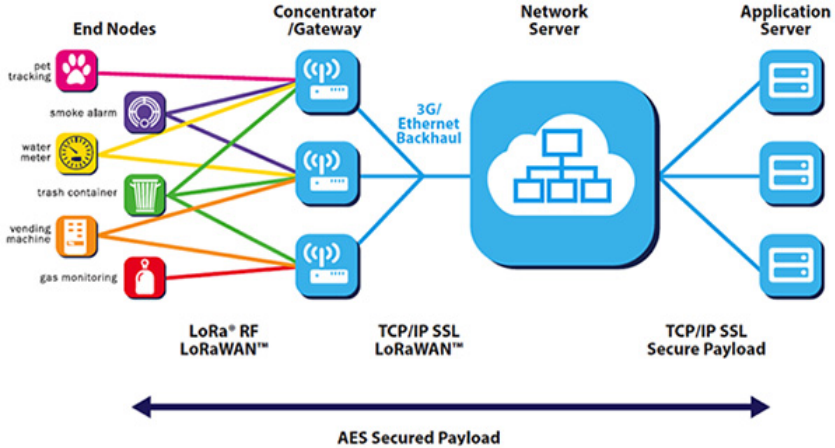
DCRPi:
data center on a Raspberry Pi

Marco Zennaro, PhD
ICTP

LoRaWAN architecture

MQTT Broker!



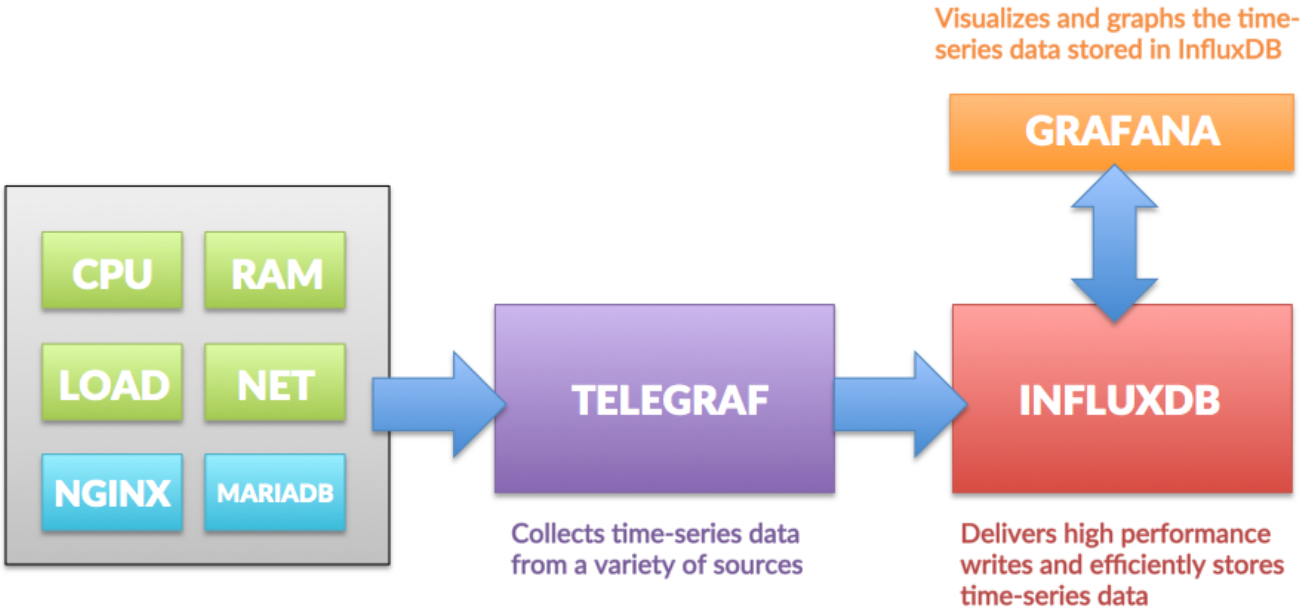
What is the TIG Stack?

The **TIG Stack** is an acronym for a platform of open source tools built to make collection, storage, graphing, and alerting on **time series data** incredibly easy.

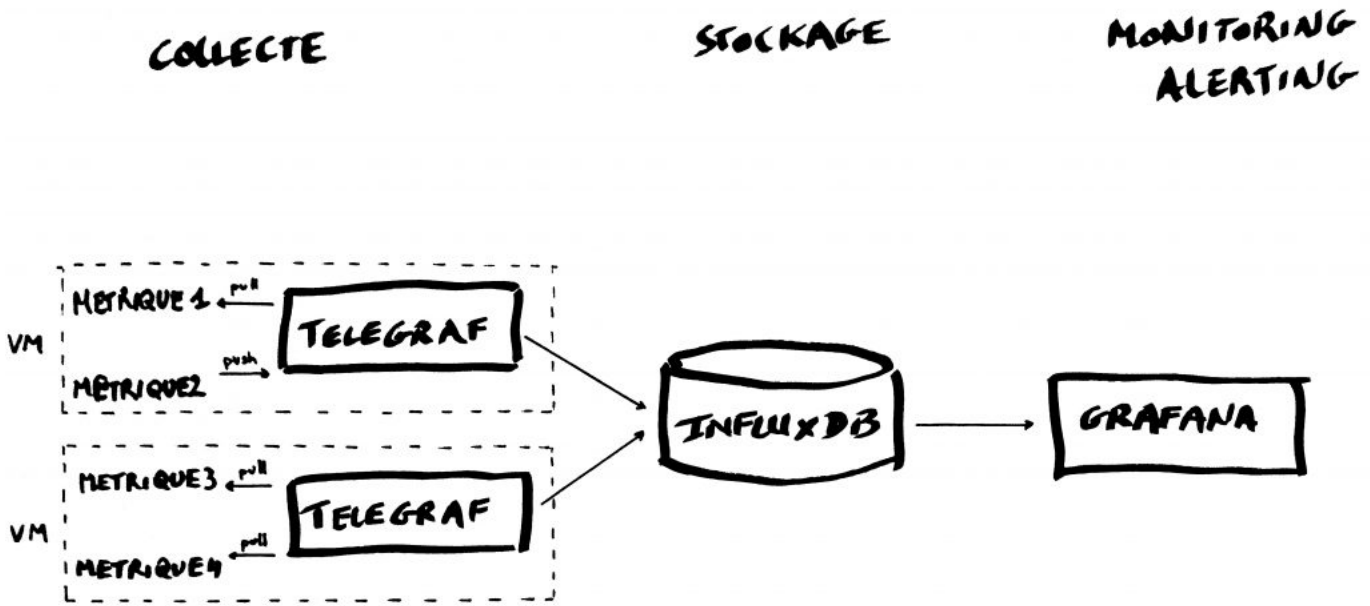
What is a time series?

