

Node-RED, InfluxDB and Grafana for IoT

Sebastian Büttrich,
IT University of Copenhagen
ICTP Trieste, 20180502
sebastian@itu.dk

Motivation

To get from

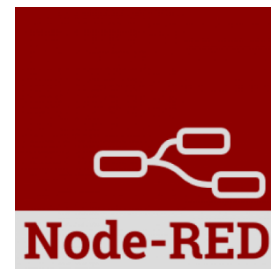
- **sensor node**

(lopy or any other)

to

- **managed data flows**

(Node-RED)



- **storage**

(InfluxDB)



- **visualization / analysis / interaction**

(Grafana)



Agenda for this talk

- **Introduce the three elements**
- **Show them in live demo**

and - for those who are hungry -

Offer you a guide on installing these yourself.

**Two things i will be using
though we will only cover them later this week:**

- **MQTT**
- **The Things Network**

MQTT is a
lean and fast
publish/subscribe
messaging protocol
running on top of **TCP/IP**
(more tomorrow!)

The Things Network /1

- **The Things Network is a global movement, a kind of “The people’s Internet of Things”.**
- Open source
- Free ... to set up and run their own, in particular:
Anyone who perpetrates a "Things Access" or a "Things Gateway" will do so **free of charge for all connecting devices and servers.**
This to some degree explains our current interest in TTN, in an educational context.

Source, Details:

<https://github.com/TheThingsNetwork/Manifest>

The Things Network /2

- **The Things Network is a global movement, a kind of “The people’s Internet of Things”.**
- It’s one possible choice when doing LoRaWan.
- It’s Open source, you are free ... to set up and run their own, in particular:

Anyone who perpetrates a "Things Access" or a "Things Gateway" will do so **free of charge for all connecting devices and servers.**

Source, Details:

<https://github.com/TheThingsNetwork/Manifest>

The Things Network /3

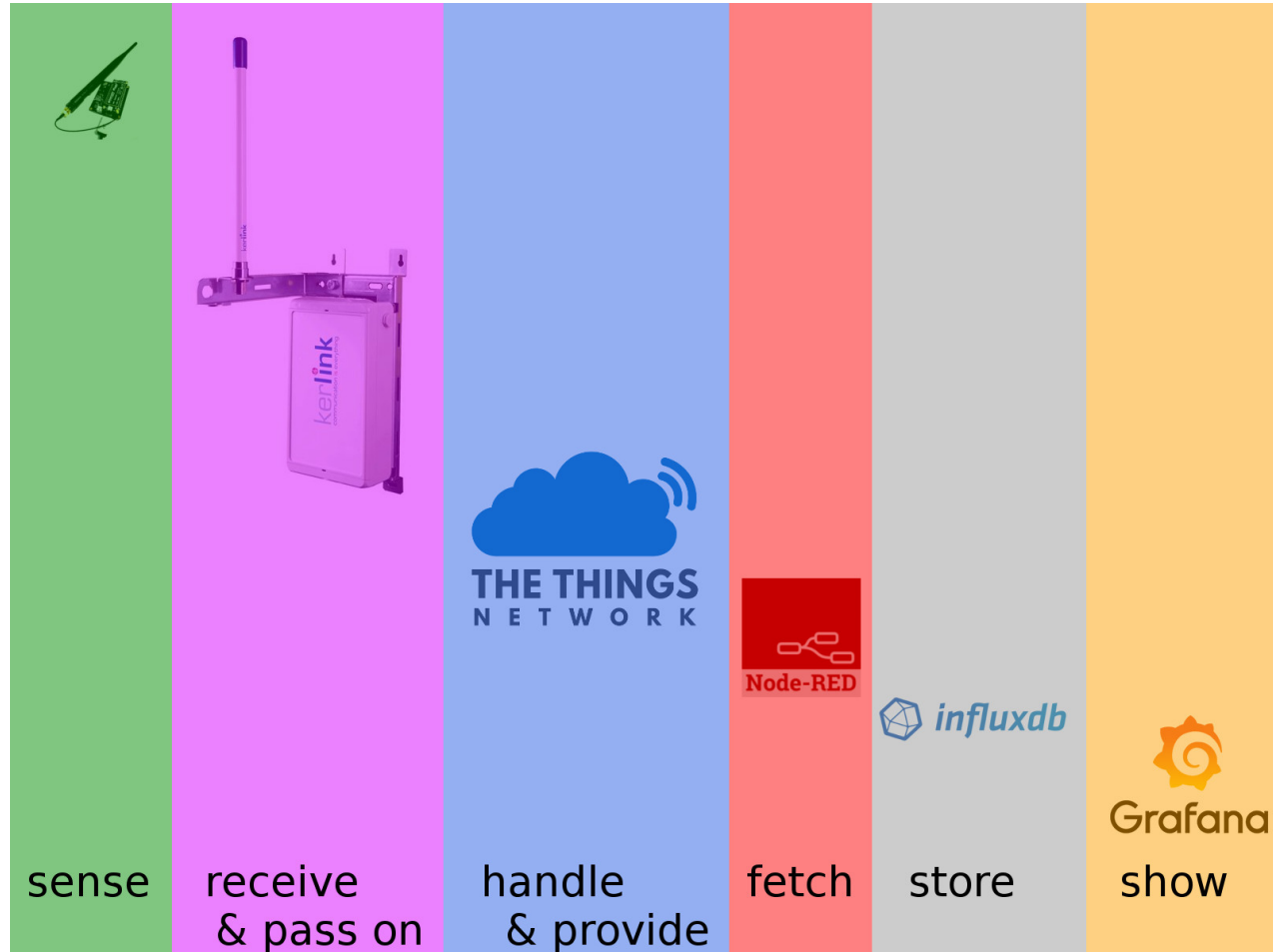


Overview /1

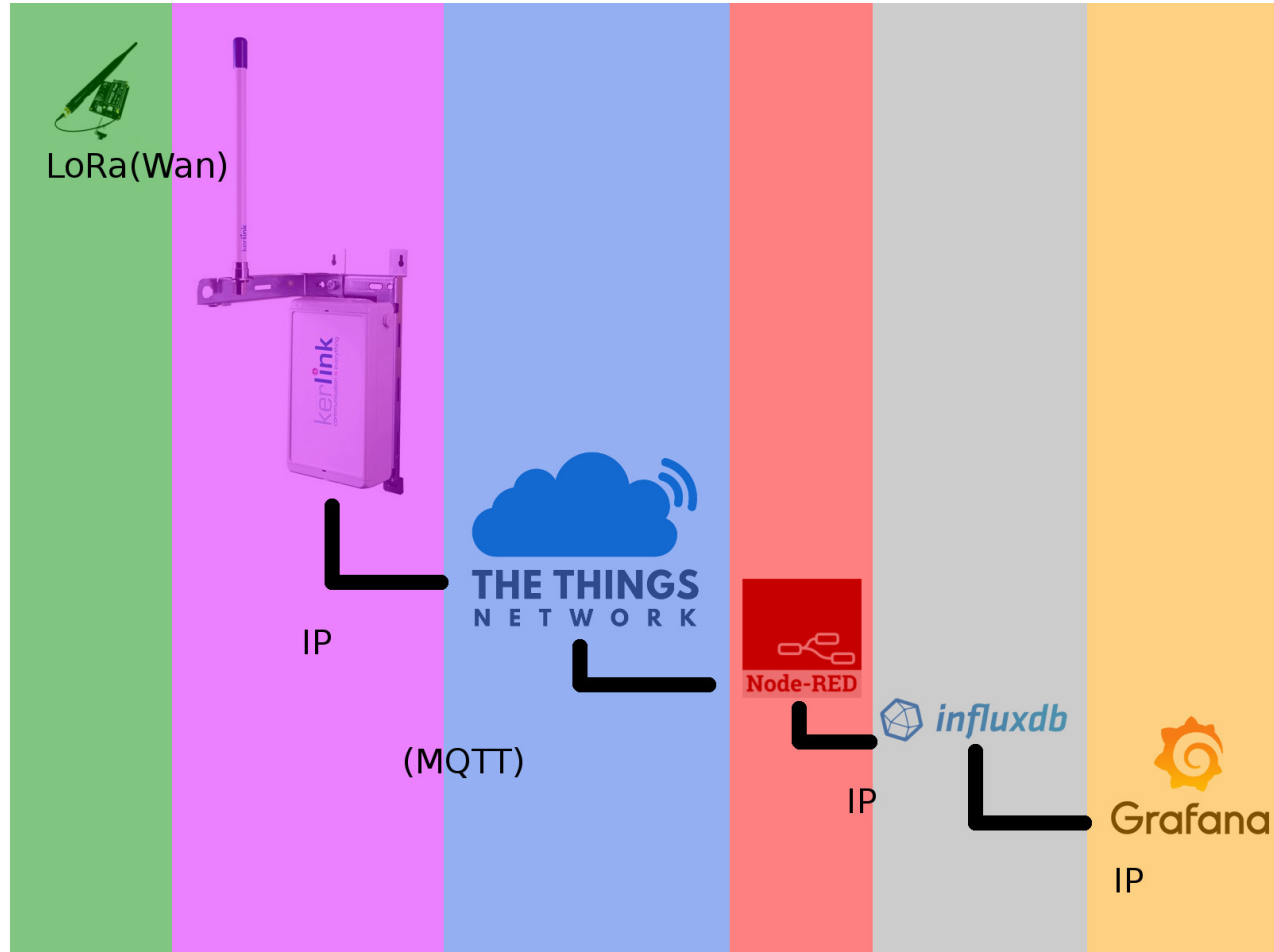


Grafana

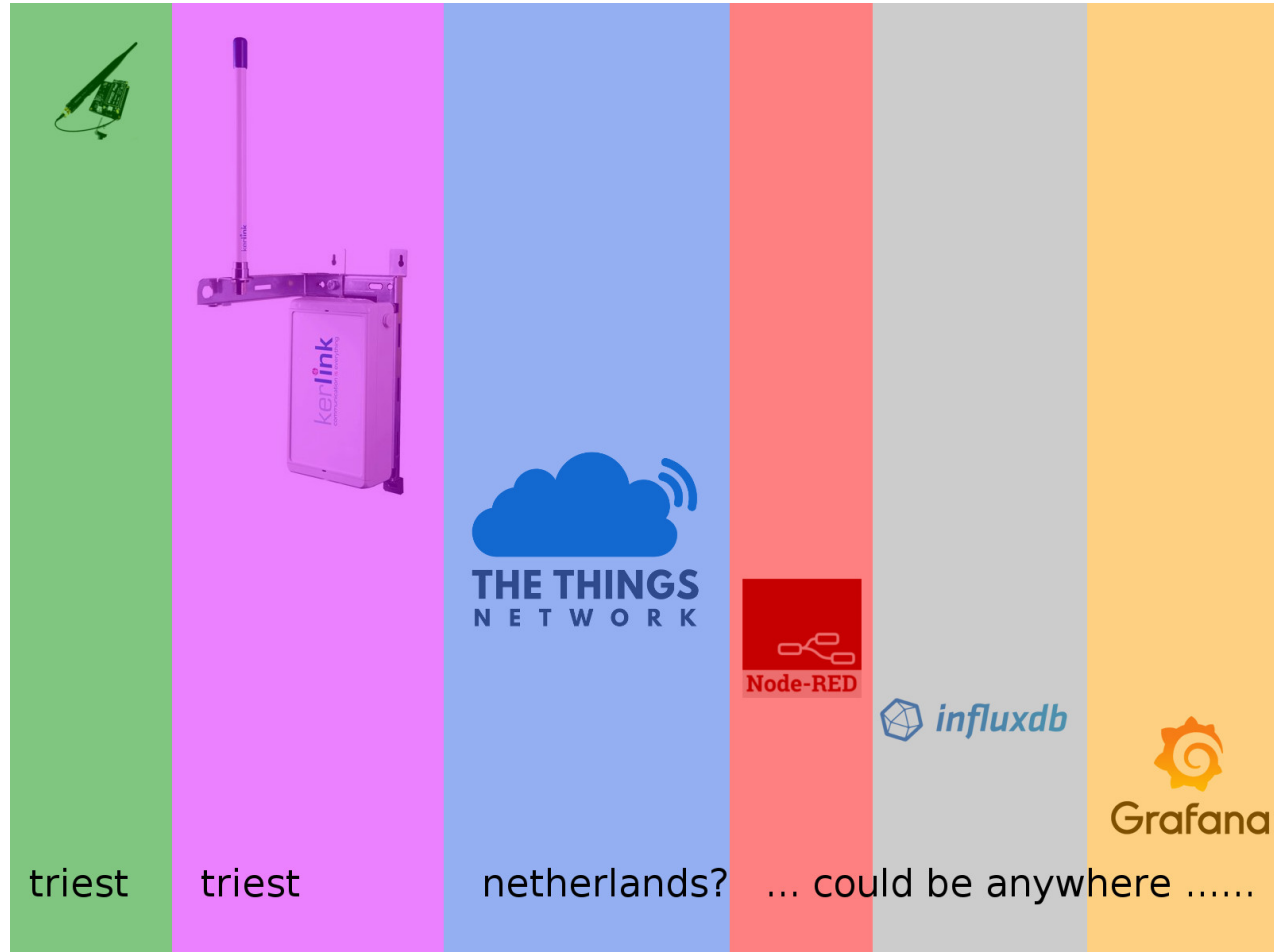
Roles



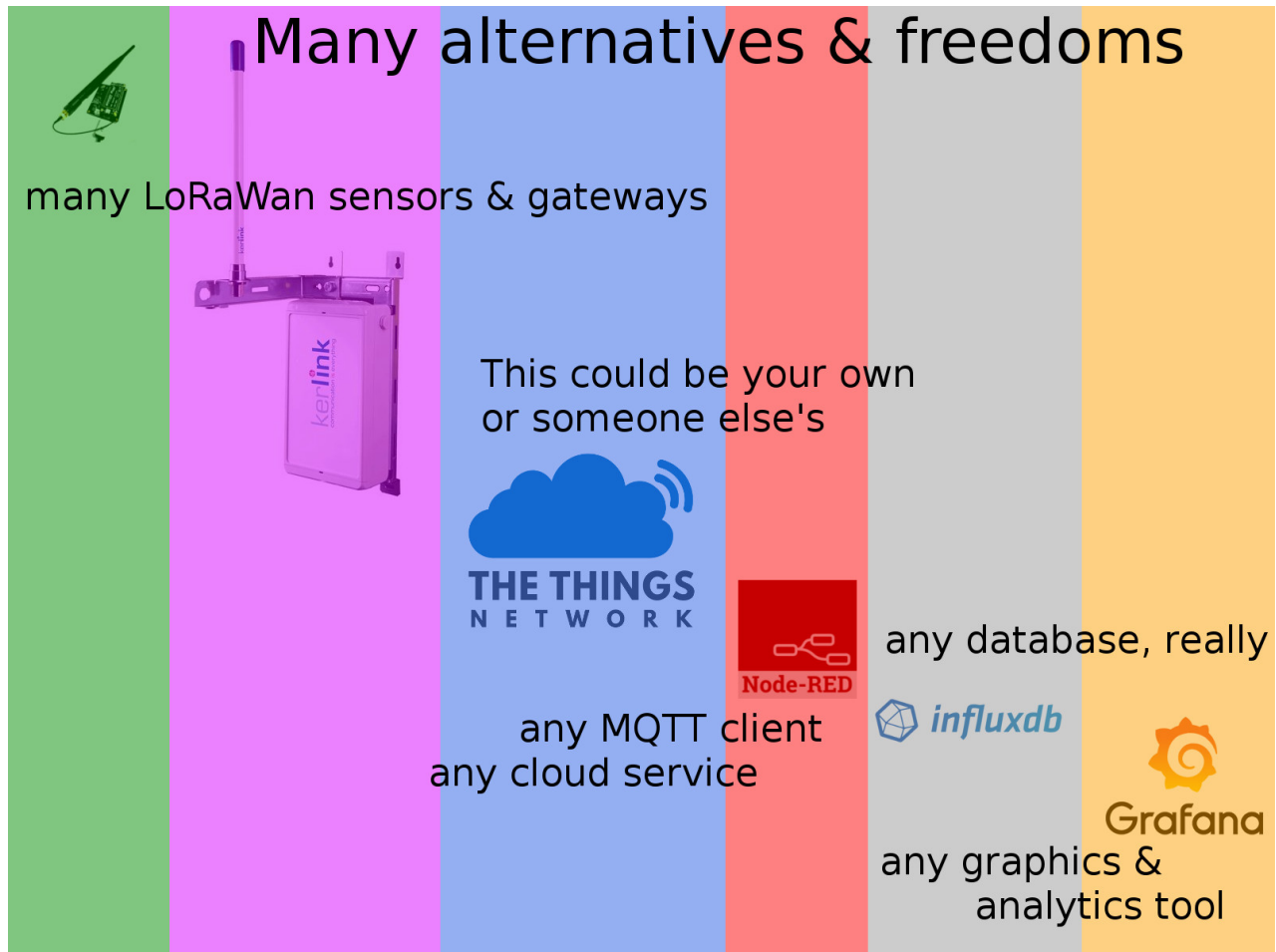
Network



Locations

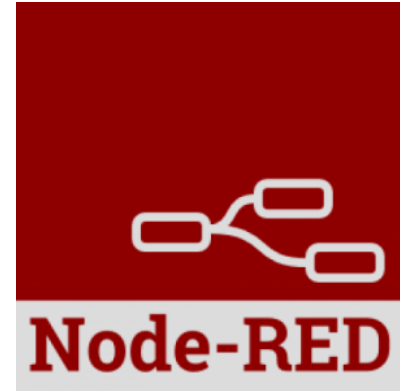


Freedom Of Choice



Node-RED /1

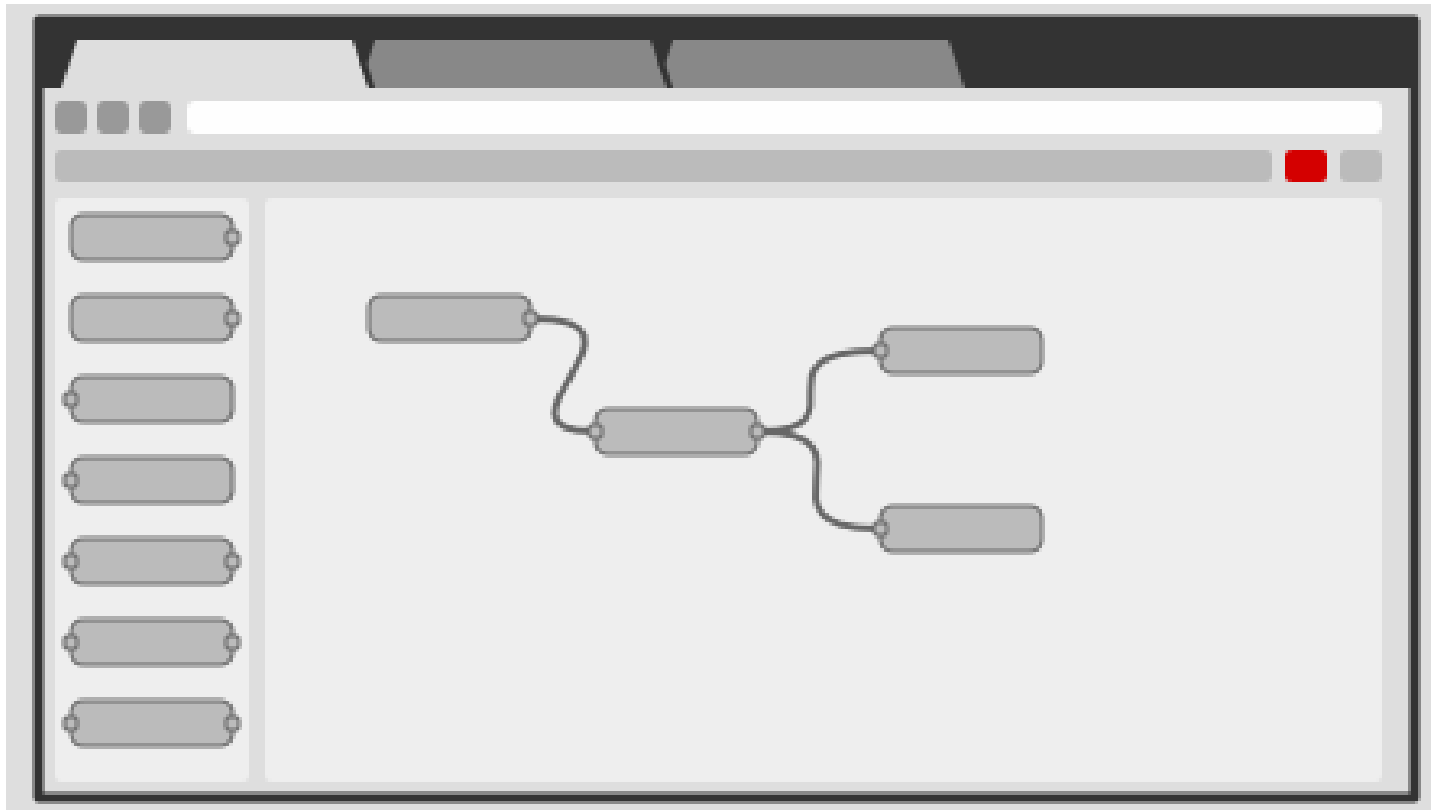
Node-RED is a
**graphical tool for wiring together
hardware devices, APIs and online services**
