GWDG
AG Computing
Marcus Merz

Tutorial 1 / March 1, 2023
HPC System Administration / WiSe 2022/23
75 Minutes Total

Learning Objectives

The learning objectives in the tutorial are:

- Installing a monitoring stack on the cloud server
- Understand how the components of the monitoring stack work together/are interlinked
- Being able to create panels with plots to display data collected by the monitoring stack

Tools

- InfluxDB, Telegraf
- Grafana
- Centos 8 server
- an editor (vim, nano,...)
- bash
- browser

Contents

Installing InfluxDB 1: Tutorial (20 min)

Goal of this task is to set up the InfluxDB v2x, which is the foundation for the rest of the tutorial. After this the InfluxDB should be running and the following information is available for the further tasks:

- $< influxip >$ - IP of the InfluxDB server: not the floating ip
- $< influxport >$ - Port of InfluxDB: the port the InfluxDB will listen to
- $< org >$ - Organization Name - the organization of the database within InfluxDB
- $< bucket >$ - Bucket Name - the name of the database within InfluxDB
- $< influxuser >$ - the main user for the Bucket
- $< influxpassword >$ Password - password of the user for the bucket
- $< token >$ Token - the access token to access the bucket
- $< grafanaadminpass >$ - the password for the admin access of the grafana server (application)
- $< frontendfloatingip >$ - the floating ip of the frontend server

The term bucket will be explained a little bit more later.
It should be ensured that the data listed above is stored somewhere - e.g. a pad or in a local editor - for easy access.

The $< frontendfloatingip >$ can be found in the cloud administration tool cloud.gwdg.de

1.1 Install the packages

The database for storing the metrics provided by the different agent on the systems to monitor has to be set up first. This will be done on the **frontend server**.

- Login to the server via ssh.
- `$ ifconfig`
- note down the *inet* ip adress of *eth0* as $< influxip >$ w/o the netmask: e.g if the *inet* is 10.254.1.9/24 $< influxip >$ would be 10.254.1.9
- note down "8086" as $< influxport >$ (will be changed later - just in case)

The standard Redhat/Centos package manager **yum** is used to install the InfluxDB. The repository is not yet in the repository list of **yum** and has to be added. Copy the following bash code block to the bash and execute it:

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdb.repo
[influxdb]
name = InfluxDB Repository { RHEL \$releasever
baseurl = https://repos.influxdata.com/rhel/\$releasever/\$basearch/stable
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdb.key
EOF
```

The InfluxDB repository should have been added. Now the InfluxDB and its according commandline tools can be installed:

- `$ sudo yum install influxdb2 influxdb2-cli --nogpgcheck`
- confirm the installation of the two packages

In order to set the InfluxDB2 up the according service has to be started:
`$ sudo systemctl start influxdb`

This gives no feedback. In order to check if the system is running, the status can be checked via
`$ sudo systemctl status influxdb`

The output should be similar to this and should not contain errors or the info that the service did not start:

```
influxdb.service - InfluxDB is an open-source, distributed, time series database
   Loaded: loaded (/usr/lib/systemd/system/influxdb.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2023-02-15 20:59:20 CET; 1h 16min ago
     Docs: https://docs.influxdata.com/influxdb/
 Main PID: 1777 (influxd)
    Tasks: 10 (limit: 11167)
   Memory: 22.0M
   CGroup: /system.slice/influxdb.service
           1777 /usr/bin/influxd -config /etc/influxdb/influxdb.conf

Feb 15 21:04:12 influx.novalocal influxd-systemd-start.sh[1777]: ts=2023-02-15T20:04:12.537837Z lvl=info
    ↪ msg="Executing query" log_id=0g1fPRtW000 service=query query="SHOW DATABASES"
Feb 15 21:04:12 influx.novalocal influxd-systemd-start.sh[1777]: [httpd] ::1 - - [15/Feb/2023:21:04:12
    ↪ +0100] "GET /query?q=SHOW+DATABASES HTTP/1.1" 200 109 "-" "curl/7.61.1"
    ↪ ea7acf28-ad6b-11ed-8003-fa163e2e1076 332
Feb 15 21:18:13 influx.novalocal influxd-systemd-start.sh[1777]: [httpd] ::1 - - [15/Feb/2023:21:18:13
    ↪ +0100] "POST /api/v2/query?org=hpcsa HTTP/1.1 " 403 100 "-" "influx/2.6.1 (linux) Sha/61c5b4d
    ↪ Date/2022-12-29T15:41:09Z" dfa73235-ad6d-11ed-8004-fa163e2e1076 107
Feb 15 21:18:24 influx.novalocal influxd-systemd-start.sh[1777]: [httpd] ::1 - - [15/Feb/2023:21:18:24
    ↪ +0100] "POST /api/v2/query?org=hpcsa HTTP/1.1 " 403 100 "-" "influx/2.6.1 (linux) Sha/61c5b4d
    ↪ Date/2022-12-29T15:41:09Z" e6389dcb-ad6d-11ed-8005-fa163e2e1076 89
Feb 15 21:29:19 influx.novalocal influxd-systemd-start.sh[1777]: ts=2023-02-15T20:29:19.633746Z lvl=info
    ↪ msg="Retention policy deletion check (start)" log_id=0g1fPRtW000 service=retention
    ↪ trace_id=0g1h7J_G000 op_name=retention_delete_check op_event=start
Feb 15 21:29:19 influx.novalocal influxd-systemd-start.sh[1777]: ts=2023-02-15T20:29:19.633798Z lvl=info
    ↪ msg="Retention policy deletion check (end)" log_id=0g1fPRtW000 service=retention
    ↪ trace_id=0g1h7J_G000 op_name=retention_delete_check op_event=end op_elapsed=0.065ms
Feb 15 21:43:24 influx.novalocal influxd-systemd-start.sh[1777]: [httpd] ::1 - - [15/Feb/2023:21:43:24
    ↪ +0100] "GET /health HTTP/1.1" 200 107 "-" "influx/2.6.1 (linux) Sha/61c5b4d
    ↪ Date/2022-12-29T15:41:09Z" 647410e2-ad71-11ed-8006-fa163e2e1076 110
Feb 15 21:59:19 influx.novalocal influxd-systemd-start.sh[1777]: ts=2023-02-15T20:59:19.633744Z lvl=info
    ↪ msg="Retention policy deletion check (start)" log_id=0g1fPRtW000 service=retention
    ↪ trace_id=0g1iqApG000 op_name=retention_delete_check op_event=start
Feb 15 21:59:19 influx.novalocal influxd-systemd-start.sh[1777]: ts=2023-02-15T20:59:19.633849Z lvl=info
    ↪ msg="Retention policy deletion check (end)" log_id=0g1fPRtW000 service=retention
    ↪ trace_id=0g1iqApG000 op_name=retention_delete_check op_event=end op_elapsed=0.116ms
Feb 15 22:13:52 influx.novalocal influxd-systemd-start.sh[1777]: [httpd] ::1 - - [15/Feb/2023:22:13:52
    ↪ +0100] "GET /health HTTP/1.1" 200 107 "-" "influx/2.6.1 (linux) Sha/61c5b4d
    ↪ Date/2022-12-29T15:41:09Z" a61e6767-ad75-11ed-8007-fa163e2e1076 64
```

To ensure that this service will be started after a boot it needs to be enabled permanentely:
`$ sudo systemctl enable influxdb`

Hints

- The reason to use the option *–nogpgcheck* is an issue with the gpg-keys that confirm the identity of the binary packages that are installed via yum. Usually these *fingerprints* should be adapted by the package provider when building new versions, but this seems to be not the case. It is usually not recommended to install packages that fail to install because of gpg-fingerprints issue are just installed with this option. The repositories and the situation should be checked before doing so.

- If a service does not start and the status from systemctl is not helpfull one should check the log files: e.g. `$ journalctl -eu influxdb`

1.2 Configuring InfluxDB

When the InfluxDB service is installed and started it can be configured via the according commandline tools. To set up an initial database a username, the password for this user and a bucket name - the database name - has to be defined. Then the following command can be executed to set up InfluxDB (the values should be replaced by the chosen ones). The general form of the setup command is:

```
$ influx setup --username <influx user> --password <influxpassword> --bucket <bucket>
```

It has to be ensured that the values are noted down and are accessible for later user.

The setup process will ask for $< org >$ - enter it and write it down. Select “0” for the retention time. An example setup command execution:

```
influx setup --username hpcuser --password hpcsa_user --bucket hpcsa
? Please type your primary organization name gwdg
? Please type your retention period in hours, or 0 for infinite 0 0
? Setup with these parameters? --- confirm with 'y'
```

The output should be similar to this:

```
> Welcome to InfluxDB 2.0!
? Please type your primary organization name gwdg
? Please type your retention period in hours, or 0 for infinite 0
? Setup with these parameters?
  Username: hpcuser
  Organization: gwdg
  Bucket: hpcsa
  Retention Period: infinite
 Yes
User Organization Bucket
hpcuser gwdg  hpcsa
```

Two tasks are performed by this command

- the initial database for the organization with the user to access it is created. The database is called bucket in InfluxDB v2x as it is a combination of the database and the retention policy.
- a profile for accessing the freshly created database is created for the user. This allows easy access w/o the need to provide the access token to the DB manually every time the user wants to utilize the *influx* command to access a database. The file created is *.influxdbv2/configs* in the users home.

Now it is possible to check the authentication information stored in the DB:

```
$ influx auth list
```

```
ID   Description Token            User Name User ID   Permissions
0ac1bf2d2a40b000 hpcuser's Token slgaGixZXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXw== hpcuser
    ↪ 0ac1bf2d0580b000 [read:/authorizations write:/authorizations read:/buckets write:/buckets
    ↪ read:/dashboards write:/dashboards read:/orgs write:/orgs read:/sources write:/sources read:/tasks
    ↪ write:/tasks read:/telegrafs write:/telegrafs read:/users write:/users read:/variables
    ↪ write:/variables read:/scrapers write:/scrapers read:/secrets write:/secrets read:/labels
    ↪ write:/labels read:/views write:/views read:/documents write:/documents read:/notificationRules
    ↪ write:/notificationRules read:/notificationEndpoints write:/notificationEndpoints read:/checks
    ↪ write:/checks read:/dbrp write:/dbrp read:/notebooks write:/notebooks read:/annotations
    ↪ write:/annotations read:/remotes write:/remotes read:/replications write:/replications]
```

The token will be important later and allows accessing the database via the port. Note the token down as $< token >$.

In this example output the token is
"slgaGixZXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXw=="
- as this is different for every single config and installation the example token shown here can not be used, so ensure that you use your token from your output.

1.3 Change port of InfluxDB

In case that another port has to be used by influx it can be changed by editing the config files. In this case that is required in order to allow later access to the influx web interface. This can be done by appending the according address to the InfluxDB server config and of course by adapting the config for the Influx CLI tool. The following has to be executed line by line.

```
echo http-bind-address = \":8009\" | sudo tee -a /etc/influxdb/config.toml
sudo systemctl restart influxdb
sed -i s/8086/8009/g ~/.influxdbv2/configs
```

Important!
The `$ tee` command with the *-a* option appends a line to an existing file. If the given command is executed multiple times, e.g. to configure another port for the *http-bind-address* the config file will end up with having multiple lines defining that option. When restarting the InfluxDB service will fail due to a malformed config while giving a non explainatory log output. In that case the wrong *http-bind-address* lines in the config file */etc/influxdb/config.toml* have to be removed.

To check if the InfluxDB has been restarted with the correct port:

```
$ sudo systemctl status influxdb | grep port
```

The output should show that InfluxDB uses port 8009 now:

```
Feb 16 21:58:42 worker.novalocal influxd-systemd-start.sh[11730]: ts=2023-02-16T20:58:42.601641Z lvl=info
    ↪ msg=Listening log\_id=0g30CKUG000 service=tcp-listener transport=http addr=:8009 port=8009
```

The $< influxport >$ value in the notes has to be modified to be "8009", too.

Installing Telegraf 2: Tutorial (20 min)

In order to fill the database at least one node agent has to run in order provide metrics to the InfluxDB. This will be done in this tutorial.

2.1 Install package

The initial Telegraf agent will be installed on the frontend as this is the only machine with access to the internet which is required for an easy download and install of the Telegraf package.

Telegraf is part of the InfluxDB repository and needs the same repository setup as InfluxDB. Therefore, Telegraf can just be installed as InfluxDB repos has already been added to the repository list of the frontend server.
`$ sudo yum install telegraf --nogpgcheck`

The service is now installed but not started yet. Telegraf will not run with the default configuration - it has to be modified first.

2.2 Configure telegraf

In order to run Telegraf it has to be configured properly. At least one input and one output plugin has to be configured. The standard input plugin is configured, but the output is not yet done.

The information to be provided in the config in the following steps are the $< influxip >$, $< influxport >$, $< bucket >$, $< org >$ and $< token >$ from the InfluxDB setup. This information is used by the Telegraf agent to contact InfluxDB and access the database (bucket) that has been defined.

Steps

1. open the file */etc/telegraf/telegraf.conf* with an editor using sudo (e.g sudo nano)
   - if nano is not installed it must be installed via `$ sudo yum install nano`
2. search for the section [[outputs.influxdb_v2]]
3. uncomment (remove the leading #) and modify following entries according to the data collected during the influxdb config:
   - # [[outputs.influxdb_v2]] $\longrightarrow$ just uncomment
   - # urls = ["http://127.0.0.1:8086"] $\longrightarrow$ urls = ["http://$< influxip >$:$< influxport >$"]
   - # token = "" $\longrightarrow$ token = "$< token >$"
   - # organization = "" $\longrightarrow$ organization = "$< org >$"
   - # bucket = "" $\longrightarrow$ bucket = "$< bucket >$"
4. save the file and leave the editor (for nano CTRL-o, Return, CTRL-X)

To check if the configuration is correct:
`$ telegraf`

If there are no errors in the Telegraf output the config is ok:

```
023-02-16T06:07:26Z I! Using config file: /etc/telegraf/telegraf.conf
2023-02-16T06:07:26Z I! Starting Telegraf 1.25.2
2023-02-16T06:07:26Z I! Available plugins: 228 inputs, 9 aggregators, 26 processors, 21 parsers, 57
    ↪ outputs, 2 secret-stores
2023-02-16T06:07:26Z I! Loaded inputs: cpu disk diskio kernel mem processes swap system
2023-02-16T06:07:26Z I! Loaded aggregators:
2023-02-16T06:07:26Z I! Loaded processors:
2023-02-16T06:07:26Z I! Loaded secretstores:
2023-02-16T06:07:26Z I! Loaded outputs: influxdb_v2
2023-02-16T06:07:26Z I! Tags enabled: host=worker2.novalocal
2023-02-16T06:07:26Z I! [agent] Config: Interval:10s, Quiet:false, Hostname:"worker2.novalocal", Flush
    ↪ Interval:10s
^C2023-02-16T06:07:43Z I! [agent] Hang on, flushing any cached metrics before shutdown
```

This output shows which plugins are loaded for input and output.

To stop telegraf:
`$ CTRL-C`

Now the telegraf service can be started and enabled:
```
$ sudo systemctl start telegraf
$ sudo systemctl enable telegraf
```

Check if telegraf has been started:
`$ sudo systemctl status telegraf`

Hints

- the pre-configured standard plugin configuration can be found in the section *[[inputs.cpu]]* of the */etc/telegraf/telegraf.conf* file. More details to the settings/metrics for this plugin can be found on the according ≪CPU Input Plugin≫ webpage.

2.3 Smoke test for Telegraf-InfluxDB connection

The influx commandline tools can be used to check if data is arriving in the influxdb. On the *frontend* do:

```
$ influx query
```

This opens a query pipeline. The influx tool now waits for a query. Copy this query to the shell with the query pipeline

```
from(bucket: "<bucket>") |> range(start: -10m)
```

Press Ctrl-d, Ctrl-d to execute the query

This should give some output with table info, system info and data. Depending of the amount of data it will look like:

```
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                 _start:time _stop:time _field:string _measurement:string device:string fstype:string
                    ↪ host:string mode:string path:string _time:time _value:int
------------------------------ ------------------------------ ----------------------
       ↪ ---------------------- ---------------------- ---------------------- ----------------------
       ↪ ---------------------- ---------------------- ------------------------------
       ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z free disk vda1 ext4 worker2.novalocal rw /
       ↪ 2023-02-16T06:07:30.000000000Z 18264272896
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                 _start:time _stop:time _field:string _measurement:string device:string fstype:string
                    ↪ host:string mode:string path:string _time:time _value:int
------------------------------ ------------------------------ ----------------------
       ↪ ---------------------- ---------------------- ---------------------- ----------------------
       ↪ ---------------------- ---------------------- ------------------------------
       ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z inodes_free disk vda1 ext4 worker2.novalocal
       ↪ rw / 2023-02-16T06:07:30.000000000Z 1274475
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                 _start:time _stop:time _field:string _measurement:string device:string fstype:string
                    ↪ host:string mode:string path:string _time:time : _value:int
------------------------------ ------------------------------ ----------------------
       ↪ ---------------------- ---------------------- ---------------------- ----------------------
       ↪ ---------------------- ---------------------- ------------------------------
       ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z inodes_total disk vda1 ext4
       ↪ worker2.novalocal rw / 2023-02-16T06:07:30.000000000Z 1310720
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                 _start:time _stop:time _field:string _measurement:string device:string fstype:string
                    ↪ host:string mode:string path:string _time:time _value:int
------------------------------ ------------------------------ ----------------------
       ↪ ---------------------- ---------------------- ---------------------- ----------------------
       ↪ ---------------------- ---------------------- ------------------------------
       ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z inodes_used disk vda1 ext4 worker2.novalocal
       ↪ rw / 2023-02-16T06:07:30.000000000Z 36245
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                 _start:time _stop:time _field:string _measurement:string device:string fstype:string
                    ↪ host:string mode:string path:string _time:time _value:int
```