A time series is simply any set of values with a timestamp where time is a meaningful component of the data. The classic real world example of a time series is stock currency exchange price data

What is the TIG Stack?



What is the TIG Stack?



What is the TIG Stack?

Telegraf is a metrics collection agent. Use it to collect and send metrics to InfluxDB. Telegraf's plugin architecture supports collection of metrics from 100+ popular services right out of the box.

InfluxDB is a high performance Time Series Database. It can store hundreds of thousands of points per second. The InfluxDB SQL-like query language was built specifically for time series.

What is the TIG Stack?

Grafana is an open-source platform for data visualization, monitoring and analysis. In Grafana, users can to create dashboards with panels, each representing specific metrics over a set time-frame. Grafana supports graph, table, heatmap and free text panels.

Installing TIG on a Linux machine

For Pi users, follow these commands:

curl -sL https://repos.influxdata.com/influxdb.key | sudo apt-key add -

echo "deb https://repos.influxdata.com/debian stretch stable" | sudo tee /etc/apt/sources.list.d/influxdb.list

sudo apt-get update

Installing TIG on a Linux machine

We can now install Telegraf and Influxdb:

sudo apt-get update

sudo apt-get install telegraf

sudo apt-get install influxdb

Installing TIG on a Linux machine

Starting from v5.2.0-beta1 Grafana introduced official support for arm64 linux platforms.

Raspberry:

sudo wget https://dl.grafana.com/oss/release/grafana-rpi_6.2.2_armhf.deb

sudo dpkg -i grafana-rpi_6.2.2_armhf.deb

For Ubuntu install it with:

wget https://dl.grafana.com/oss/release/grafana_6.2.5_amd64.deb

sudo dpkg -i grafana_6.2.5_amd64.deb

Installing TIG

We can now activate all the services:

```
sudo systemctl enable influxdb

sudo systemctl start influxdb

sudo systemctl enable telegraf

sudo systemctl start telegraf

sudo systemctl enable grafana-server

sudo systemctl start grafana-server
```

Getting started with InfluxDB

InfluxDB is a time-series database compatible with SQL, so we can setup a database and a user easily. You can launch its shell with the *influx* command.

```
pi@raspberrypi:~ $ influx
```

Creating a database

Next step is creating a database. Choose your name!

> CREATE DATABASE database_name

> SHOW DATABASES

name: databases

name

_internal

database_name

Retention Policy

A Retention Policy (RP) is the part of InfluxDB's data structure that describes for how long InfluxDB keeps data.

InfluxDB compares your local server's timestamp to the timestamps on your data and **deletes data that are older than the RP's DURATION.** A single database can have several RPs and RPs are unique per database.

Retention Policy

> CREATE RETENTION POLICY thirty_days ON database_name DURATION 30d REPLICATION 1 DEFAULT

> SHOW RETENTION POLICIES ON database_name

thirty_days 720h0m0s 1 TRUE

> exit

Configuring Telegraf

Next, we have to configure the Telegraf instance to read from the TTN (The Things Network) server.

Luckily TTN runs a simple MQTT broker, so all we have to do it to edit the Telegraf configuration file to connect via MQTT to TTN.

Configuring Telegraf

First create a backup copy of the config file:

> mv /etc/telegraf/telegraf.conf /etc/telegraf/telegraf.conf_original

Then edit the config file:

> sudo nano /etc/telegraf/telegraf.conf

Telegraf config 1/3

[agent]

hostname = "localhost"

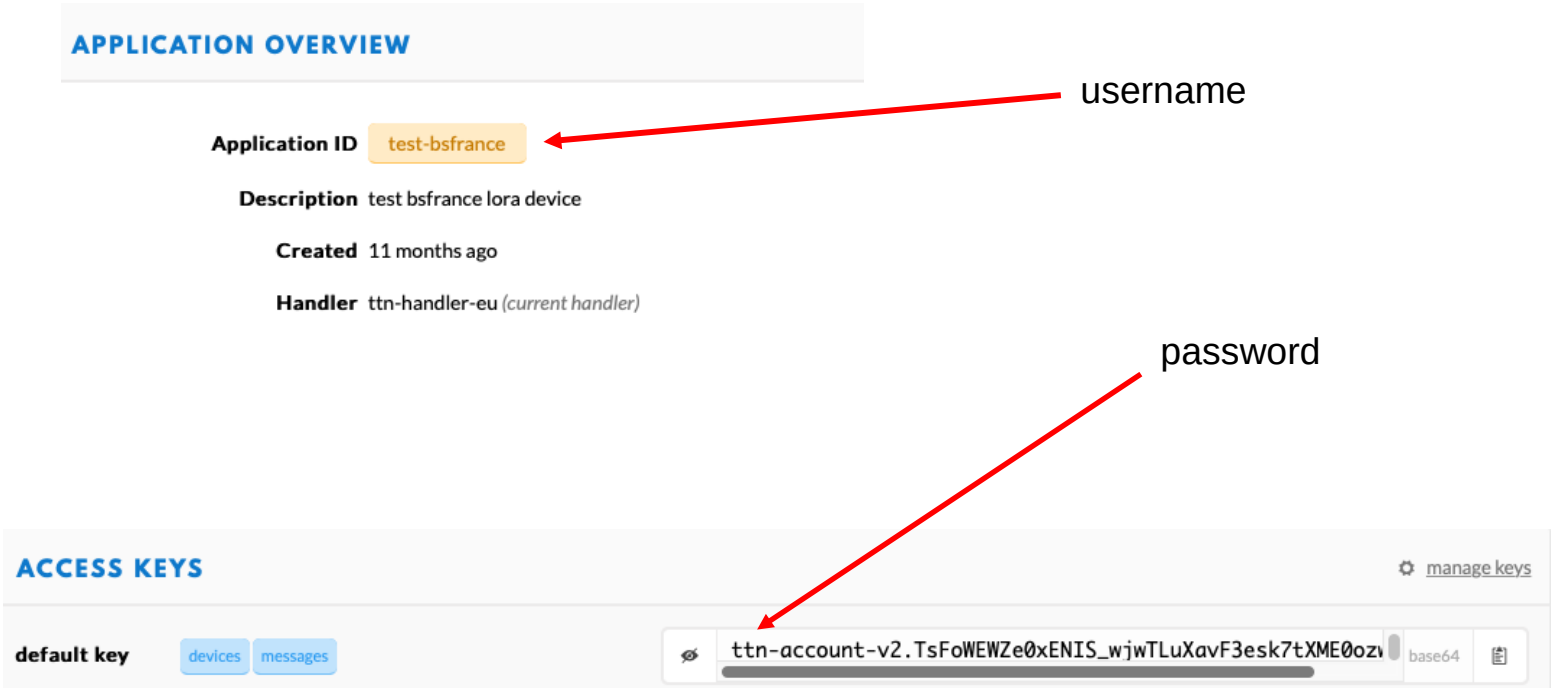
flush_interval = "15s"