in new and interesting ways.



It provides a **browser-based editor** that makes it easy to wire together **flows** using the wide range of **nodes** in the **palette** that can be deployed to its runtime in a single-click.

Built on **Node.js**

Node-RED /2 principle



nodes flows

Node-RED /3 nodes

Examples of existing nodes:

Input/Output: tcp, udp, http, mqtt, ttn, ...

debug, status, inject, link, trigger

Functions: logic, analytics

Storage: e.g. databases

Social: e.g. tweet, mail

You can write your own nodes!

It is a bit like a DJs patchbay.



Node-RED /4 node config

Nodes are configured by double-clicking and editing the necessary info,

e.g.

MQTT topics,

http URLs,

TheThingsNetwork applications and security keys

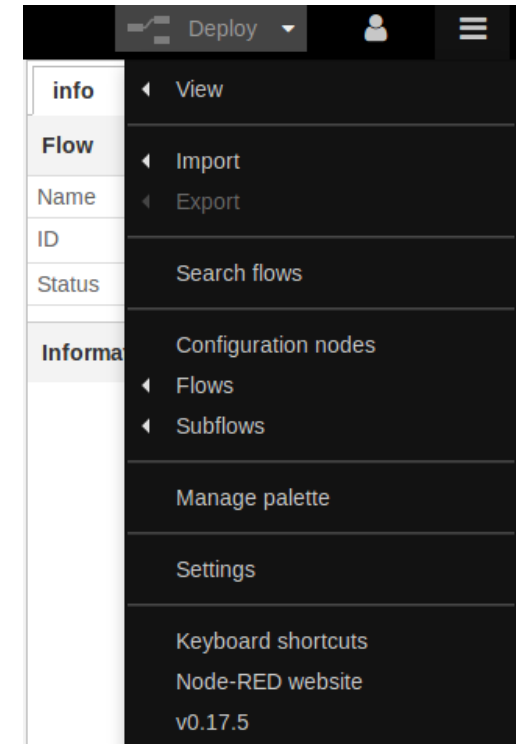
Node-RED /5 adding nodes

You may add nodes, e.g. the ttn node,
<https://flows.nodered.org/node/node-red-contrib-ttn>
via the command line, like so:

```
$ npm install node-red-contrib-ttn
```

