Initial screen

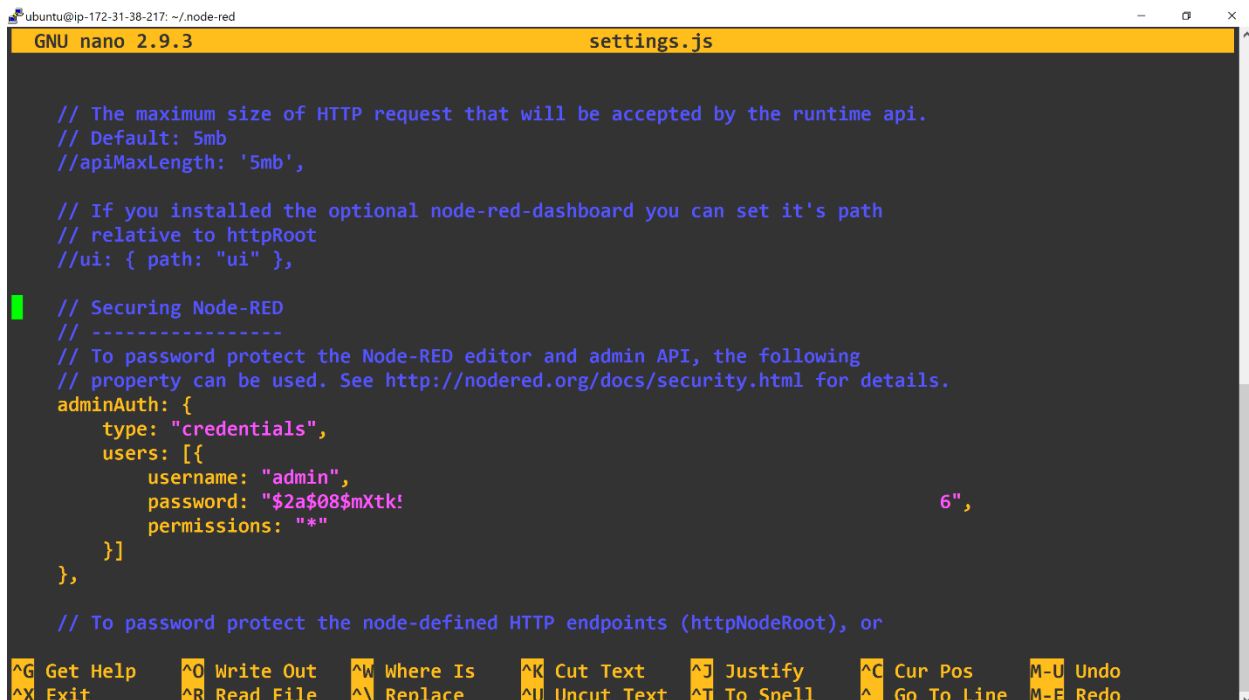
Secure Node-RED, enabling user authentication, by running these commands:

```
sudo npm install -g node-red-admin  
cd ~/.node-red  
node-red-admin hash-pw
```

Fill in the password and copy the hash, open the settings file:

```
sudo nano settings.js
```

Locate the area present in figure3, uncomment and replace the password.



```
GNU nano 2.9.3 settings.js

// The maximum size of HTTP request that will be accepted by the runtime api.
// Default: 5mb
//apiMaxLength: '5mb',

// If you installed the optional node-red-dashboard you can set it's path
// relative to httpRoot
//ui: { path: "ui" },

// Securing Node-RED
// -----
// To password protect the Node-RED editor and admin API, the following
// property can be used. See http://nodered.org/docs/security.html for details.
adminAuth: {
  type: "credentials",
  users: [{
    username: "admin",
    password: "$2a$08$mXtk!6",
    permissions: "*"
  }],
},

// To password protect the node-defined HTTP endpoints (httpNodeRoot), or
```

Figure 3 Edit settings.js

To enable a user and an admin change settings.js to



```
adminAuth: {
  type: "credentials",
  users: [
    {
      username: "admin",
      password:
"$2a$08$zZwtXTja0fB1pzD4sHCMYz2Z6dNbM6tL8sJogENOMcxwV9DN.",
      permissions: "*"
    },
    {
      username: "george",
      password:
"$2b$08$wuAqPiKJlVN27eF5qJp.RuQYuy6ZYONW7a/UWYxDtTwKFCd88F19y",
      permissions: "read"
    }
  ]
}
```

Figure 4 Settings.js user and admin

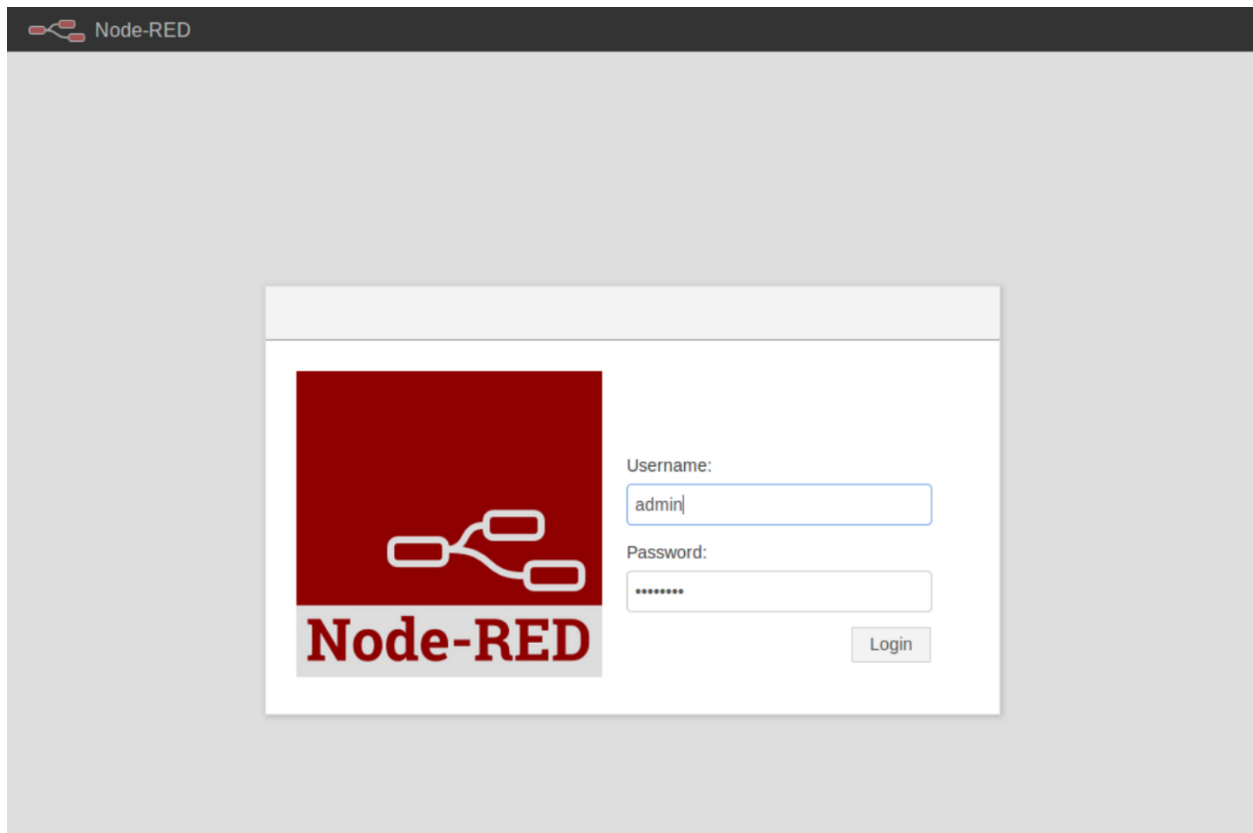


Figure 5 login page

For Node-RED start automatically whenever the instance is restarted run,

```
sudo npm install -g pm2
pm2 start `which node-red` -- -v
pm2 save
pm2 startup
```

To install a node run the next command:

```
sudo npm i node-red-node-<node-name>
```

this task can be done later in a graphic way.

More nodes are available in the following repository <https://flows.nodered.org/>

3. Use Node-RED

Node-RED is built on Node.js, and is Low-code programming for event-driven applications, taking full advantage of its event-driven, non-blocking model. This makes it ideal to run at the edge of the network on low-cost hardware such as the Raspberry Pi as well as in the cloud.

3.1 Initial Test

For an initial test go to the left menu and simply drag and drop the “inject” and “debug” nodes, in the top right the “Deploy” button will appear red and the nodes will have a blue circle in the right corner. This means the changes are not running yet.