interval = "15s"

Telegraf config 2/3

```
[[inputs.mqtt_consumer]]
servers = ["tcp://asia-se.thethings.network:1883"]
qos = 0
connection_timeout = "30s"
topics = ["+/devices/+/up"]
client_id = ""
username = "interlab-201909"
password = "ttn-account-
v2.ofsc_guNLbwyyfC7s6OPaCMQQIzwUXJXSA6CD-NgH4g"
data_format = "json"
```

Telegraf config 2/3



Telegraf config 3/3

[[outputs.influxdb]]

database = "database_name"

urls = ["http://localhost:8086"]

Restart Telegraf

Then we can restart telegraf and the metrics will begin to be collected and sent to InfluxDB.

```
pi@raspberrypi:~ $ service telegraf restart
```

Check database

We can now check if the data is sent from Telegraf to InfluxDB:

pi@raspberrypi:~ $ influx

Enter an InfluxQL query

> use database_name

Using database telegraf

> select * from "mqtt_consumer"

Database is populated!

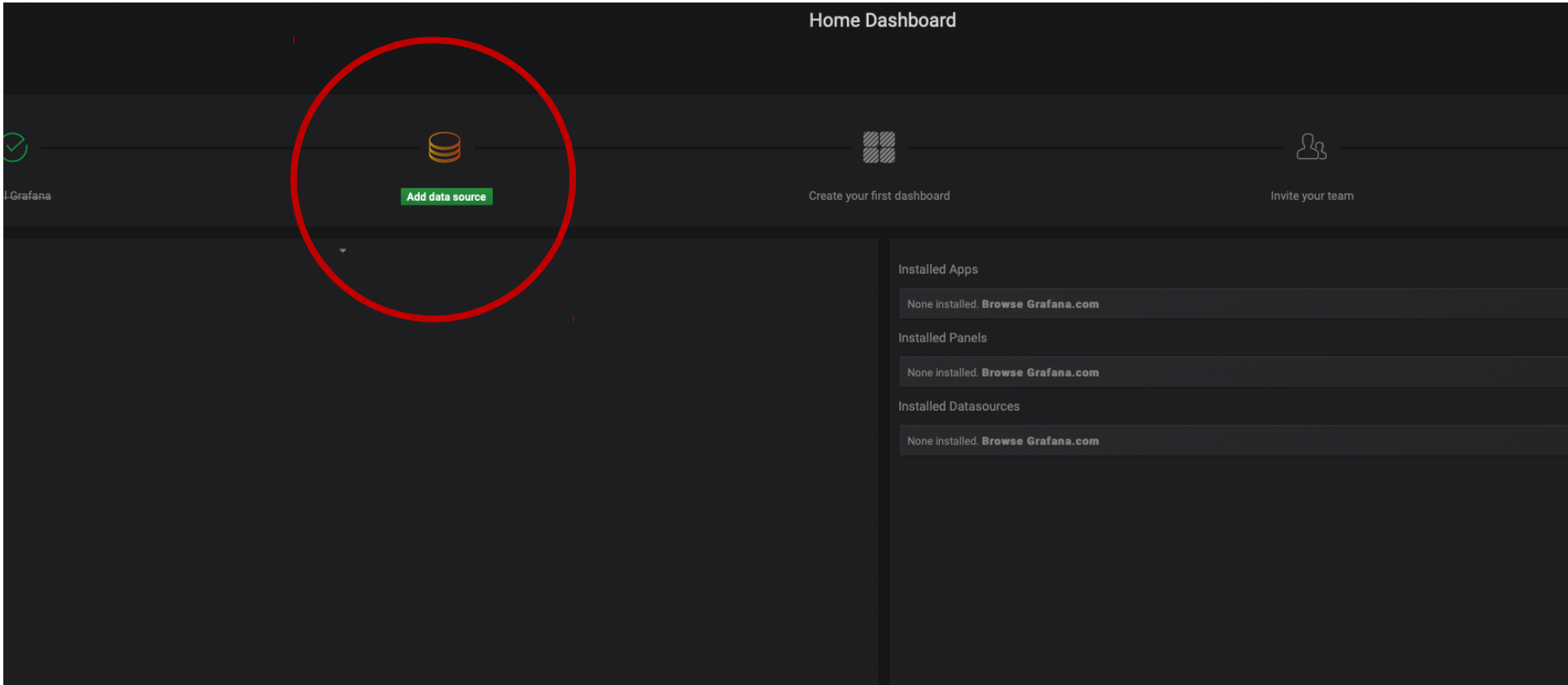
1557323990319369114 292 myserver 287744000 868.3 15
1 45.703526 13.72079 1
-112 -5.8 294082396 0
0 1008.1 23.6 45 0
2.92 7204 23.3 3.9 0
0 292 8459640 1
test-bsfrance/devices/bsfabp0001/up

1557324301943104151 293 myserver 287744000 868.5 15
2 45.703526 13.72079 1
-112 -6.2 605705244 0
0 1008.1 23.5 45 0
2.92 7204 23.3 3.9 0
0 293 8482785 1
test-bsfrance/devices/bsfabp0001/up