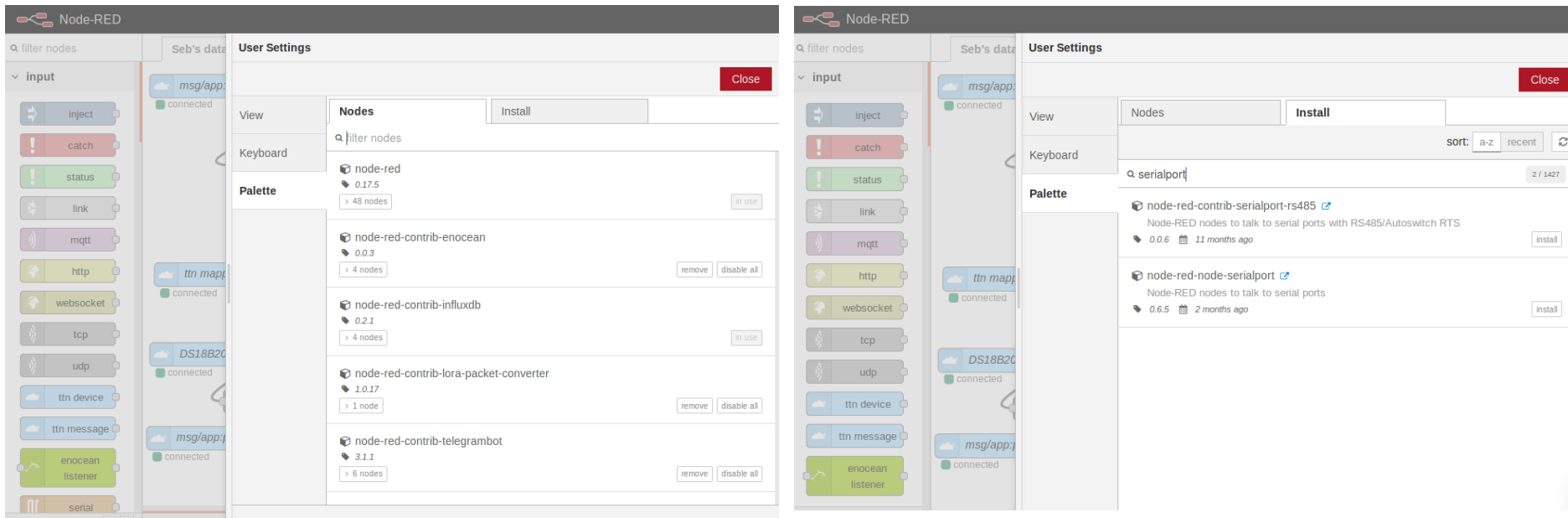
Or you can go to the node-RED menu ==>

and then ...



Node-RED /6 adding nodes

Use the palette manager to find and install new nodes, in this example,
A serial port node to read direct input over USB serial



Node-RED /7 node config ttn

Example of a TTN node:


Edit ttn device node

Delete


Cancel

Done


▼ node properties

 Name


dev/lopy-seb04

 App

pitlab-core

 Device ID

70B3D54990E351DA

 Event

up

info

debug

Node

Name	dev/lopy-seb04
Type	ttn device
ID	"e927e84.f6b3e18"

show more ▼

Information

A node to receive events from devices on The Things Network.

The application and event must be configured in the node. A device ID to filter on can also be configured.

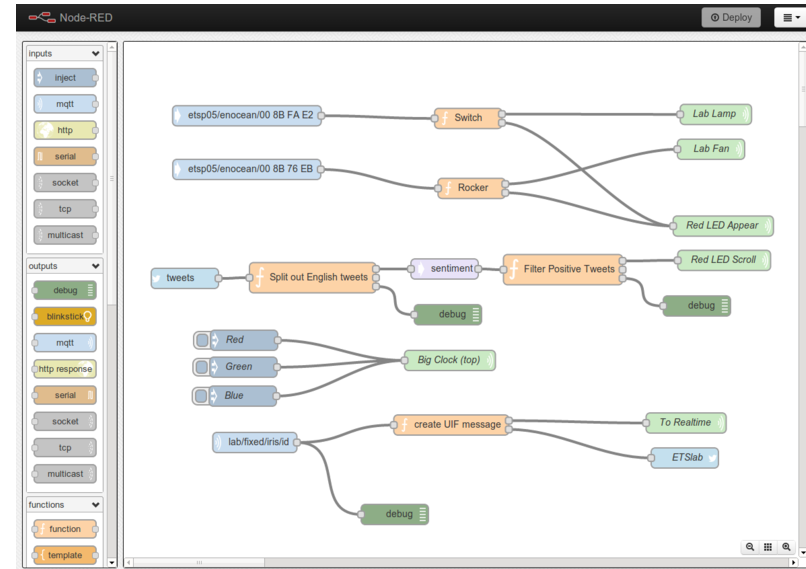
The output message:

- `dev_id`, the ID of the device that sent the message.
- `payload`, the original [MQTT events](#)

Node-RED /8 flows

Flows are combinations of inputs, outputs, connections, Actions, etc which you can share and reuse.

e.g. receive sensor messages, do calculations on the values, keep averages or deltas, put them in a database, trigger an actuator, and inform the owner via messages.