```
------------------------------ ------------------------------ ----------------------
    ↪ ---------------------- ---------------------- ---------------------- ----------------------
    ↪ ---------------------- ---------------------- ------------------------------
    ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z total disk vda1 ext4 worker2.novalocal rw /
    ↪ 2023-02-16T06:07:30.000000000Z 21046689792
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                 _start:time _stop:time _field:string _measurement:string device:string fstype:string
                     ↪ host:string mode:string path:string _time:time _value:int
```

Installing Grafana and setting up a simple dashboard 3: Tutorial (35 min)

Grafana will be used to display data collected by the Telegraf agent. In the following the Grafana service/server will be setup and configured. At the end a simple dashboard will be created.

3.1 Installing Grafana package

Grafana will be the outward facing user interface that will be used to display plots from time-series data. Therefor it needs to be installed on the *frontend*:

```
$ sudo yum install grafana
```

This will install two packages.

Before starting Grafana the port has to be adjusted to avoid conflicts:

- open /etc/grafana/grafana.ini in an editor with sudo right (e.g sudo nano - install it if not installed)
- search for the option *http_port*
- set the value to 8000
- save the file and exit

Now the Grafana server can be started:

```
$ sudo systemctl start grafana-server
```

The status should be checked as before with InfluxDB and Telegraf.

```
$ sudo systemctl status grafana-server
```

If the server is running it should be checked if the port is set correctly to “8000”:

```
$ sudo systemctl status grafana-server | grep address
```

The output should show the correct port to be used by grafana-server:

```
Feb 17 06:57:16 frontend.novalocal grafana-server[27906]: t=2023-02-17T06:57:16+0100 lvl=info msg="HTTP
    ↪ Server Listen" logger=http.server address=[::]:8000 protocol=http subUrl= socket=
```

If everything is ok the service can be enabled permanetly:

```
$ sudo systemctl enable grafana-server
```

3.2 First admin login

Setting up Grafana is done via the web-interface. Grafana provides its own web server. First the “admin” account has to be setup. This happens automatically when trying to login to Grafana for the first time.

Steps

1. open a browser
2. URL to use: $< frontendfloatingip >$:8000
3. enter "admin" as user name
4. enter "admin" as password
5. Grafana ask for a new password and the according confirmation - enter an arbitrary password
6. the password should be noted down as $< grafanaadminpass >$ - just in case

Hints

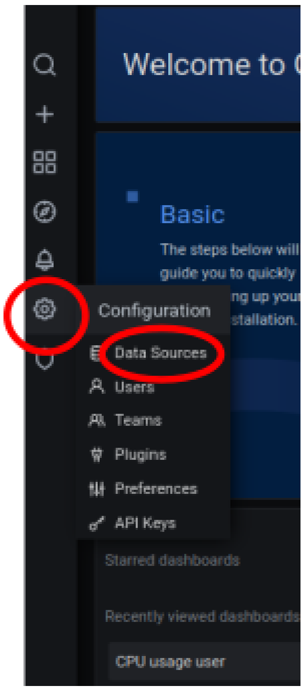
- If you loose the Grafana admin password it can be reset on the frontend:

```
$ sudo grafana-cli --homepath "/usr/share/grafana" admin reset-admin-password <new password>
```

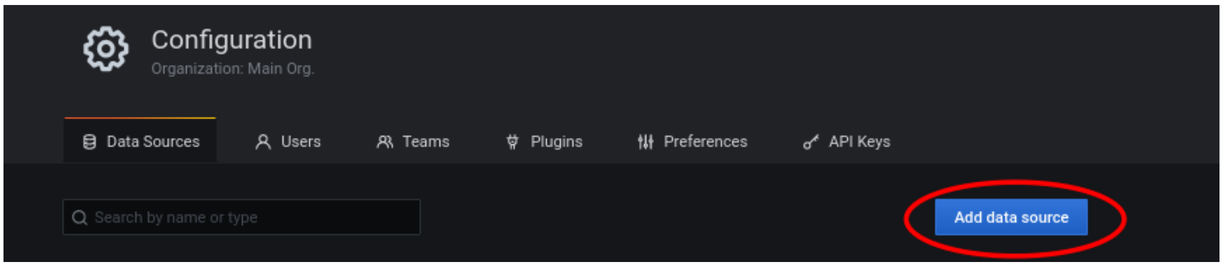
3.3 Create datasource

In order to display data in Grafana the application requires the information where and how to retrieve the data from. This is information a "Datasource". It is possible to define multiple datasources, but in this case only one is created.

As "admin" on the Grafana server select "Configuration – Data Sources" from the toolbar on the left:

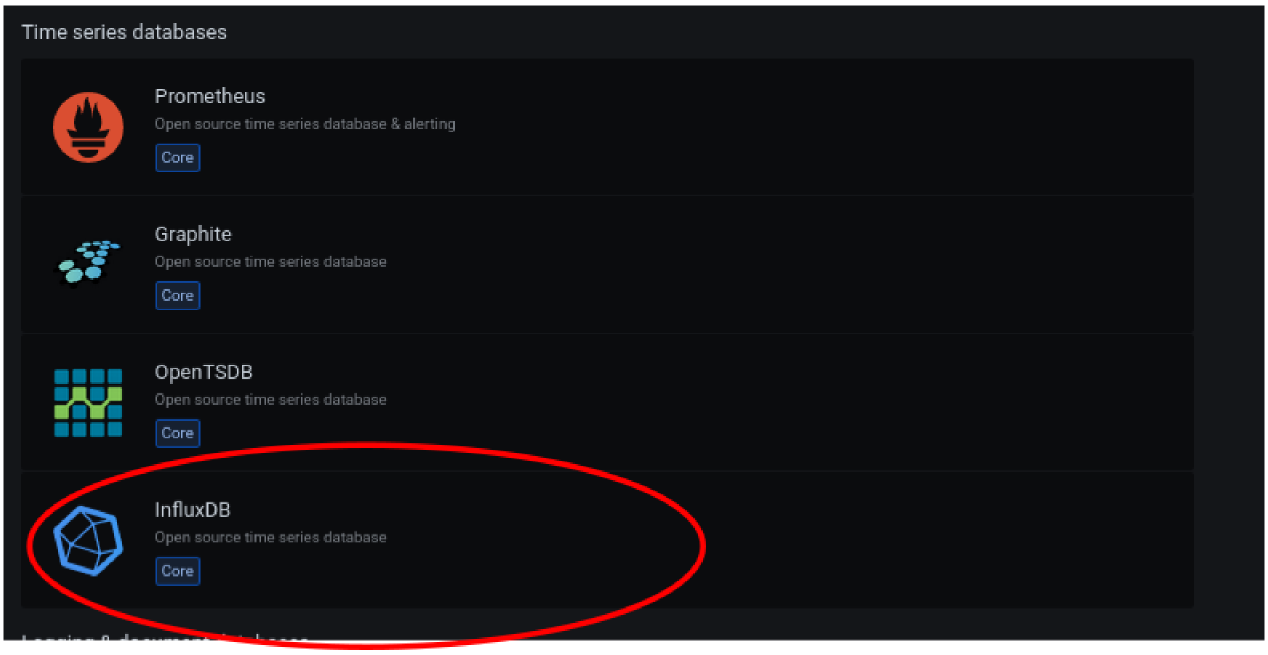


To add a datasource select the "Add data source" button:

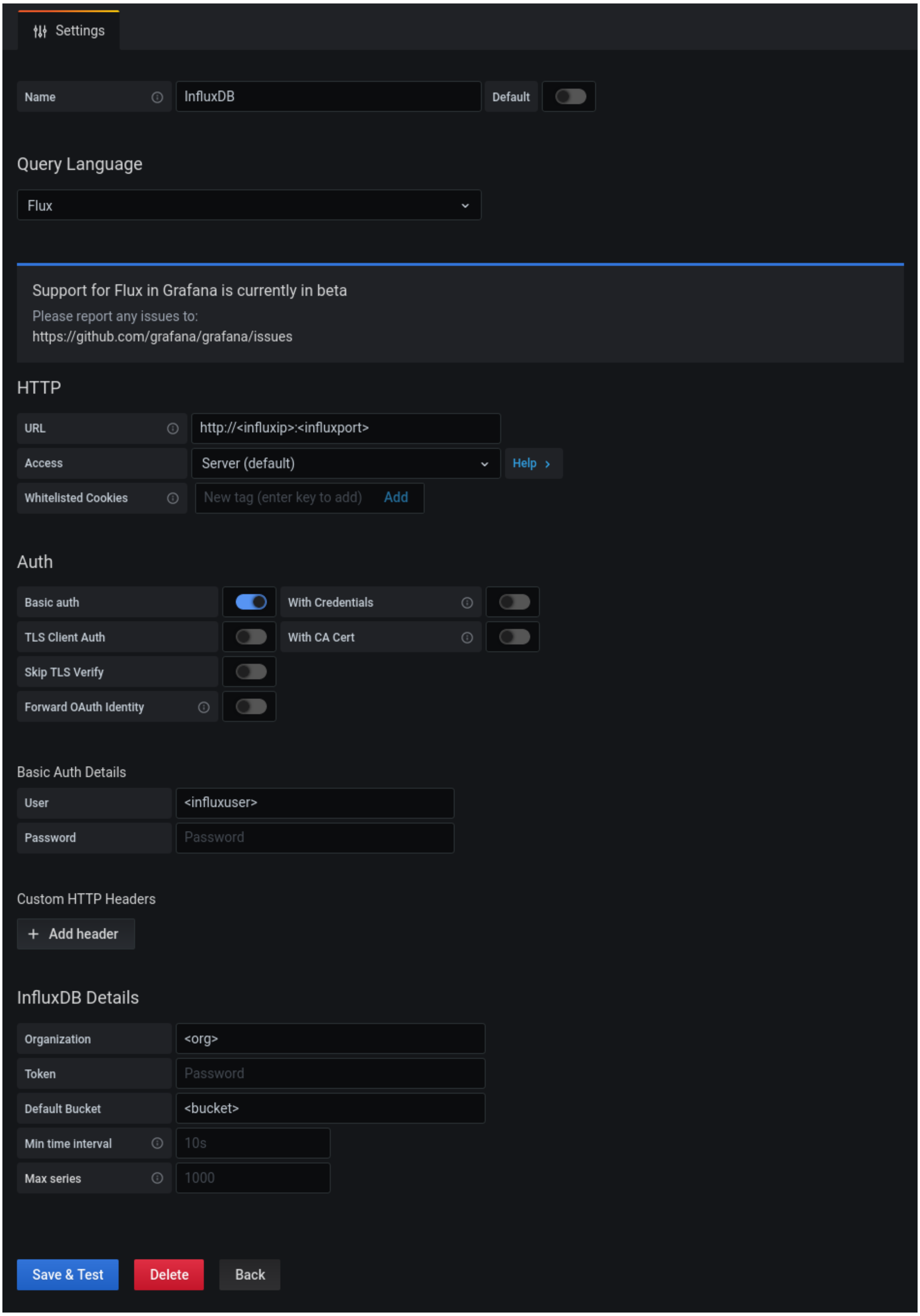


Now the data gathered before has to be provided in the form:

Select the option “InfluxDB” in the section time series db:


Time series databases
Prometheus
Open source time series database & alerting
Core
Graphite
Open source time series database
Core
OpenTSDB
Open source time series database
Core
InfluxDB
Open source time series database
Core

Fill out the given form with the collected information, check the image and the text below for hints:



Settings

Name	InfluxDB
Default	off

Query Language

Flux

Support for Flux in Grafana is currently in beta

Please report any issues to:
https://github.com/grafana/grafana/issues

HTTP

URL	http://<influxip>:<influxport>
Access	Server (default)
Whitelisted Cookies	New tag (enter key to add)

Auth

Basic auth	on	With Credentials	off
TLS Client Auth	off	With CA Cert	off
Skip TLS Verify	off		
Forward OAuth Identity	off		

Basic Auth Details

User	<influxuser>
Password	Password

Custom HTTP Headers

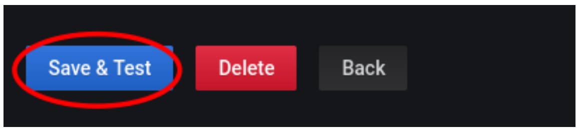
InfluxDB Details

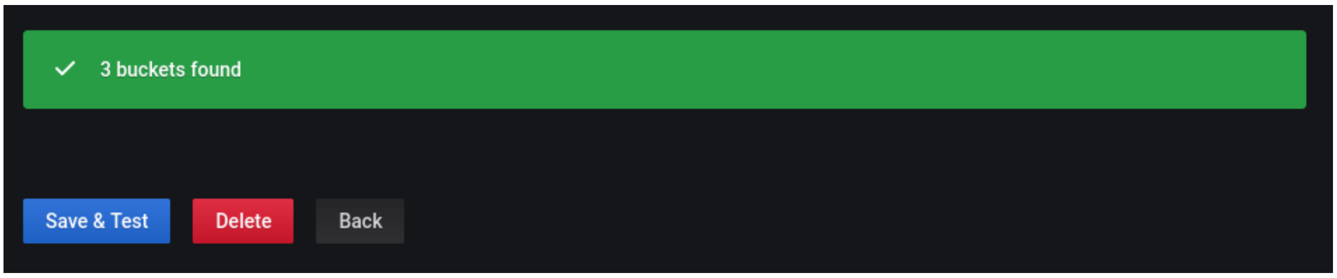
Organization	<org>
Token	Password
Default Bucket	<bucket>
Min time interval	10s
Max series	1000

- Name — select any name

- Query Language — select "Flux"
- Password for $< influxuser >$ is $< influxpassword >$
- Token — $< token >$

When done press the "Save and test" button at the bottom of the form. If everything is setup correclty there will be green feedback:




3 buckets found

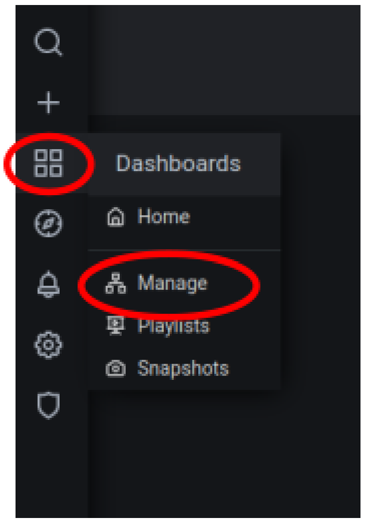
Grafana is now able to connect to the given database and retrieve metrics to display

3.4 Create simple dashboard

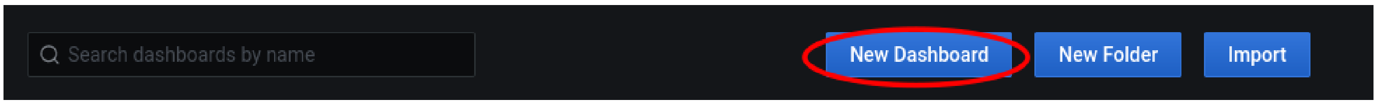
As the datasource is setup it is possible to create panels showing plots from metrics of this source. In the following a simple dashboard is created using the Flux query language. Flux is a topic on its own, so the according query will be provided.