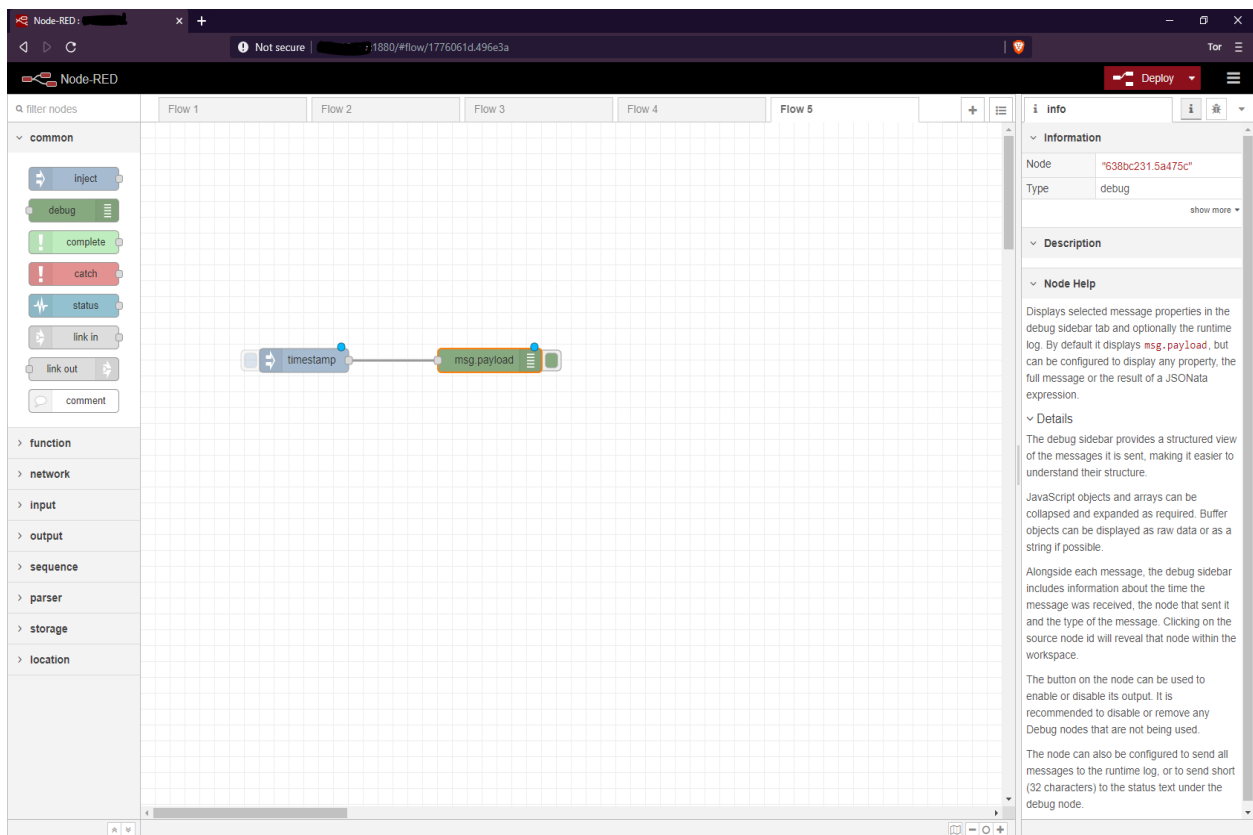


Figure 6 Node-RED simple test

After pressing the “Deploy” button, a message will appear and the blue circles will disappear the “Deploy” button should turn black, the nodes are now running.

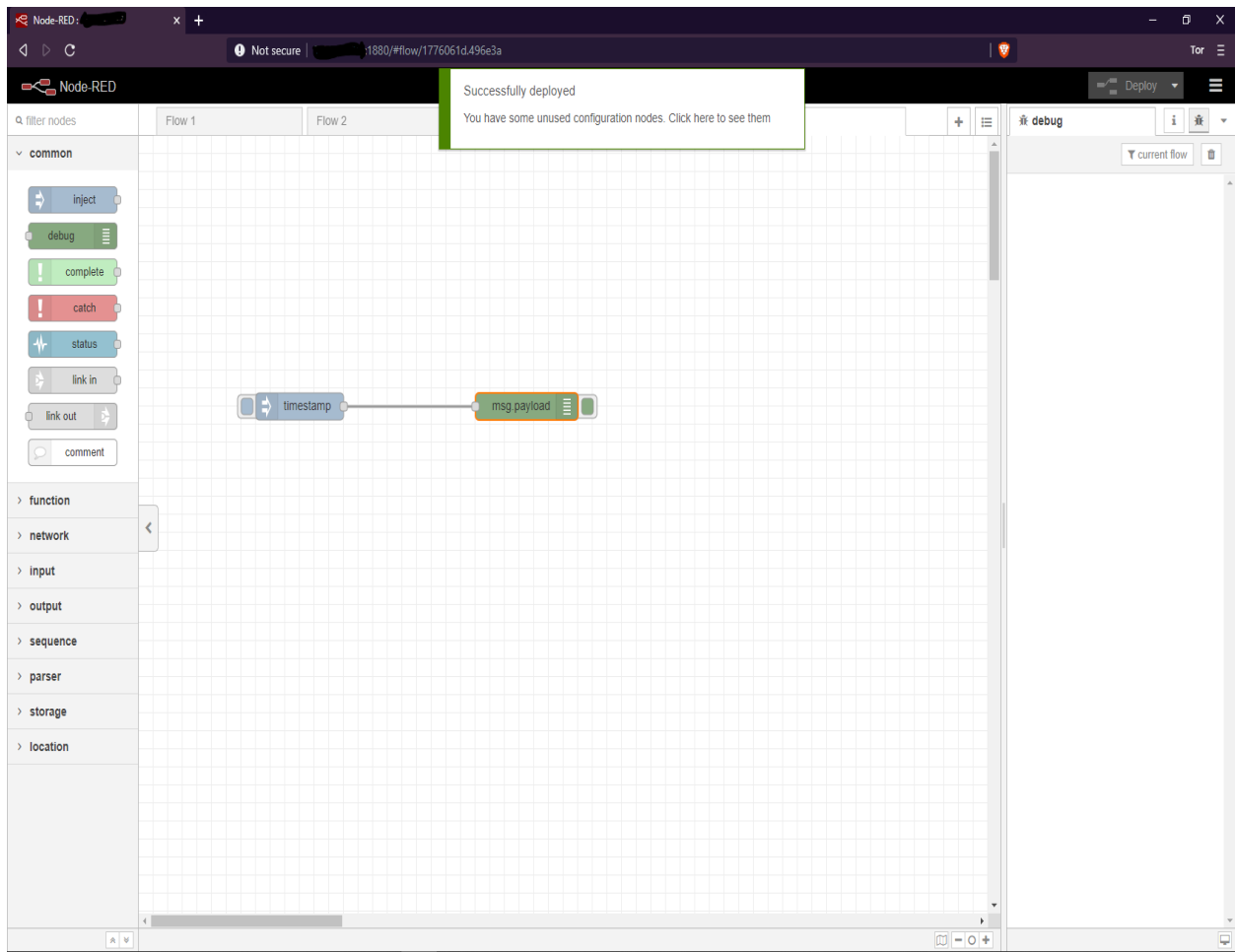


Figure 7 Node-RED deploy success

In order to get the output, go to right menu and select the icon with a bug. It's also a good idea to select current flow in the filter icon. After pressing the blue square in the first node, a timestamp will appear in the window to the right.