Node-RED /9 text

**While this is a graphical tool,
behind the scenes it s all text files**

```
{
  "id": "e7fb9fb.21ef46",
  "type": "tab",
  "label": "Seb's data flows",
  "disabled": false,
  "info": "",
  "id": "fe383a2e.538d78",
  "type": "ttn app",
  "z": "",
  "appId": "pitlab-ds18b20",
  "region": "eu",
  "accessKey": "ttn-account-v2.D7-uySQHbEVBj0v65hV_DI4chfWAPAC0tWTGNQoiGYno",
  "id": "7714594.09e8ca8",
  "type": "ttn app",
  "z": "",
  "appId": "pitlab-ds18b20",
  "region": "eu",
  "accessKey": "ttn-account-v2.D7-uySQHbEVBj0v65hV_DI4chfWAPAC0tWTGNQoiGYno",
  "id": "2e1a5f7.ecc9aa",
  "type": "influxdb",
  "z": "",
  "hostname": "127.0.0.1",
  "port": "8086",
  "protocol": "http",
  "database": "pit001",
  "name": "",
  "usetls": false,
  "tls": "",
  "id": "132c2fb3.7d3c",
  "type": "ttn app",
  "z": "e7fb9fb.21ef46",
  "appId": "pitlab-core",
  "region": "eu",
  "accessKey": "ttn-account-v2.XLXDZg4cTd0X-4obr4F3ASStKGLt49VeqBMQL-QIRo",
  "id": "c433206c.cb88b",
  "type": "mqtt-broker",
  "z": "",
  "broker": "influx.itu.dk",
  "port": "1883",
  "clientId": "",
  "usetls": false,
  "compatmode": true,
  "keepalive": "60",
  "cleansession": true,
  "willTopic": "",
  "willQos": "0",
  "willPayload": "",
  "id": "7dd9c9c.698d7c",
  "type": "ttn app",
  "z": "",
  "appId": "pitlab-test-seb-201804",
  "region": "eu",
  "accessKey": "ttn-account-v2.uNQLci02r441nJVyb9fgkVPEgNllj80wN8tzY8Lrng",
  "id": "bbe92aff.b8c38",
  "type": "ttn app",
  "z": "",
  "appId": "pitlab-seb-temperature-humidity",
  "region": "eu",
  "accessKey": "ttn-account-v2.d1HGrFbctUYngVrcXcO_D8V2Qw4meJ_M7VC4B9qk5Kg",
  "id": "ebf65f2f.3c1748",
  "type": "ttn app",
  "z": "",
  "appId": "dk-cph-itu-pitlab-01",
  "region": "eu",
  "accessKey": "ttn-account-v2.2ohV653kNiCAOPsm0F9u9KI8nAjJP3Mq62Ywv6FyZdM",
  "id": "e927e84.f6b3e18",
  "type": "ttn device",
  "z": "e7fb9fb.21ef46",
  "name": "dev/lopy-seb04",
  "app": "4e72e8d2.31e58",
  "devId": "70B3D54990E351DA",
  "event": "up",
  "x": "96.53570556640625",
  "y": "715.321533203125",
  "wires": [
    [
      "71231f80.fdba38"
    ]
  ],
  "id": "6e482785.6e1468",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "",
  "retentionPolicy": "",
  "x": "1180",
  "y": "40",
  "wires": [
    [
      "d6ac24de.c86878"
    ]
  ],
  "id": "d6ac24de.c86878",
  "type": "ttn message",
  "z": "e7fb9fb.21ef46",
  "name": "msg/app:dk-cph-itu-pitlab-01",
  "app": "ebf65f2f.3c1748",
  "devId": "field",
  "x": "129",
  "y": "31",
  "wires": [
    [
      "5ff3781c.a6152",
      "2d37e02c.830c28",
      "1db6ebe3.327dec"
    ]
  ],
  "id": "2d37e02c.830c28",
  "type": "function",
  "z": "e7fb9fb.21ef46",
  "name": "function 3 - 3 sticks and waterTemp",
  "func": "node.log (\\function 3 called\\)\\nvar msgAll = { payload: msg.payload;\\nvar msg0 = { payload: msg.payload[0] };\\nvar msg1 = { payload: msg.payload[1] };\\nvar msg2 = { payload: msg.payload[2] };\\nvar msg3 = { payload: msg.payload[3] };\\nvar msg4 = { payload: msg.payload[4] };\\nvar msg5 = { payload: msg.payload[5] };\\nvar msg6 = { payload: msg.payload[6] };\\nvar msg7 = { payload: msg.payload[7] };\\nvar msg8 = { payload: msg.payload[8] };\\nvar msg9 = { payload: msg.payload[9] };\\nvar msg10 = { payload: msg.payload[10] };\\nvar msg11 = { payload: msg.payload[11] };\\nvar msg12 = { payload: msg.payload[12] };\\nreturn [ msg0, msg1, msg2, msg3, msg4, msg5, msg6, msg7, msg8, msg9, msg10, msg11, msg12 ];\\n",
  "outputs": "13",
  "noerr": "0",
  "x": "462.857177734375",
  "y": "145.71426391601562",
  "wires": [
    [
      "195b976d.dd29e9",
      "6f89d82d.af84b8",
      "12d1fc81.2209a3",
      "62791b96.df9c84",
      "863f73e.2f8189",
      "10afd128.2aabcf",
      "a23ae5fd.b2d2c8",
      "5e7d504e.3905e8"
    ]
  ],
  "id": "195b976d.dd29e9",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "lopySeb02",
  "retentionPolicy": "",
  "x": "1180",
  "y": "80",
  "wires": [
    [
      "5ff3781c.a6152"
    ]
  ],
  "id": "5ff3781c.a6152",
  "type": "function",
  "z": "e7fb9fb.21ef46",
  "name": "function 04",
  "func": "node.log (\\function 04 called\\)\\nvar msg1 = { payload: msg.payload[2];\\nreturn [ msg1 ];\\n",
  "outputs": "1",
  "noerr": "0",
  "x": "390.0000305175781",
  "y": "24.28571319580078",
  "wires": [
    [
      "f6162812.22b968"
    ]
  ],
  "id": "f6162812.22b968",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "003-01-moist",
  "retentionPolicy": "",
  "x": "1272",
  "y": "182",
  "wires": [
    [
      "109102f9.a35295"
    ]
  ],
  "id": "109102f9.a35295",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "003-01-light",
  "retentionPolicy": "",
  "x": "1257",
  "y": "224",
  "wires": [
    [
      "f4314019.a81c58"
    ]
  ],
  "id": "f4314019.a81c58",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "003-05-temp",
  "retentionPolicy": "",
  "x": "1264",
  "y": "622",
  "wires": [
    [
      "12d1fc81.2209a3"
    ]
  ],
  "id": "12d1fc81.2209a3",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "003-02-light",
  "retentionPolicy": "",
  "x": "1257",
  "y": "344",
  "wires": [
    [
      "6f89d82d.af84b8"
    ]
  ],
  "id": "6f89d82d.af84b8",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "003-02-moist",
  "retentionPolicy": "",
  "x": "1264",
  "y": "392",
  "wires": [
    [
      "195b976d.dd29e9"
    ]
  ],
  "id": "195b976d.dd29e9",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "003-02-temp",
  "retentionPolicy": "",
  "x": "1267",
  "y": "264",
  "wires": [
    [
      "10afd128.2aabcf"
    ]
  ],
  "id": "10afd128.2aabcf",
  "type": "influxdb",
  "z": "e7fb9fb.21ef46",
  "influxdb": "2e1a5f7.ecc9aa",
  "name": "measurement",
  "precision": "003-03-light",
  "retentionPolicy": "",
  "x": "1257",
  "y": "464",
  "wires": [
    [
      "9c3e73e.2f8189"
    ]
  ],
  "id": "9c3e73e.2f8189",
  "type": "influxdb"
}
```

which you can access via shell, edit, export, import, ...