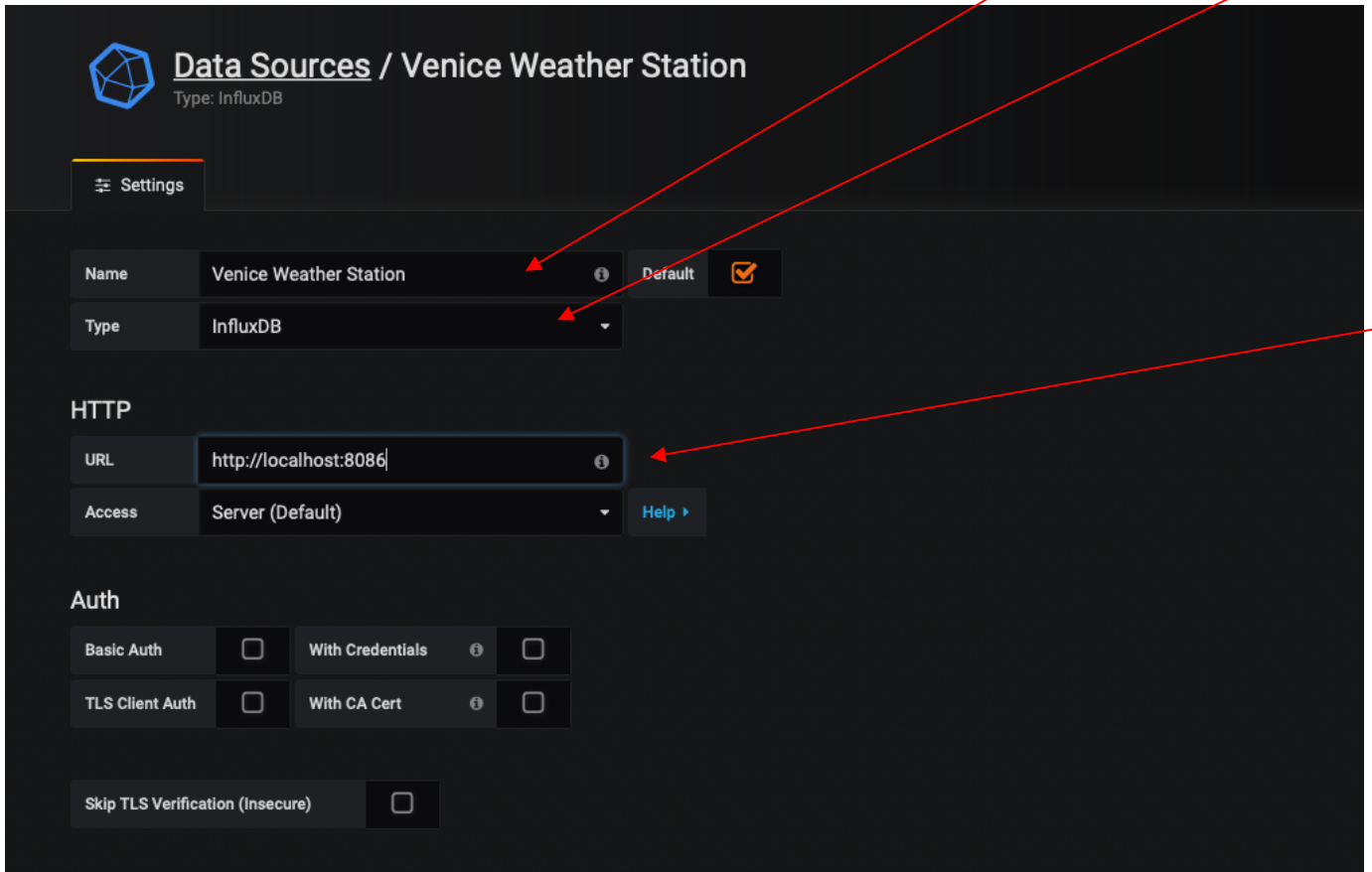
Log into Grafana

- Address: http://<YOUR_IP>:3000
- Username: admin
- Password: admin

Add data source



Add data source 1/2



Name

Type: InfluxDB

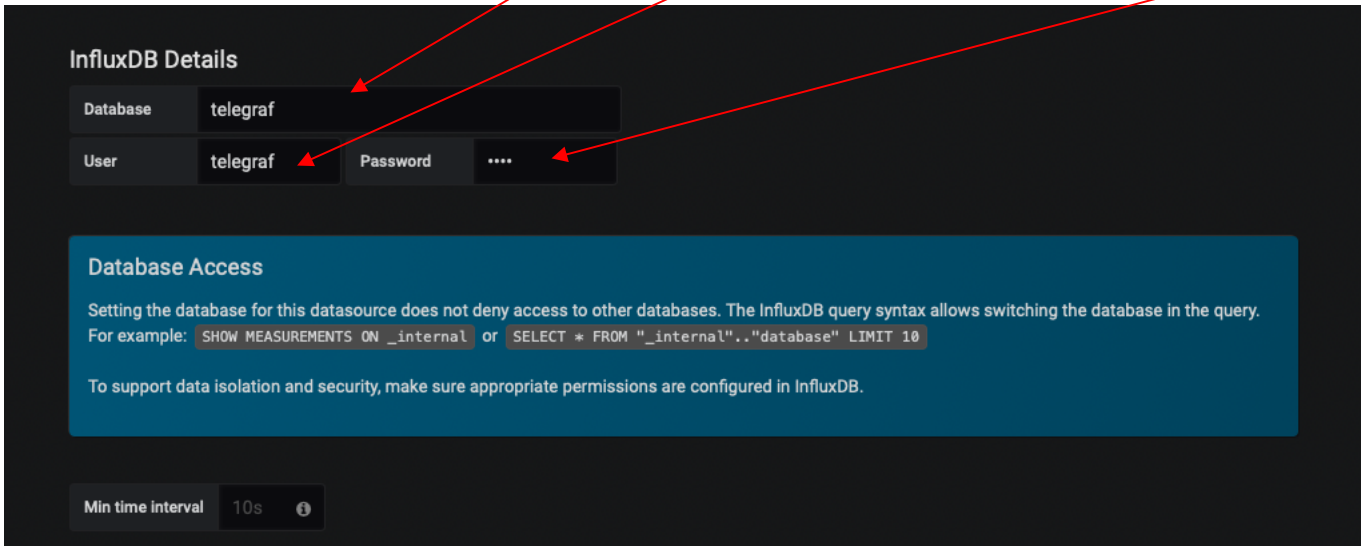
Address

Add data source 2/2

InfluxDB database name