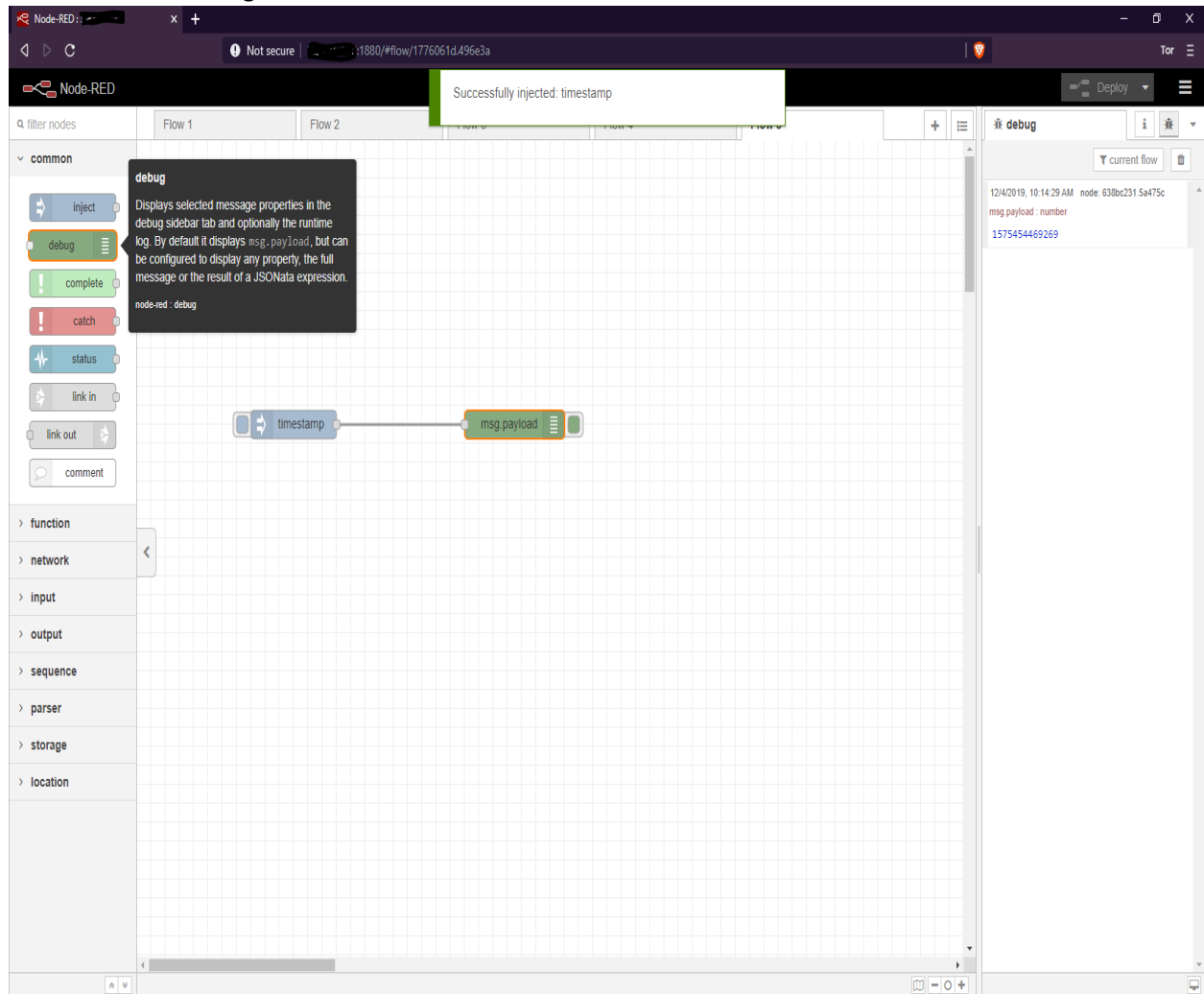


Figure 8 Node-RED debug window

Now the basic operation has been explained the only thing missing is how to export the work for future use and importing it again. To export please select top right menu and choose “Export”, a menu will appear simple press “Download”, a json file will be downloaded, this file is used as a backup or to run the flows in another machine.

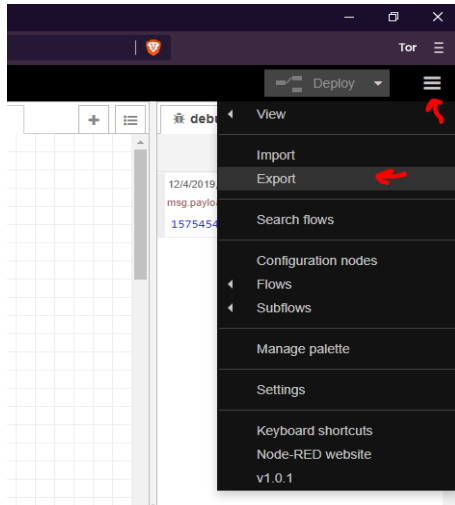


Figure 9 Node-RED export

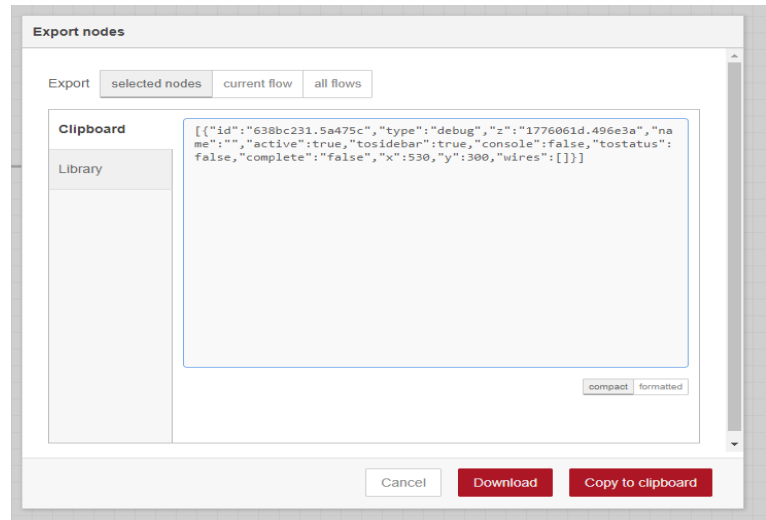


Figure 10 Node-RED export nodes

In the same menu select “Import”, and a similar option will pop up, is possible to copy and paste a json or select one that is stored locally.

The place where these nodes are going to be added is chosen in the “Import to”, by the default Node-red will import the nodes to the current flow.

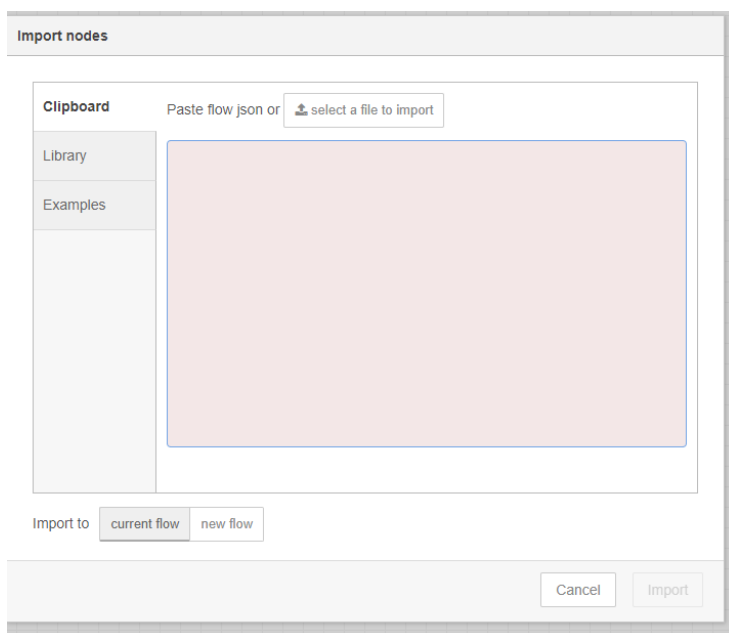


Figure 11 Node-RED import nodes

After importing the nodes, it may happen that some are missing (shown traced in Fig. 9). In order to install them it is necessary to select "Manage palette" in the menu of figure 6. Another menu will appear. Select install, and search for the missing node. An example is shown below