This can be done as user "admin" in Grafana as no other user is created yet.

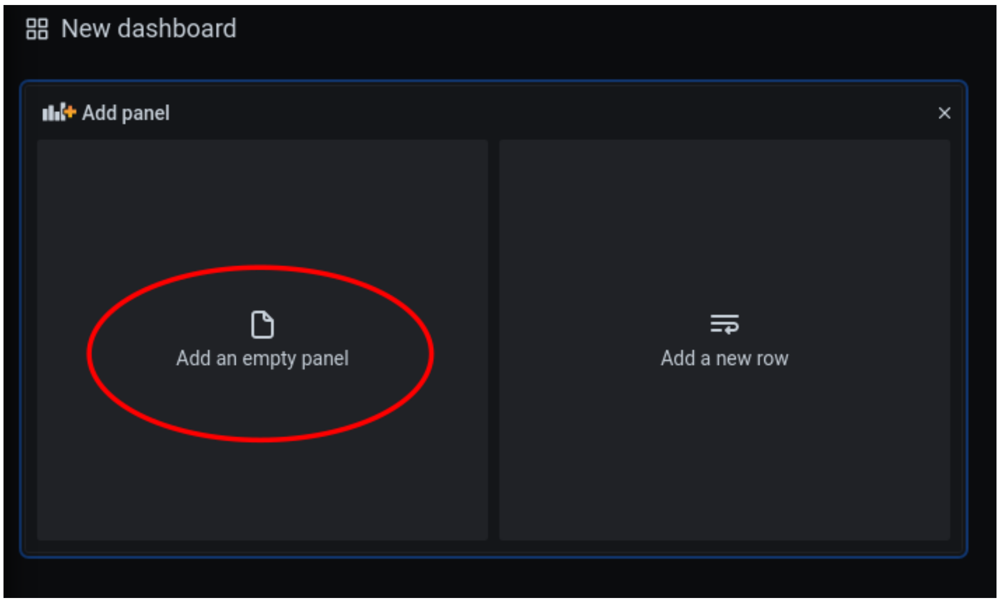
Select the menu "Dashboards — Manage" from the toolbar on the left:



Create a new dashboard by pressing "New dashboard":

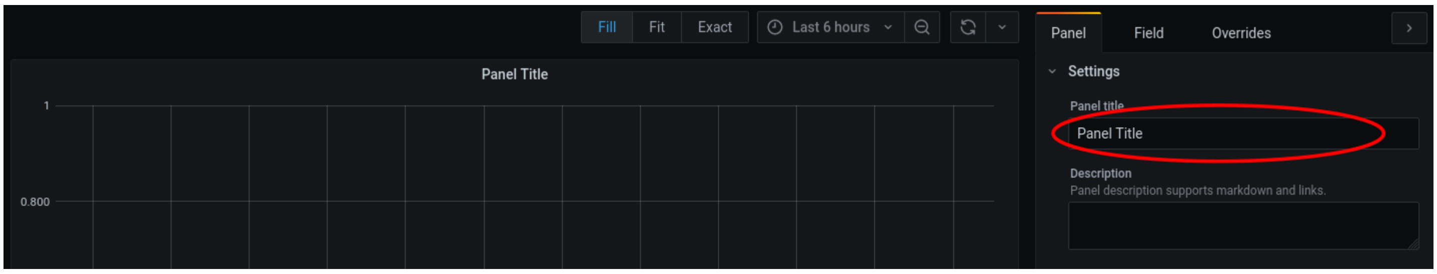


The dashboard creation interface shows up. Select the "Add an empty panel" option in the "Add panel" field:

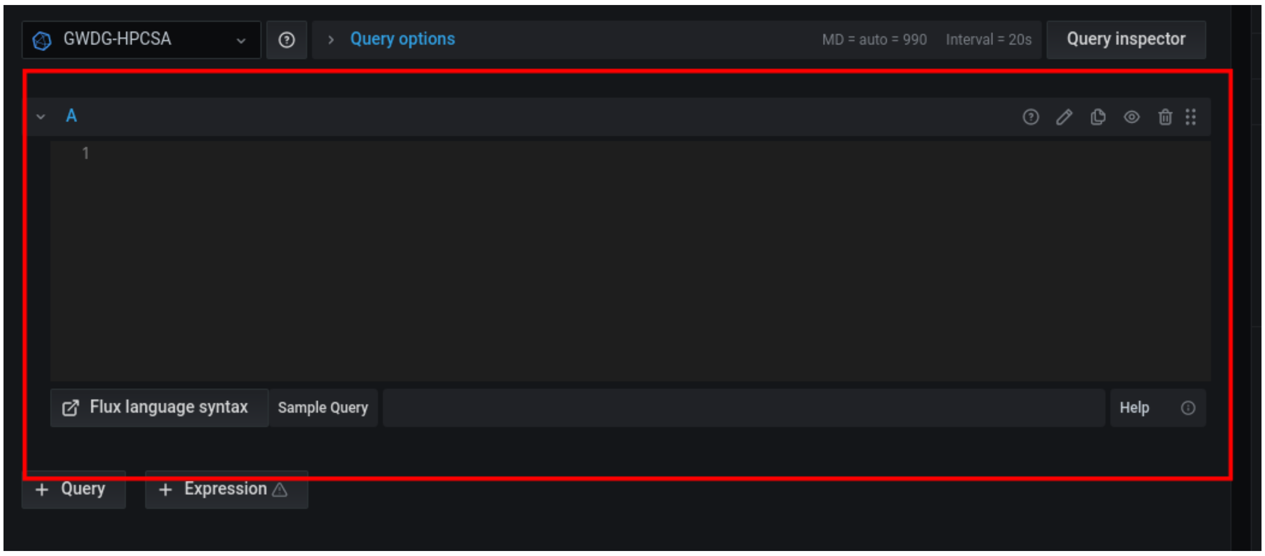


The panel editor is now presented. It is already setup to the correct datasource as it is the only one created yet.

Enter a panel title in the top right:



On the bottom is the query editor:

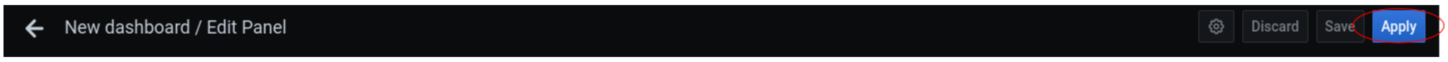


Copy and paste this Flux query into the editor and modify the bucket to match $< bucket >$ from the InfluxDB config:

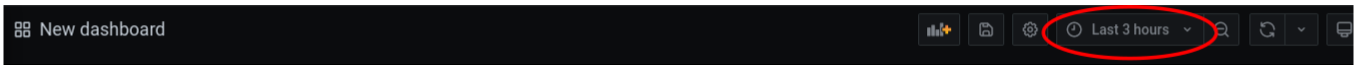
```
from(bucket: "<bucket>")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "cpu")
  |> filter(fn: (r) => r["_field"] == "usage_user")
  |> filter(fn: (r) => r["cpu"] == "cpu-total" or r["cpu"] == "cpu0" or r["cpu"] == "cpu1")
  |> yield(name: "mean")
```