InfluxDB /1

InfluxDB is an **open-source time series database** developed by InfluxData.

It is written in **Go** and optimized for fast, high-availability storage and retrieval of time series data in fields such as operations monitoring, application metrics, **Internet of Things sensor data**, and real-time analytics.

Open Source: Each component (except for some ...) of the InfluxData platform or TICK Stack is on github and available via a simple download.

The company Influxdata is venture funded.



SQL-like language with builtin **time-centric** functions for querying a data structure composed of measurements, series, and points.

Each point consists of several **key-value pairs** called the fieldset and a **timestamp**. When grouped together by a set of key-value pairs called the tagset, these define a series. Finally, series are grouped together by a string identifier to form a **measurement**.

```
measurement(,tag_key=tag_val)* field_key=field_val(,field_key_n=field_value_n)*  
(nanoseconds-timestamp)
```

Values can be 64-bit integers, 64-bit floating points, strings, and booleans.

Main differences to a classical SQL database or NoSQL database:

It s for **time series** - nothing else!

A "table" (called measurement here) has 2 “columns”, not more *:
a **timestamp** and the **value** for that point in time.

(You can add optional “tags” to create some structure).

You would not keep an address database or such in InfluxDB.

* it s actually possible, but not advised to do so: <https://stackoverflow.com/questions/45368535/influxdb-single-or-multiple-measurement#45545405>

InfluxDB /4 look inside: the shell: show databases

```
InfluxDB shell 0.10.0
```

```
> show databases
```

```
name: databases
```

```
-----
```

```
name
```

```
_internal
```

```
pit001
```

```
pit002
```

```
pit003
```

InfluxDB /5 look inside: the shell: show measurements

```
> use pit001
```

```
Using database pit001
```

```
> show measurements
```

```
name: measurements
```

```
-----
```

```
name
```

```
003-01-light
```

```
003-01-moist
```

```
003-01-temp
```

```
003-02-light
```

InfluxDB /6 look inside: the shell: select

```
> select * from "004-01-temp"
```

```
1525096049108783758 50
```

```
1525096124247923458 49
```

```
1525096199476486703 49
```

```
1525096274435089220 50
```

```
1525096349502370874 50
```

```
1525096424982100010 50
```

```
1525096499794455963 49
```

InfluxDB /6 look inside: the shell: create database

```
> CREATE DATABASE MyNewDatabase
```

```
>
```

InfluxDB /6 look inside: the shell

```
> select * from "004-01-temp"
```

```
1525096049108783758 50
```

```
1525096124247923458 49
```

```
1525096199476486703 49
```

```
1525096274435089220 50
```

```
1525096349502370874 50
```

```
1525096424982100010 50
```

```
1525096499794455963 49
```

InfluxDB /7 access

Typically, you would access data via network rather than direct -

InfluxDB accepts data via **HTTP, TCP, and UDP**.