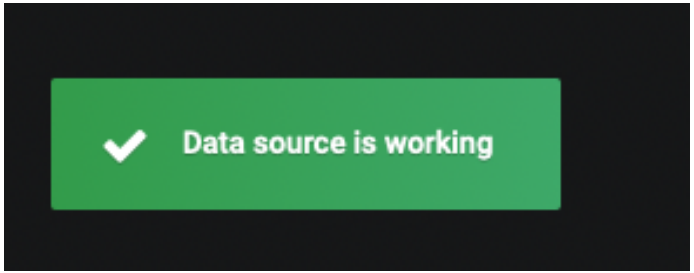
InfluxDB database username

InfluxDB database passwd



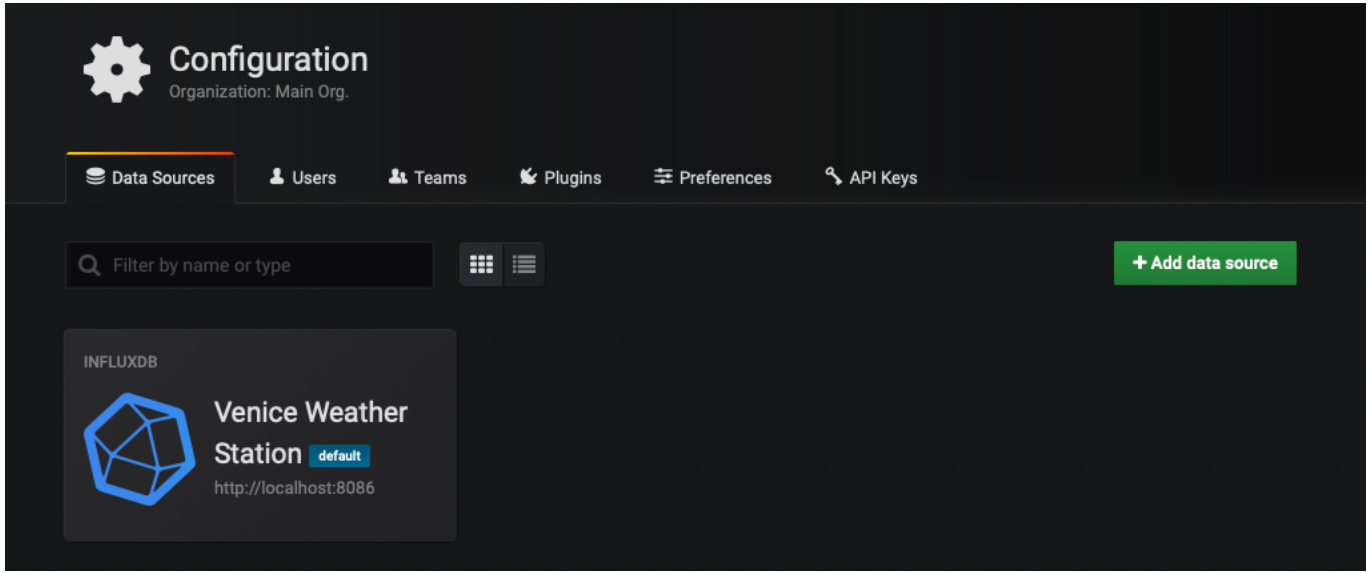
Add data source

If everything is fine you should see:

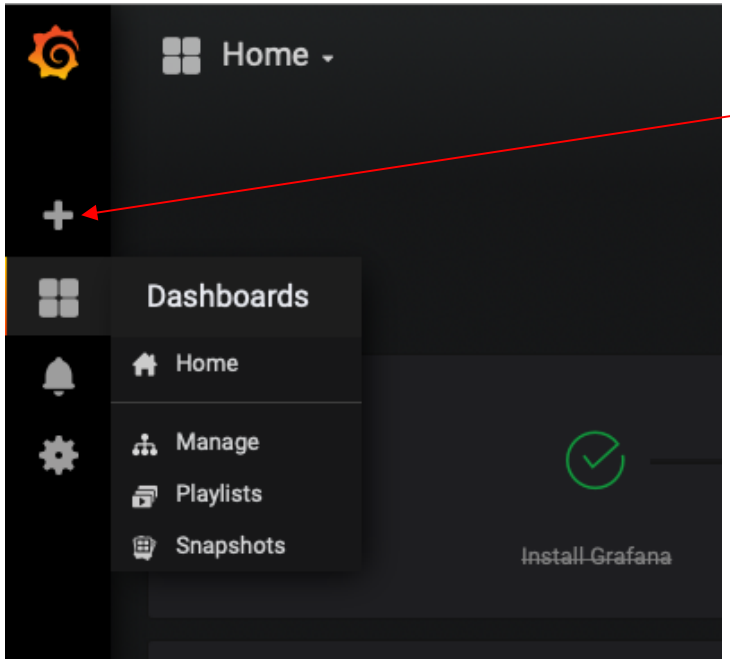


Add data source

If everything is fine you should see:

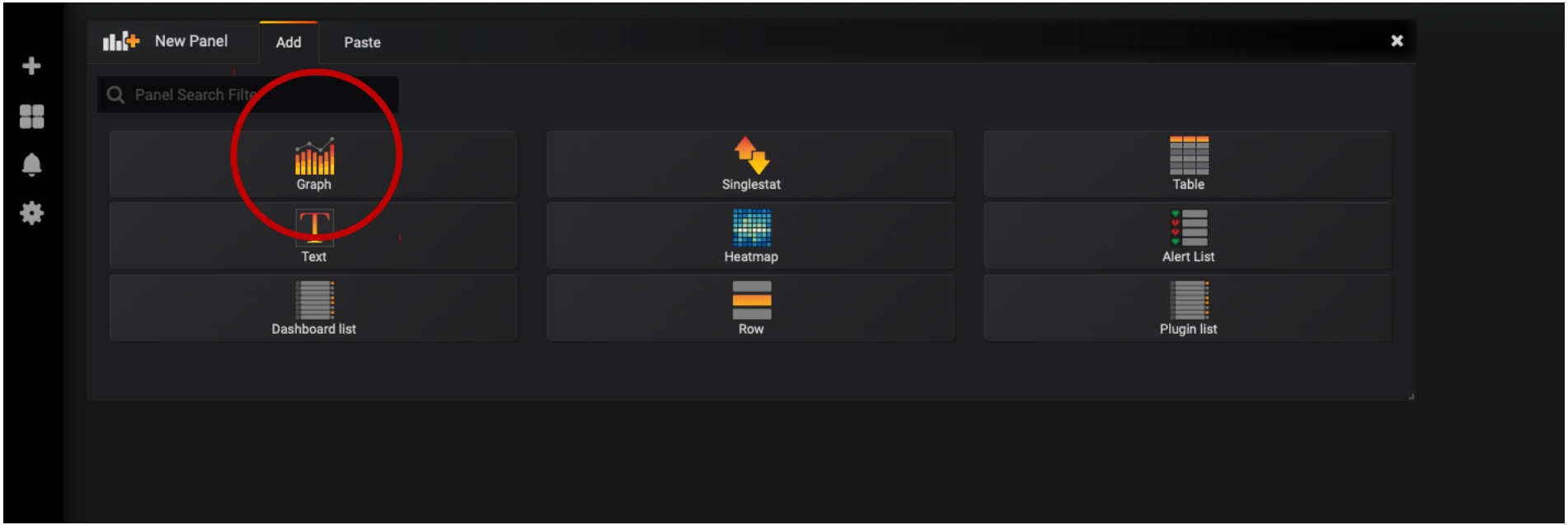


Add Dashboard



New

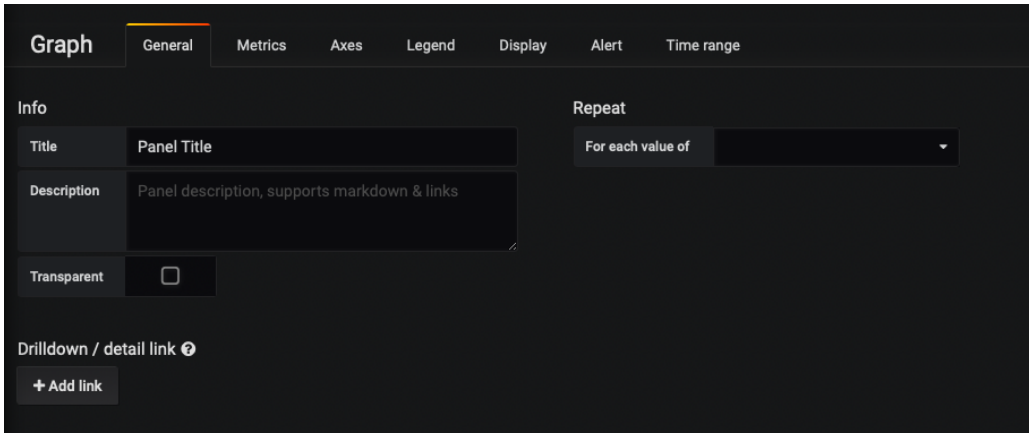
Add graph


New Panel
Add
Paste
Panel Search Filte
Graph
Singlestat
Table
Text
Heatmap
Alert List
Dashboard list
Row
Plugin list

Empty graph!