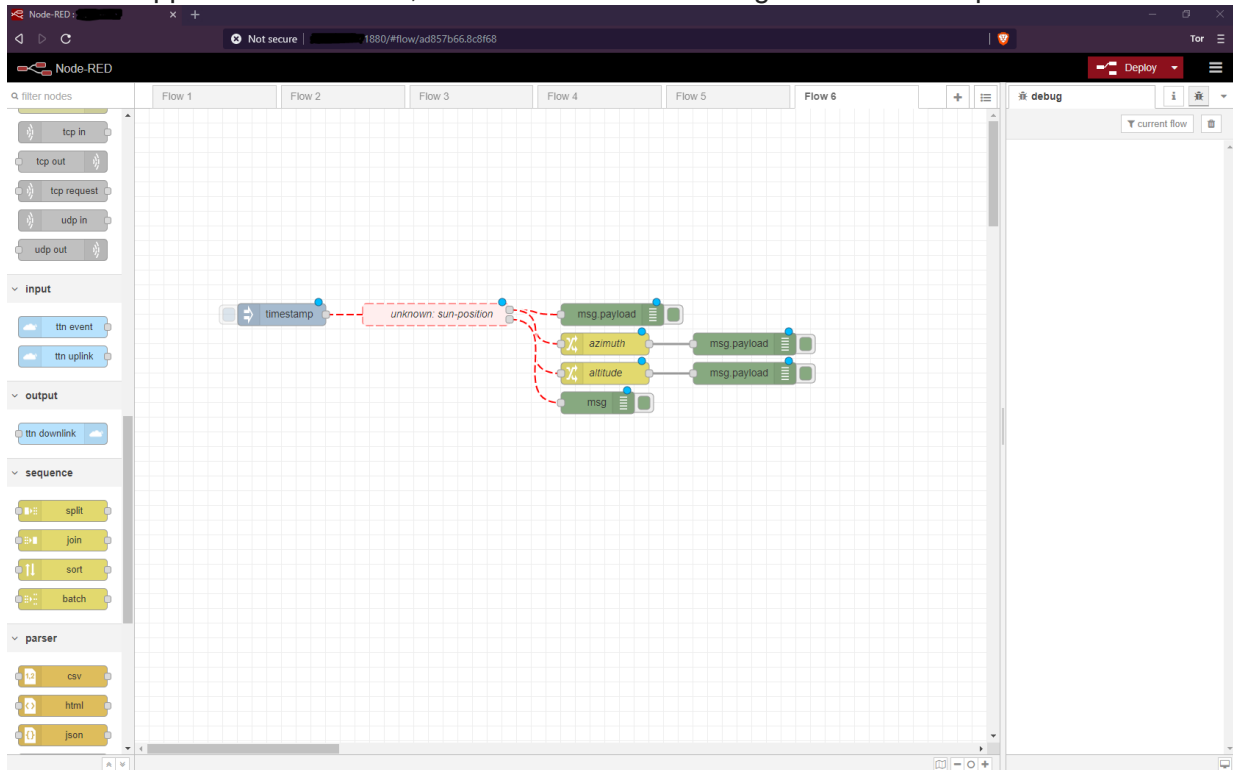


Figure 12 unknown node after import

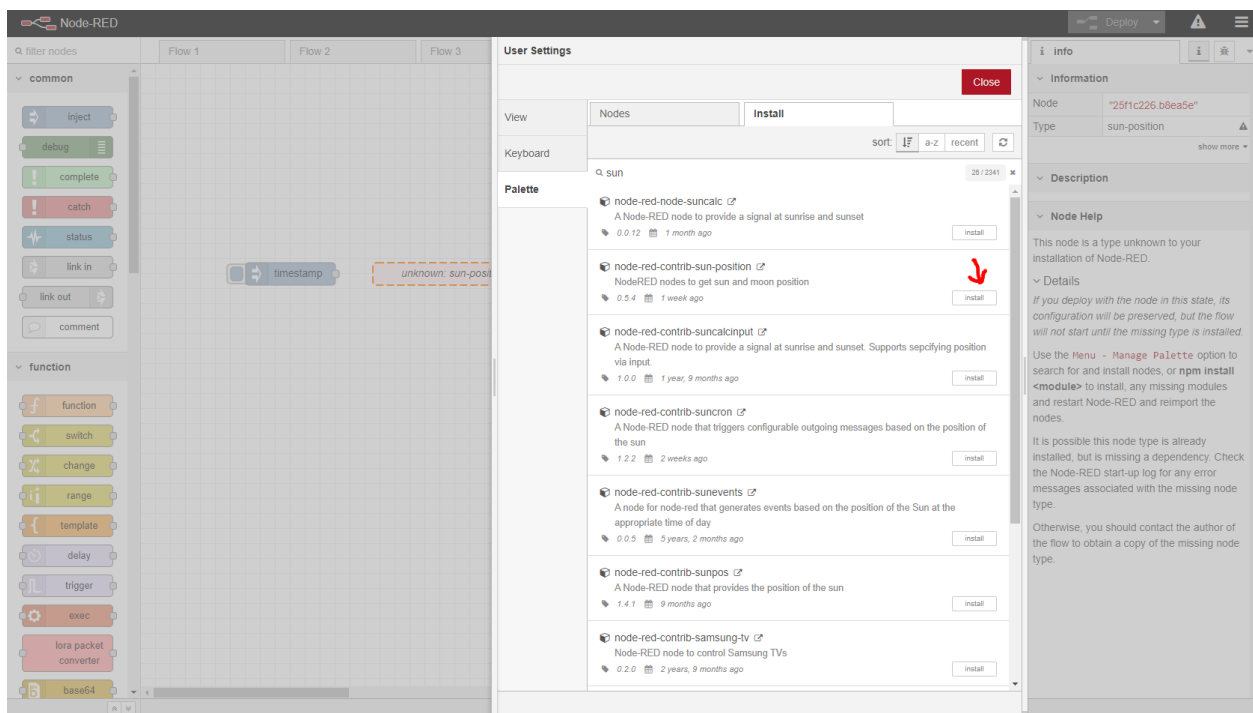


Figure 13 Node-RED install node

Select "Install" in the confirmation pop-up window. If the correct node is installed it will be automatically placed on the current flow, as shown in Fig. 12

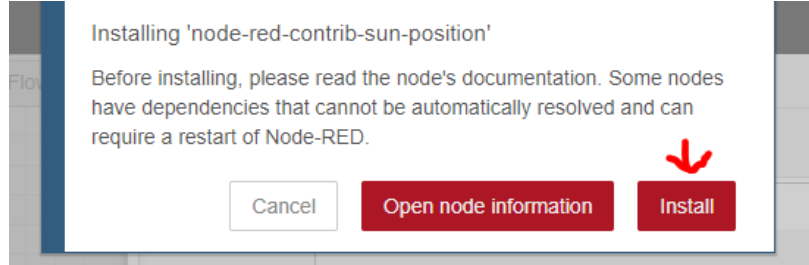


Figure 14 Node-RED confirm install

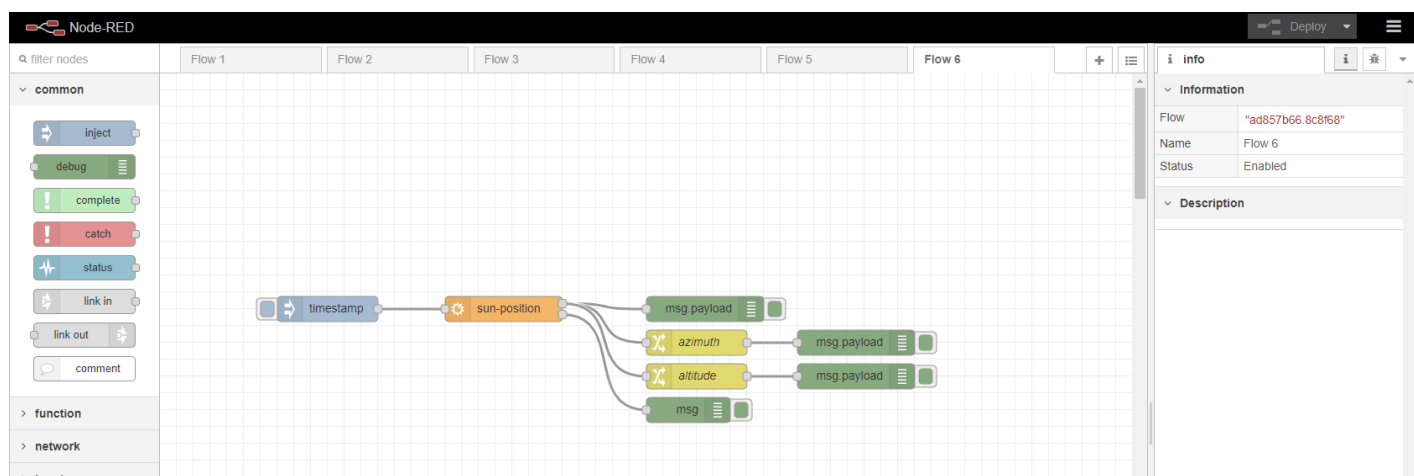


Figure 15 Node-RED success install

For the chapter 4, the following nodes will need to be installed:

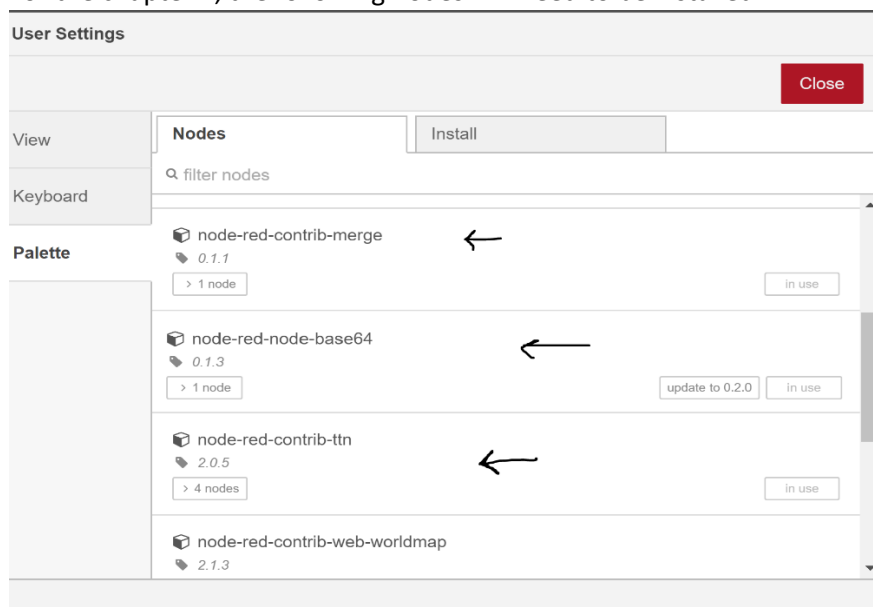


Figure 16 Node-RED installed nodes and versions

4. Importing the Carelink flows

In order to replicate the work done for connecting TTN to the Carelink platform import the provided json. For security reasons some nodes need passwords. These passwords will be in a separate file.

After importing the json file, three different flows will be placed in the Node-Red as in the figures 13-15,

The first flow is the one responsible for receiving the uplink messages from TTN, and check if there's a location in the status message. If the location from the GPS is valid, it is kept. If the location is none, the field is replaced by the location provided by the LoRa Cloud API. After this, the message is published to the "status" topic in the Carelink platform MQTT broker

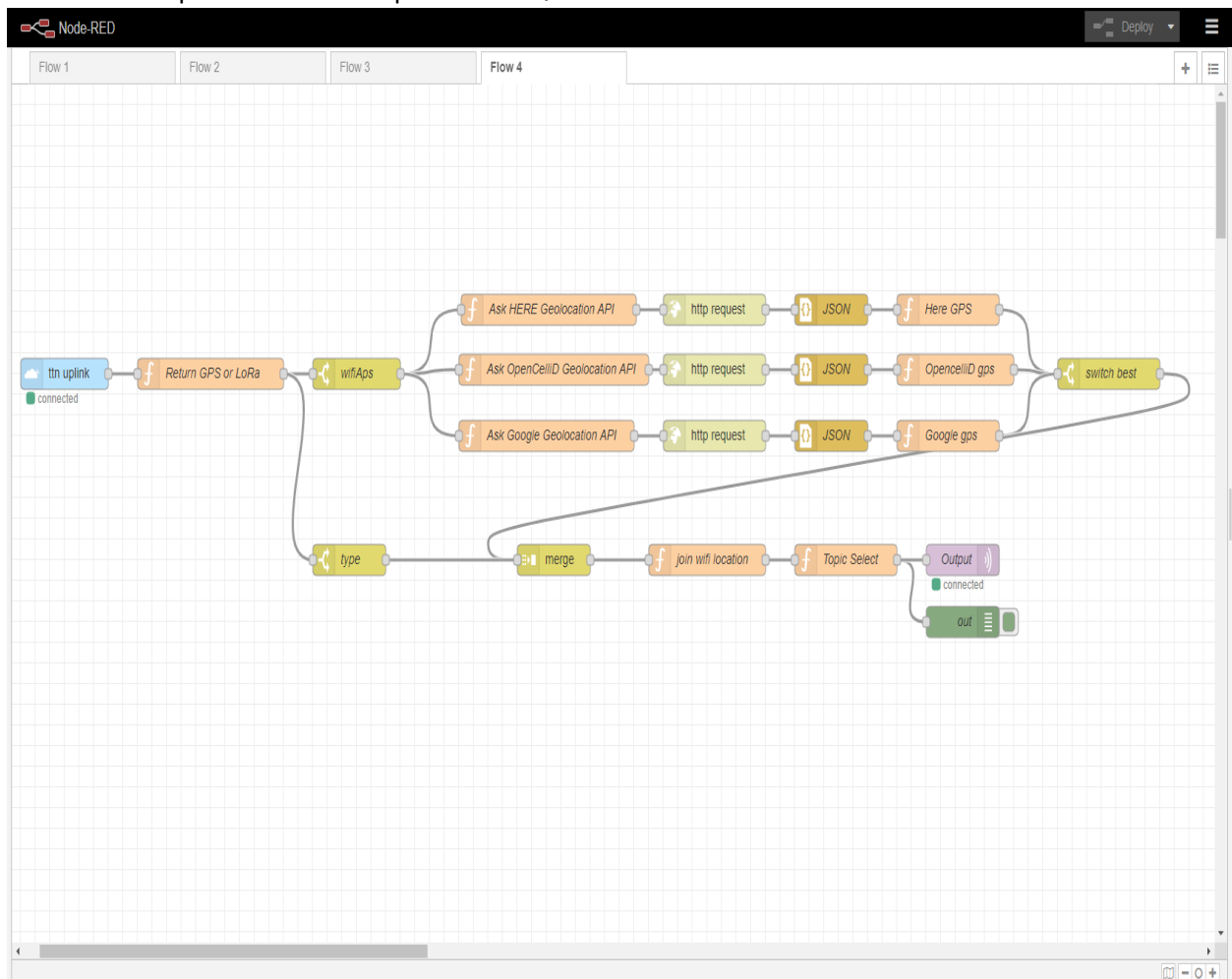


Figure 17 Node-RED flow wifi Assisted location for LoRa communication , TTN Uplink Status MQTT Downlink

The second flow is responsible for the LoRa downlink messages from the subscription of the different MQTT topics. The subscribed messages are converted to the right format so that they can be sent as downlink for TTN.

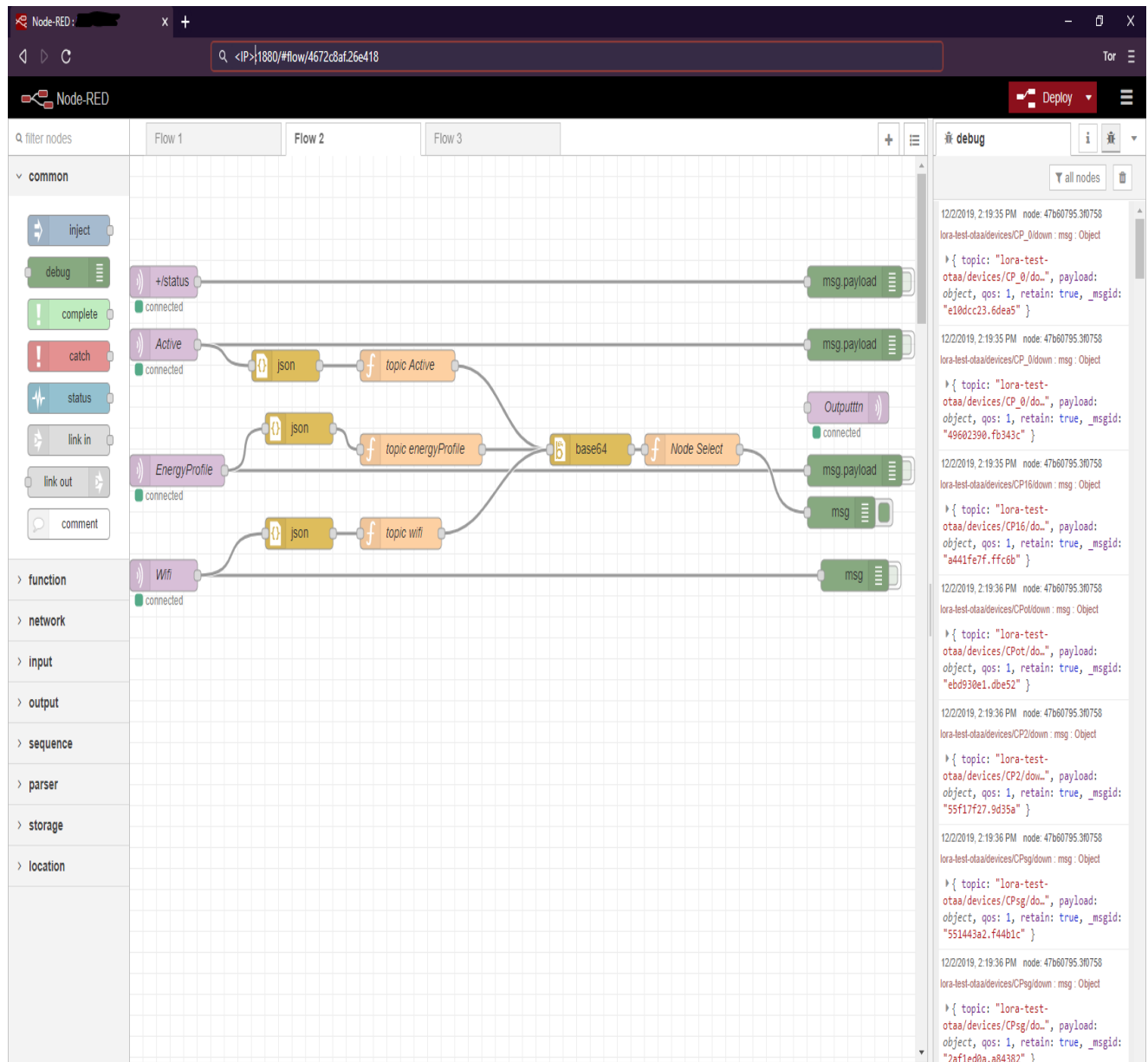


Figure 18 Node-RED flow Subscribe MQTT to Downlink TTN

The third flow is responsible for processing of the assisted location using the wifi information provided by in the "statusWifiAPs" MQTT topic, by the devices, when not using LoRa.