For example you could connect from

Visualizing, Monitoring, Analyzing, Querying, Alerting



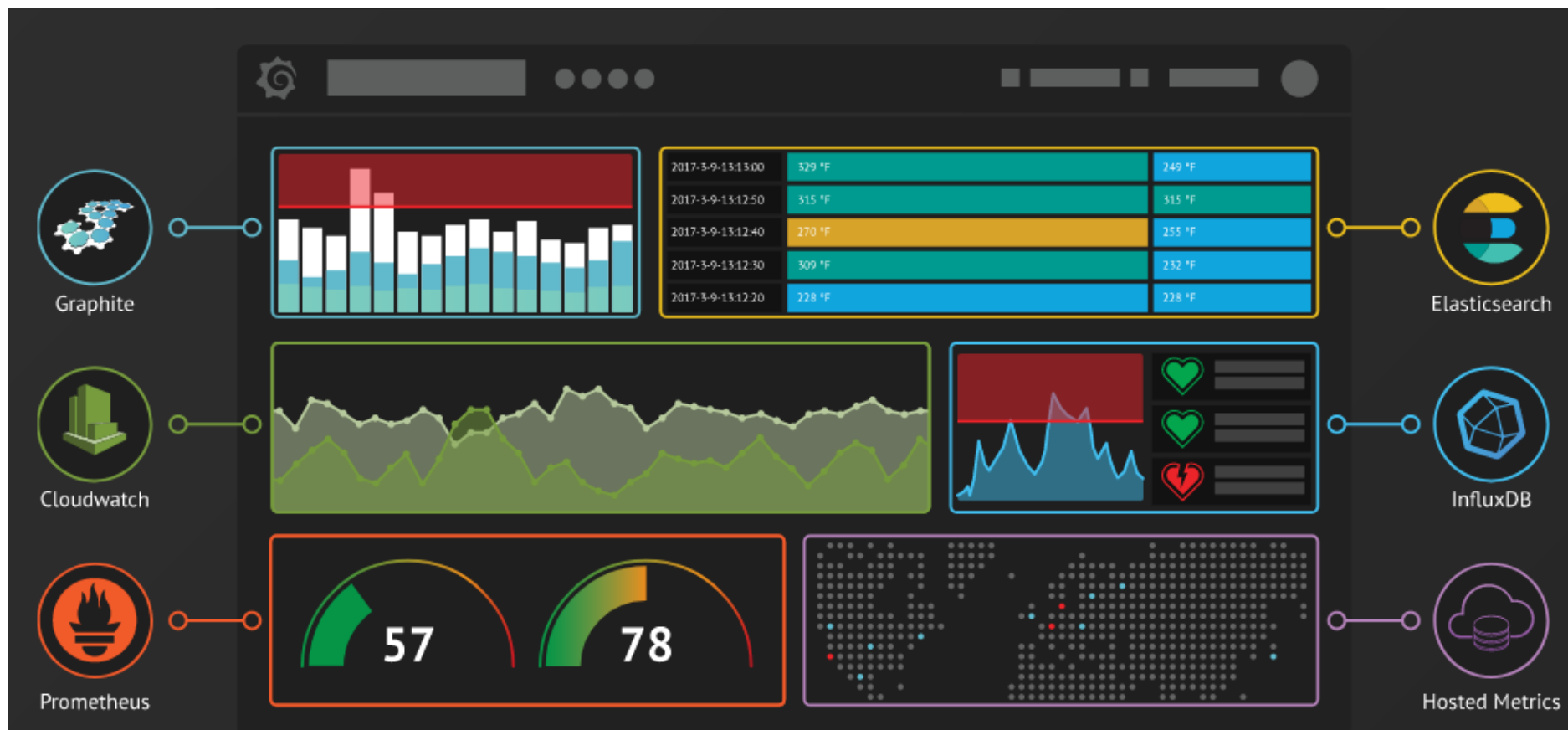
Grafana

As of right now, there are

41 data sources, 30 panels, 17 apps and 857 dashboards available.

Data sources include: influxDB, MySql, Postgres, Azure, ...

Grafana /2



Dashboards, Graphs, Queries:

Setting up a new one is mainly click, drag and drop.

The screenshot shows the Grafana dashboard creation interface. On the left, a sidebar contains icons for settings, a plus sign, a dashboard grid, a bell, and a gear. The main area is titled 'New dashboard' and features a 'New Panel' button with 'Add' and 'Paste' sub-buttons. Below this is a 'Panel Search Filter' input field. A grid of panel types is displayed: Graph, Singlestat, Table, Text, Heatmap, Alert List, Dashboard list, Row, and Plugin list. An inset window titled 'Graph' is open, showing tabs for 'General', 'Metrics', 'Axes', 'Legend', 'Display', 'Alert', and 'Time range'. The 'Metrics' tab is active, showing a 'Data Source' dropdown set to 'default' and a query editor with the query: `SELECT "value" FROM "008-01-temp"`.

Alerts

let you define flexible alert conditions

and
may trigger

emails,
telegram messages,
slack messages, ...

Alert Config

Name	DS18B20 - temp only - in halfCelsius :) al...	Evaluate every	60s
------	---	----------------	-----

Conditions

WHEN	last ()	OF	query (A, 10s, now)	IS ABOVE	60	
------	---------	----	---------------------	----------	----	--

Edit Notification Channel

Name	sebAlert
Type	Telegram
Send on all alerts	<input checked="" type="checkbox"/>
Include image	<input checked="" type="checkbox"/>

Telegram API settings

BOT API Token	539
Chat ID	923

Node-RED, Grafana & Influx demo

Installing your own

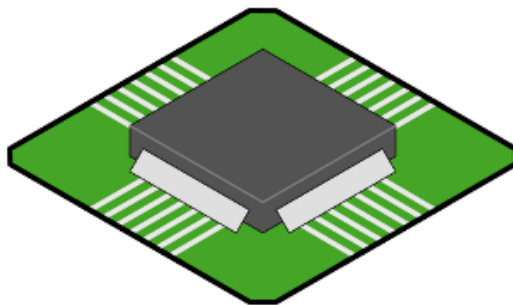
Get Started

Node-RED is built on Node.js, 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.



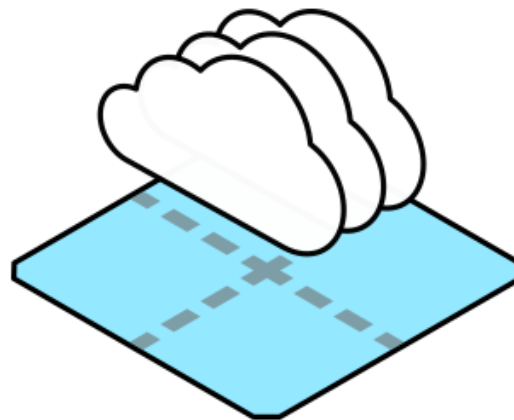
Run locally

- Getting started
- Docker



On a device

- Raspberry Pi
- BeagleBone Black
- Interacting with Arduino
- Android



In the cloud

- IBM Bluemix
- SenseTecnica FRED
- Amazon Web Services
- Microsoft Azure

Installing your own

- While a server is the preferred place for these, you can do a demo on your laptop or on a Raspberry Pi or such.
- Our guide is at
<https://github.com/ITU-PITLab/public/blob/master/TheThingsNetwork+node-red+influxdb+grafana.md>
It mainly links to guides on the websites of TTN, Node-RED, influxdb and Grafana, and adds a few tips and hints.
- It is targeted at Debian/Ubuntu – installation on other operating systems is a bit different – check this and the links therein:
<https://www.thethingsnetwork.org/labs/story/store-and-visualize-data-using-influxdb-and-grafana>
and use your search engine, e.g. with query like “Influxdb installation Windows”

Installing your own

For the impatient, here s the short version:

```
# Installation / commands summary (Debian/Ubuntu)
```

```
// Node-RED
```

```
# apt-get install nodejs    (<<< or nodejs-legacy)
```

```
# nodejs -v                ( or: node -v)
```

```
# apt install npm
```

```
# npm install -g --unsafe-perm node-red
```

```
(start with)
```

```
# node-red
```

```
// Influx
```

```
# apt-get install influxdb influxdb-client
```

```
(test with)
```

```
# influx
```

```
// Grafana
```

```
# wget https://s3-us-west-2.amazonaws.com/grafana-releases/release/grafana_5.1.0_amd64.deb
```

```
# sudo dpkg -i grafana_5.1.0_amd64.deb
```

```
# service grafana-server start
```

Thank you!

- **Contact me via**
sebastian@itu.dk