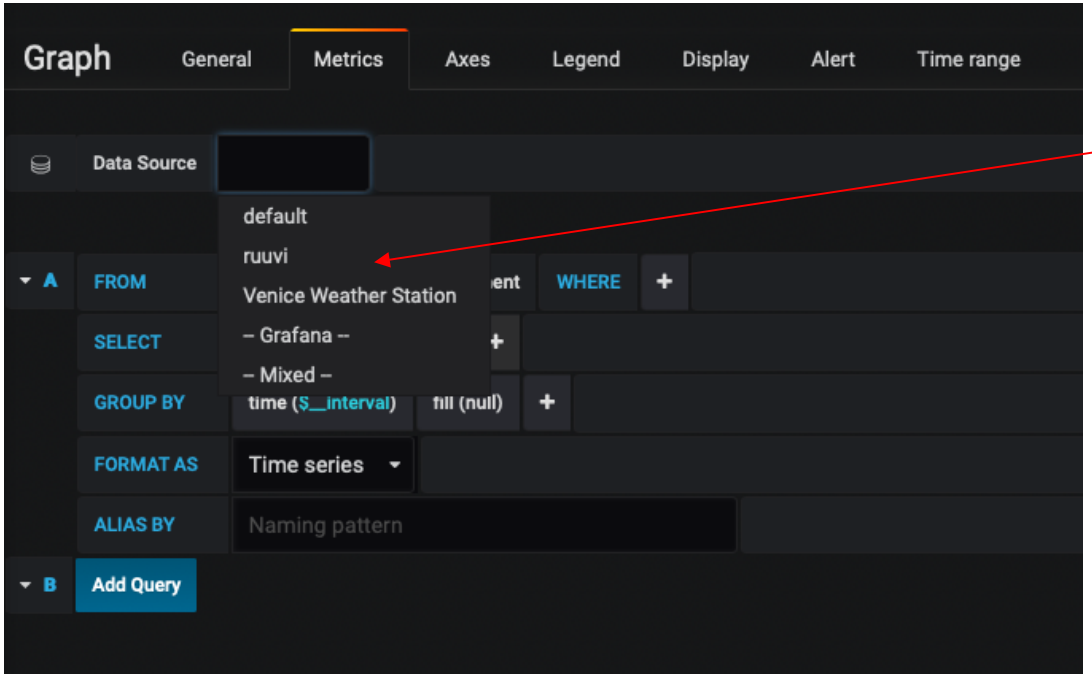
Select Edit

Add info to Graph: General



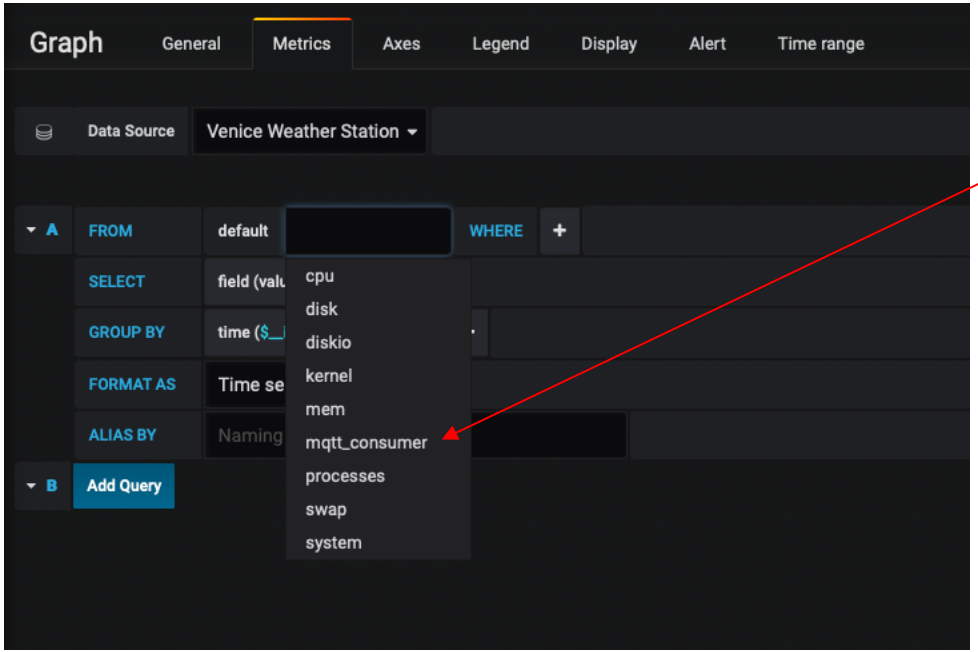
Add Title and Description

Add info to Graph: Metrics



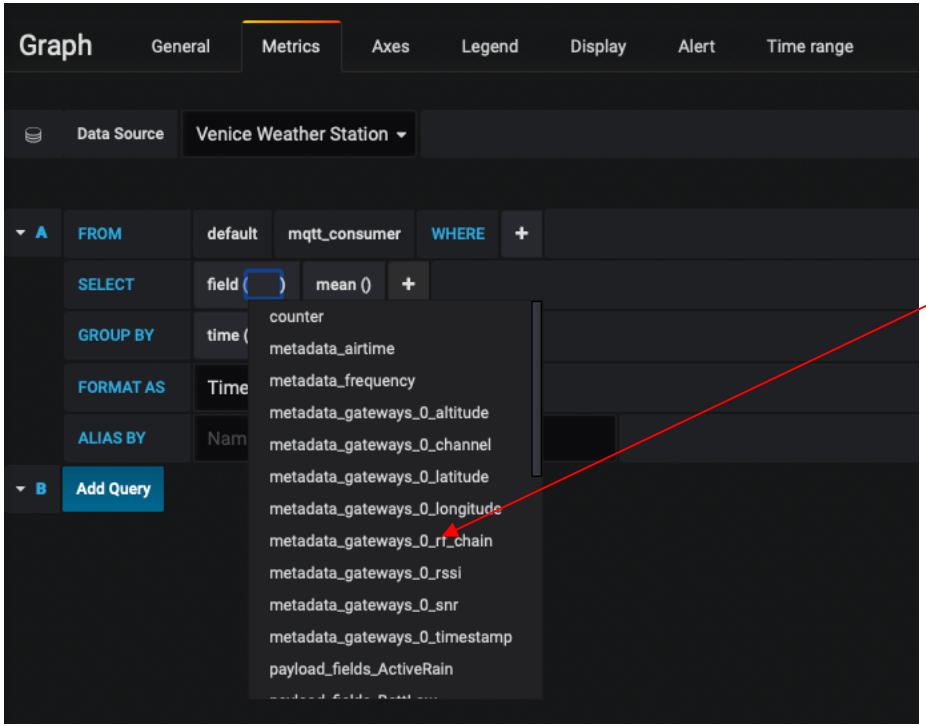
Your InfluxDB database name

Add info to Graph: Metrics



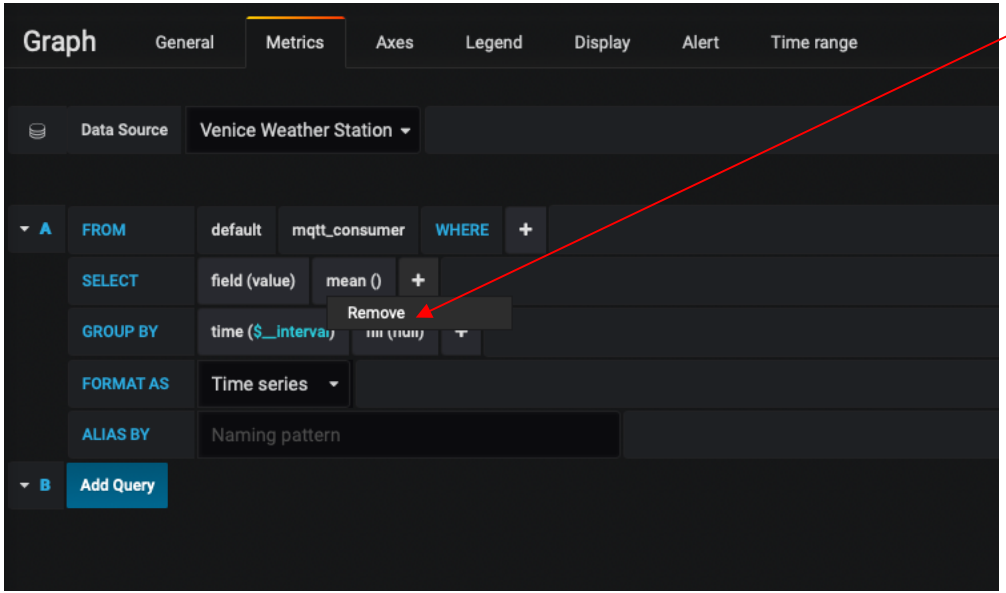
Select
mqtt_consumer

Add info to Graph: Metrics



Select
the variable
you want to
graph

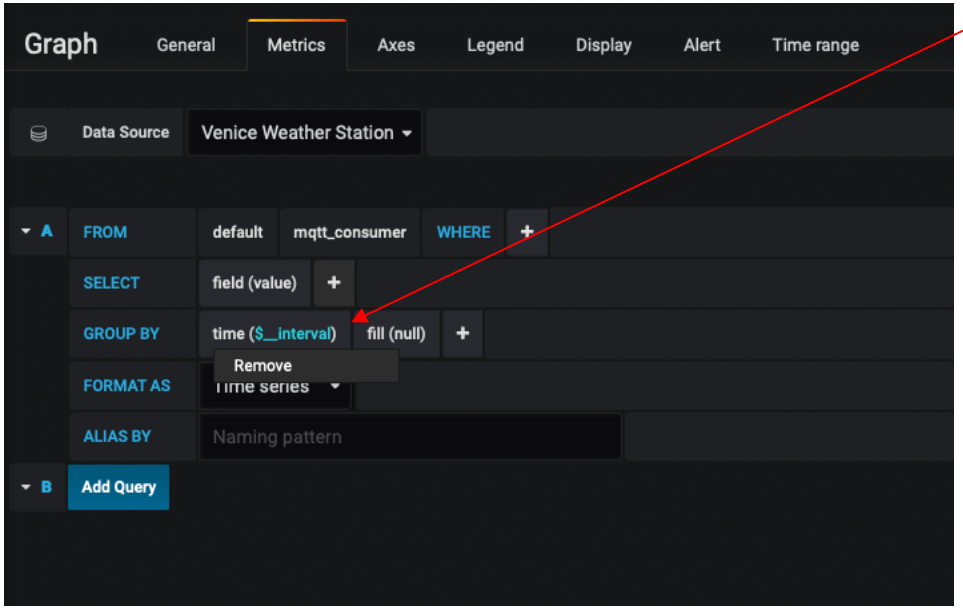
Add info to Graph: Metrics



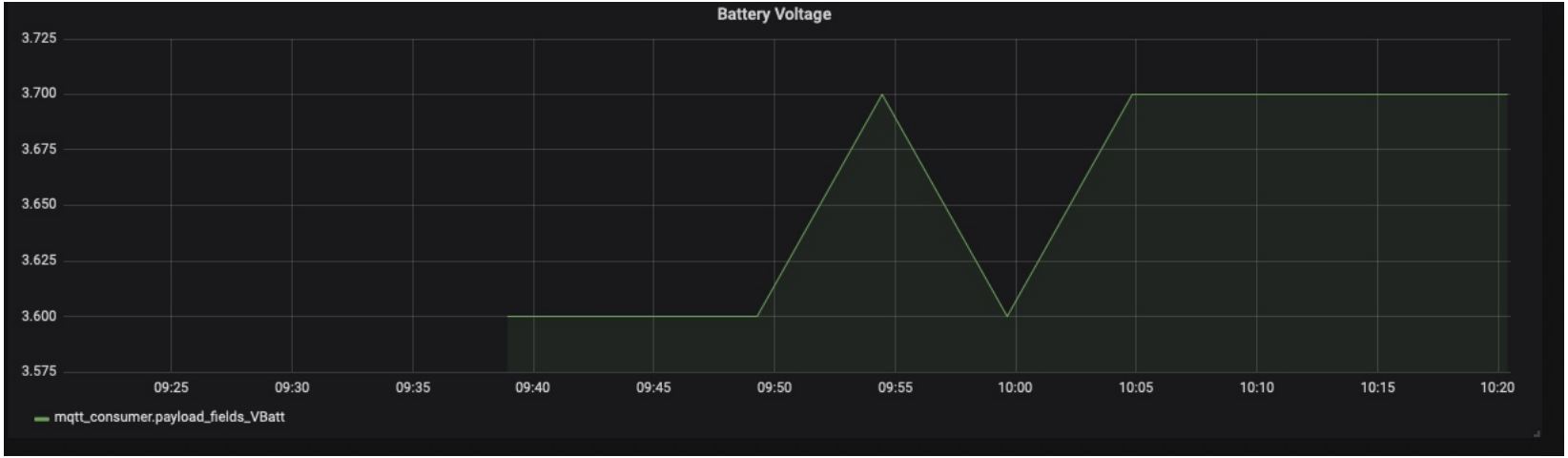
Remove mean()

Add info to Graph: Metrics

Remove
time($_interval)



Final result



Final result

- You can add as many variables as you want to the same Dashboard
- You can add users and different users can have access to different Dashboards
- You can export Dashboards

- Have fun exploring Grafana!

InfluxDB and Python

- You can interact with your Influx database using Python
- You need to install a library called *influxdb*
- Complete instructions are here: https://www.influxdata.com/blog/getting-started-python-influxdb/

InfluxDB and Python

Like many Python libraries, the easiest way to get up and running is to install the library using pip:

```
$ python3 -m pip install influxdb
```

Now let’s launch Python and import the library:

```
>>> from influxdb import InfluxDBClient
```

InfluxDB and Python

Next we create a new instance of the InfluxDBClient with information about the server that we want to access.

```
>>> client = InfluxDBClient(host='localhost', port=8086)
```

If Influx has username and password then:

```