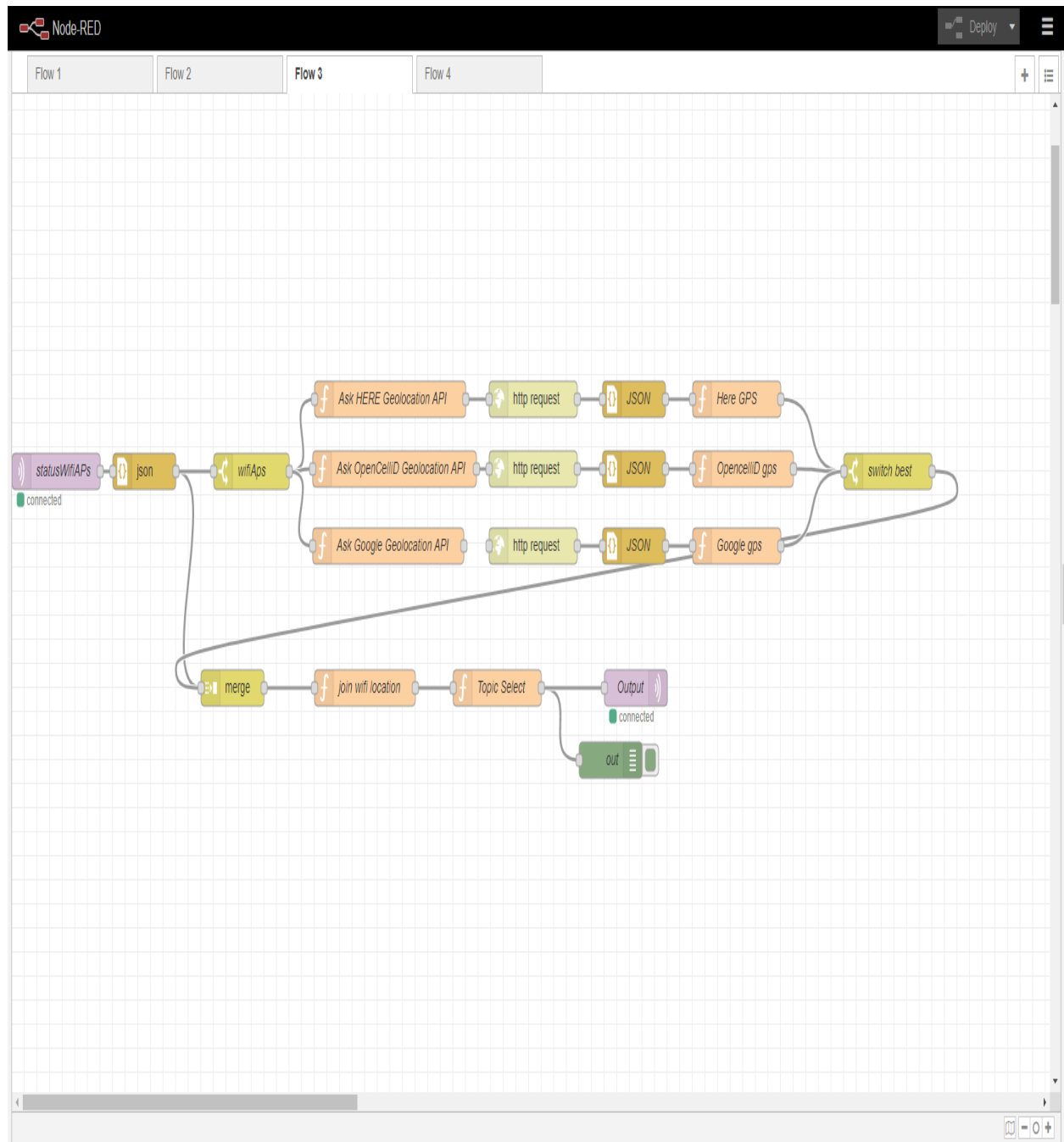


Figure 19 Node-RED flow wifi Assisted location for non-LoRa communication

5. Api's

For the flows represented by the figures 13-15, three different API's are used. These API's have the functionality to read the data from "wifiAPs" field in the json, and returning a location based in the mac addresses and rssi values.

For using the "here" API it is required an appid and appcode, for the "OpenCellID" API it is required a token and for the "google" API it is required a key.

The next step is testing API. This task is done through postman as show in the next figures.