>>> client = InfluxDBClient(host='mydomain.com', port=8086, username='myuser', password='mypass' ssl=True, verify_ssl=True)
```

InfluxDB and Python

Finally, we will list all databases and set the client to use a specific database:

```
>>> client.get_list_database()
```

```
>>> client.switch_database(‘database_name')
```

InfluxDB and Python

Let’s try to get some data from the database:

```
>>> client.query('SELECT * from "mqtt_consumer”’)
```

The query() function returns a ResultSet object, which contains all the data of the result along with some convenience methods. Our query is requesting all the measurements in our database.

InfluxDB and Python

You can use the get_points() method of the ResultSet to get the measurements from the request, filtering by tag or field:

```
>>> points=results.get_points()

>>> for item in points:

    print(item['time'])
```

InfluxDB and Python

You can get mean values, number of items, etc:

```
>>> client.query('select count(payload_fields_Rainfall) from mqtt_consumer’)

>>> client.query('select mean(payload_fields_Rainfall) from mqtt_consumer’)

client.query('select * from mqtt_consumer  WHERE time > now() - 7d')
```

Influx and Python: Exercises

1) Send some temperature and humidity data to InfluxDB via TTN. Save the data as csv (comma separated values) using Python and InfluxDB.

2) Produce a graph of the last 20 temperature measurements using Python and InfluxDB.

Summary

We learned how to install Telegraf, InfluxDB and Grafana.

We learned how to use Grafana to visualize data coming from an IoT network.

We learned how to interact with InfluxDB using Python.

Feedback?

Email mzennaro@ictp.it