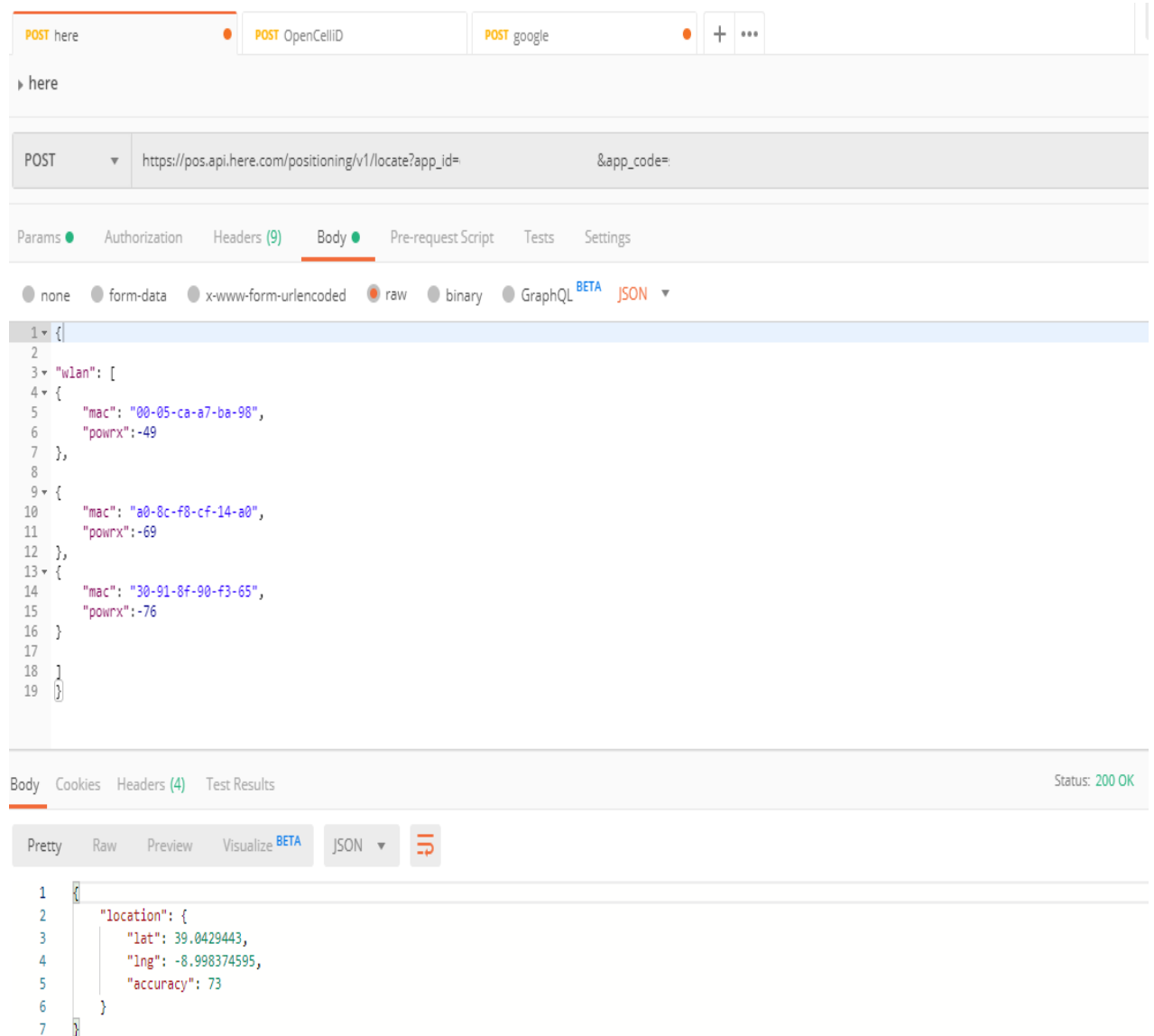


Figure 20 Postman here API test

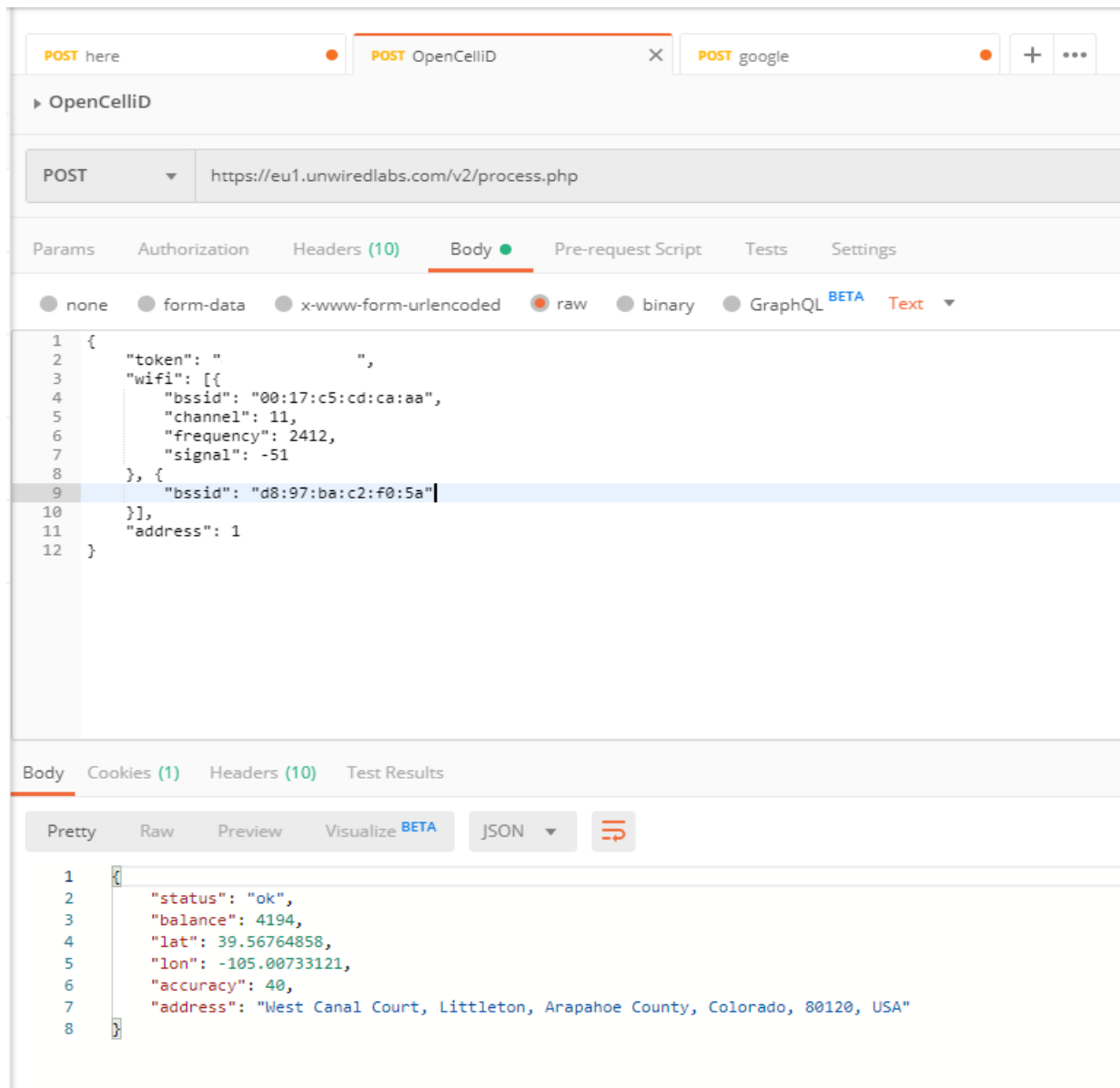


Figure 21 Postman OpenCellID API test

POST here POST OpenCellID POST google + ...

google

POST https://www.googleapis.com/geolocation/v1/geolocate?key=

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL BETA Text

```
1 {
2   "considerIp": "false",
3   "wifiAccessPoints": [
4     {
5       "macAddress": "00:25:9c:cf:1c:ac",
6       "signalStrength": -43,
7       "signalToNoiseRatio": 0
8     },
9     {
10      "macAddress": "00:25:9c:cf:1c:ad",
11      "signalStrength": -55,
12      "signalToNoiseRatio": 0
13    }
14  ]
15 }
```

body Cookies Headers (13) Test Results Status: 200 OK

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   "location": {
3     "lat": 33.3631941,
4     "lng": -117.0872285
5   },
6   "accuracy": 30
7 }
```

Figure 22 Postman Google Geolocation API test

The three APIs receive the same information but with different names, that's why in figures 13, 15 exists an "ask" function. The output is also different, thus the function "API_name gps" returns the same format regardless the used API.

All of the three got limitations as shown in the next figures.

The screenshot displays the HERE Developer Pricing page. The navigation bar at the top includes the HERE logo, 'Developer', 'Products', 'Documentation', 'Pricing' (highlighted), and 'Resources'. The main content is divided into two columns for the 'Freemium' and 'Pro' plans.

Plan	Price	Description	Sign up
Freemium	0 €/mo	Build your app, service or web map for free	Sign up
Pro	449 €/mo	Boost your limits and add support	Sign up

Service	Freemium Features	Pro Features
Location Services	<ul style="list-style-type: none">✓ 250K Transactions per month✓ 5K SDK Monthly Active Users✓ 250 Assets per month✓ Pay per additional Transactions	<ul style="list-style-type: none">✓ 1 Million Transactions per month✓ 5K SDK Monthly Active Users✓ 250 Assets per month✓ Pay per additional Transactions✓ SLA: 99.9% up-time
XYZ	<ul style="list-style-type: none">✓ 2.5GB Data transfer per month✓ 5GB Database storage per month✓ Pay per additional Data transfer or Database storage	<ul style="list-style-type: none">✓ 2.5GB Data transfer per month✓ 5GB Database storage per month✓ Pay per additional Data transfer or Database storage
Support	<ul style="list-style-type: none">✓ Stack Overflow	<ul style="list-style-type: none">✓ Support via Customer Support Portal

Figure 23 here API info

API Sandbox

API

Downloads

User Reports

User reports

Contribution reports

Account details

Support

If you wish to change any details, please [contact us here](#).

Name	Rafael Rodrigues
Company	
Account Type	Free
Satellite Maps	Not Enabled, Contact Support
Requests	5,000
Reset period	Daily
Payment period	None

Figure 24 OpenCellID API plan



Products Use Cases Pricing Contact

Hi Rafael Rodrigues! Welcome to your OpenCellID Dashboard.

The dashboard allows you to check activity levels of your account. If you need any help, we're just an [email away](#).

API Sandbox

LocationAPI will return a location if you provide Cell Towers, WIFI APs near a device. View the [API documentation here](#).

API

Downloads

User Reports

User reports

Contribution reports

Account details

Support

Logout

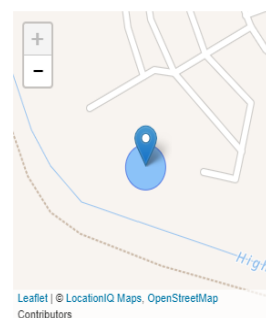
Request: 2 WiFiS

- Sample requests
- 2 WiFiS, 1 Cell
- 2 WiFiS
- IP fallback
- LAC fallback
- SCF fallback
- PSC / PCI fallback
- 1 Cell - GSM
- 1 Cell - CDMA
- 1 Cell - UMTS
- 1 Cell - LTE
- 1 Cell - Nb-Iot
- 6 Cells
- Multiple radios
- Address with details
- Per device plan

Response:

```
1 {
2   "status": "ok",
3   "balance": 100,
4   "lat": 39.56763197,
5   "lon": -105.00727917,
6   "accuracy": 10,
7   "address": "High Line Canal T
8 }
```

Location:



Submit

the tower your phone is connected to & nearby WiFi APs with an [Android app](#)

ationAPI. Reliable. Affordable. Extensive.

Figure 25 OpenCellID API functions

[Overview](#)
[Products](#)
[Pricing](#)
[Documentation](#)
[Blog](#)

Language

R

Get Started
Developer Guide
Get API Key
Best Practices Geocoding Addresses
Geocoding FAQ

Web Services
Best Practices
Client Libraries

Policies and Terms
Usage and Billing
Optimizing Quota Usage
Policies
Terms of Service

Other Web Service APIs
Directions API
Distance Matrix API
Elevation API
Geolocation API
Places API
Roads API
Time Zone API

How usage and billing work under the pay-as-you-go model

- The Google Maps Platform APIs are billed by SKU.
- Usage is tracked for each Product SKU, and an API may have more than one [Product SKU](#).
- Cost is calculated by: SKU Usage x Price per each use.
- For each billing account, for qualifying Google Maps Platform SKUs, a \$200 USD Google Maps Platform credit is available each month, and automatically applied to the qualifying SKUs.

See [guide to understanding billing](#) for more information.

Pricing for the Geocoding API

Under the pay-as-you-go pricing model, requests for the Geocoding API are billed using the SKU for Geocoding.

SKU: Geocoding

A **Geocoding** SKU is charged for requests to the [Geocoding API](#) or the [Maps JavaScript API's Geocoding service](#).

MONTHLY VOLUME RANGE (Price per REQUEST)		
0–100,000	100,001–500,000	500,000+
0.005 USD per each (<u>5.00 USD per 1000</u>)	0.004 USD per each (4.00 USD per 1000)	Contact Sales for volume pricing

Other Usage Limits

While you are no longer limited to a maximum number of requests per day (QPD), the following usage limits are still in place for the Geocoding API:

- 50 requests per second (QPS), calculated as the sum of [client-side](#) and server-side queries.

Terms of Use Restrictions

Contents

[Pay-As-You-Go Pricing](#)
[How usage and billing work under the pay-as-you-go model](#)
[Pricing for the Geocoding API](#)
[SKU: Geocoding](#)
[Other Usage Limits](#)
[Terms of Use Restrictions](#)
[Manage Your Cost of Use](#)
[Premium Plan Customers](#)

Figure 26 Google API info

Beside this three, other options were tried such as Mozilla location service, and combain, the first one is not distributing keys at this moment the second the free limit is 100 request per month as shown below.

[COVERAGE](#)
[API](#)
[INDOOR](#)
[PRICING](#)
[NEWS](#)
[JOBS](#)

It's fast and free to get started.

Within short you will have an API key to test our service in our online demo or from your own code. More than 10. signed up – we are flattered!

Company Name

Full Name

Email

Password

Portugal

The evaluation account has 100 free requests and is valid from one month. For paid accounts pricing table with more credits [click here](#). By signing up you agree to our [Terms of Use](#) and [Privacy](#).

Figure 27 combain API free tier

Credits and Pricing

By purchasing credits here, you will have access to world largest cell-id and wifi database. One request to our API is equal to one credit. A request can include just one cell, or several cells and/or wifis. If we receive multiple cells and/or wifis, we do a computation and returns the most likely position. If you need access to our premium support and customization services, please contact us for a separate quotation.

SMALL PREPAID	MEDIUM PREPAID	LARGE PREPAID	SUBSCRIPTION PLAN
25 000 Requests	100 000 Requests	500 000 Requests	1 000 000 Requests per month
100€	300€	1000€	1000€/month
Buy	Buy	Buy	Subscribe

Figure 28 combain API plans

Enter your secret api key for trying CPS API below. If you do not have one, please click [here](#) to get one for free.

YOUR_API_KEY

Request

```

1 {
2   "blueToothBeacons": [
3     {
4       "macAddress": "ca:66:1f",
5       "signalStrength": -55
6     },
7     {
8       "macAddress": "ff:74:27",
9       "signalStrength": -52
10    }
11  ]
12 }
```

BT

GSM

WCDMA

LTE

CDMA

WIFI

Indoor Wifi

BT

Indoor BT

GSM with geoname

GSM with credits

Gsm cell with neighbour

Wcdma cell with neighbour

Lte cell with neighbour

Neighbour with psc

Cdma cell with neighbour

Combined Cell and Wifi

Not Found

WCDMA w4f

WCDMA lacf

GSM lacf

Make Req

Request parameters

Mandatory URL Request Par

key - The unique key for the account,

Optional URL Request Par

id - A unique device id. Recommended

Response

```

1 {
2   "location": {
3     "lat": 55.71079,
4     "lng": 13.20664
5   },
6   "accuracy": 10
7 }
```

Location

Map

Satellite

Matematikhuset

Map data ©2019 Google Terms of Use

Figure 29 combain API functions

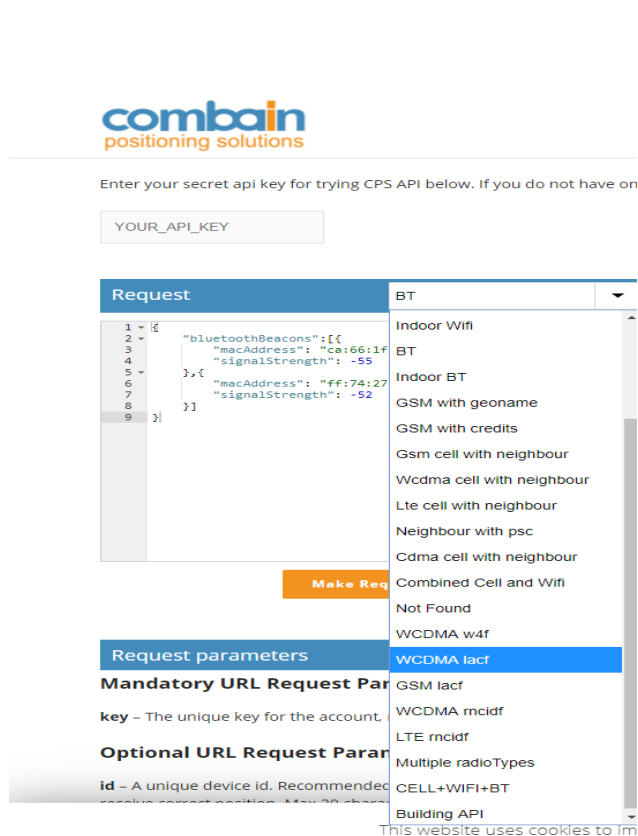


Figure 30 combain API functions

User documentation

Services API

Requesting an API Key

API Access Keys

Errors

APIs

History

Applications / Libraries

Development/Deployment documentation

Algorithms

Changelog

Glossary

Read the Docs

Requesting an API Key

The api key has a set daily usage limit of about 100,000 requests. As we aren't offering a commercial service, please note that we do not make any guarantees about the accuracy of the results or the availability of the service.

Please make sure that you actually need the raw API access to perform geolocation lookups. If you just need to get location data from your web application, you can directly use the [HTML5 API](#).

To apply for an API key, please [fill out this form](#). When filling out the form, please make sure to describe your use-case and intended use of the service. Our [Developer Terms of Service](#) govern the use of MLS API keys.

We'll try to get back to you within a few days, but depending on vacation times it might take longer.

API Access Keys

Note

Mozilla is currently evaluating its MLS service and terms and is not currently distributing API keys.

You can anonymously submit data to the service without an API key via any of the submission APIs.

You must identify your client to the service using an API key when using one of the [Region: /v1/country](#) or [Geolocate: /v1/geolocate](#) APIs.

If you want or need to specify an API key, you need to provide it as a query argument in the request URI in the form:

```
https://location.services.mozilla.com/<API>?key=<API_KEY>
```

Figure 31 Mozilla API info

References

<https://nodered.org/docs/getting-started/aws#running-on-aws-ec2-with-ubuntu>

<https://www.youtube.com/watch?v=eF5QeC-DZ2Q&t=116s>

<https://www.youtube.com/watch?v=ygapqKijVQ4>

<https://developer.here.com/>

<https://opencellid.org/>

<https://developers.google.com/maps/documentation/geolocation/get-api-key>

<https://location.services.mozilla.com/>

<https://combain.com/>