This query request data from the given bucket with in the time-range given (this is a dynamic timerange). The InfluxDB will first select all data in the given timespan, and they apply the filters on it. In this case the query filters for the general CPU usage of the users on the systems which provide the according measurements.

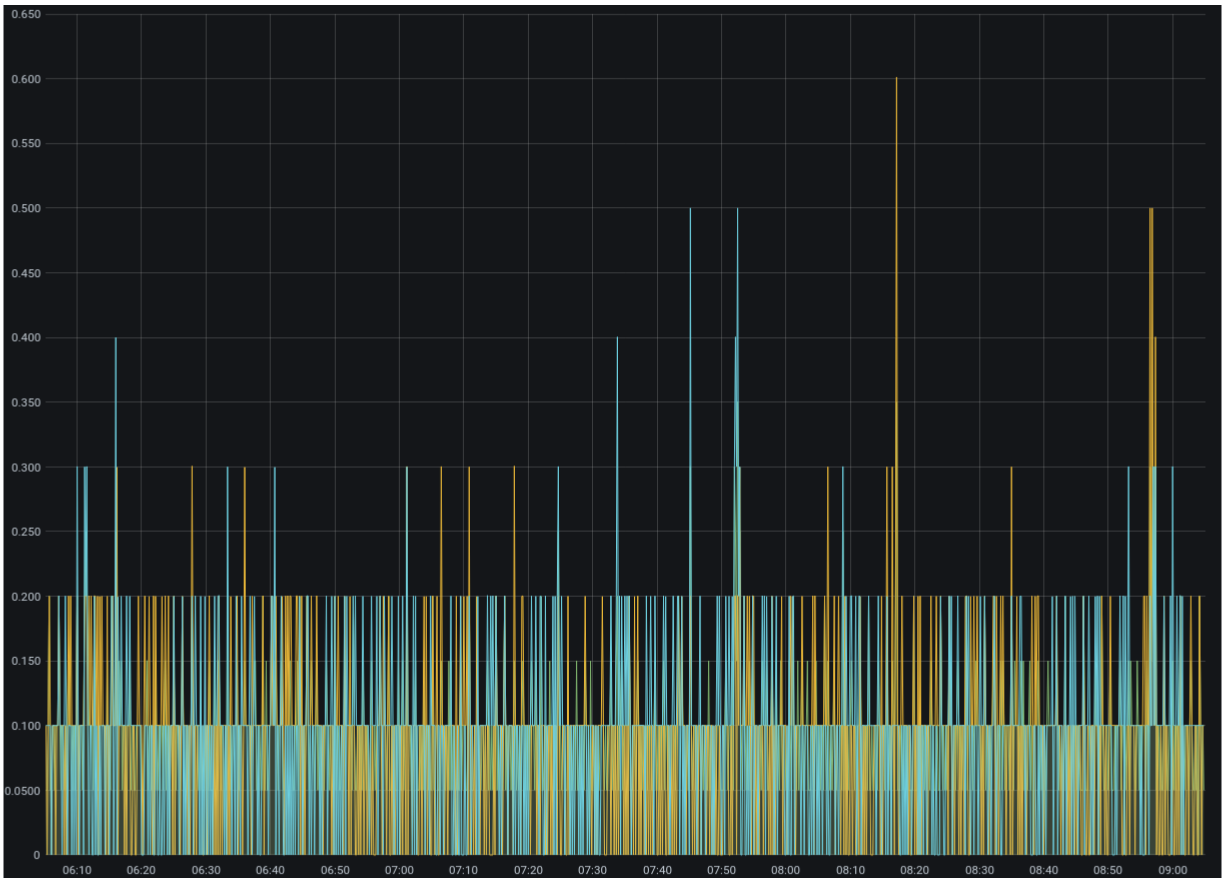
Press apply in the top right corner:



The panel is shown, but it is empty - the timeframe has to be selected in the dropdown menu first



Now there should be data displayed in the panel



To save this dashboard for future use press the name/title of the dashboard and select “edit”. The editor shows up again. Save it with the according button in the top right.

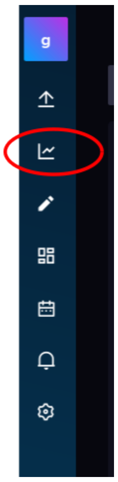
Optional: Explore data and create own queries 4: Tutorial (0 min)

The dashboard created is quite simple. The issue with creating more interesting panel is to know the available metrics and the syntax. InfluxDB provides a web interface for data exploration to make this easier for the users.

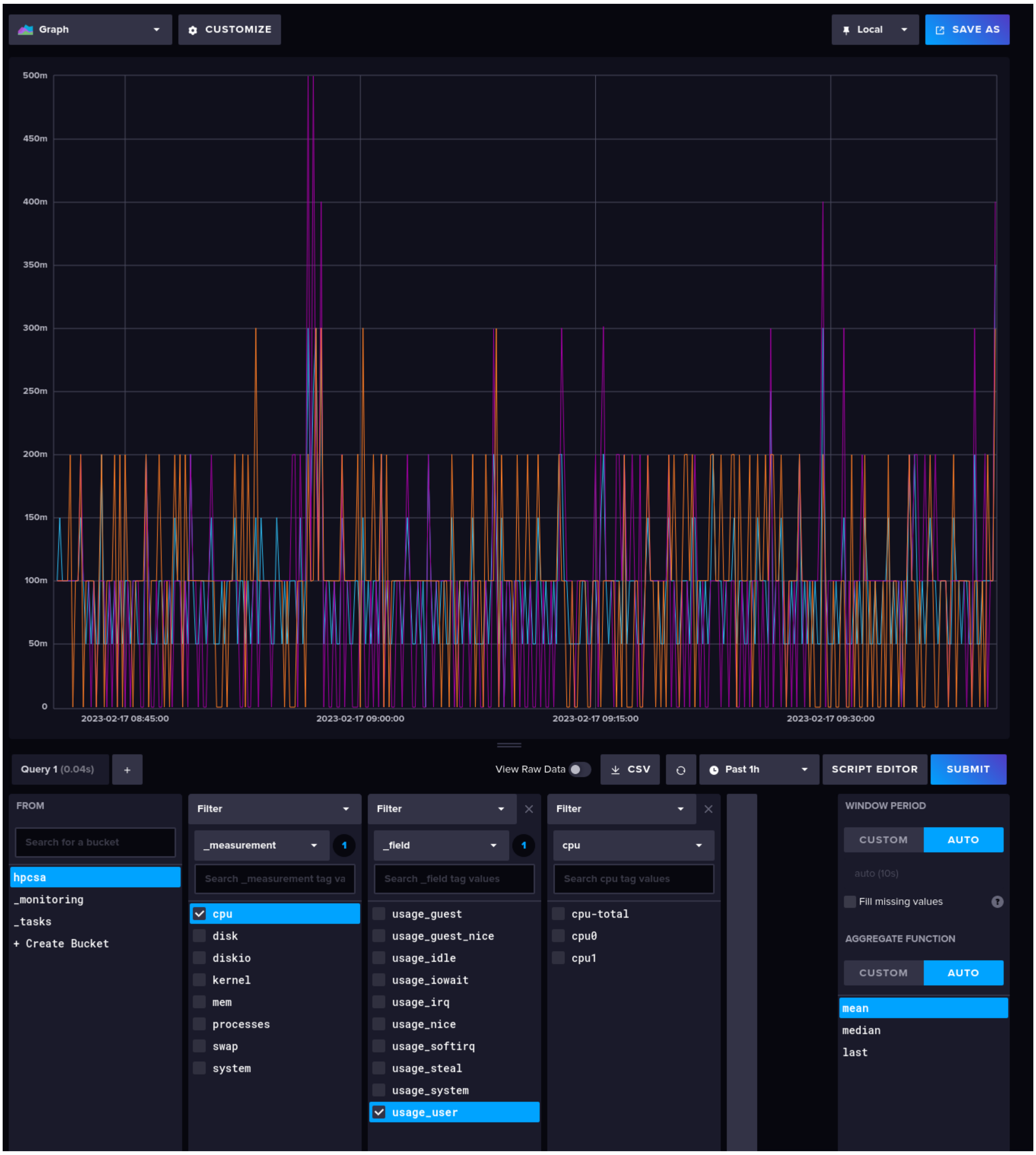
This web interface can be reached via web browser on $< frontendfloatingip:< influxport >$.

Login is $< influxuser >$ and $< influxpassword >$.

Select the dataexplorer from the toolbar on the left:

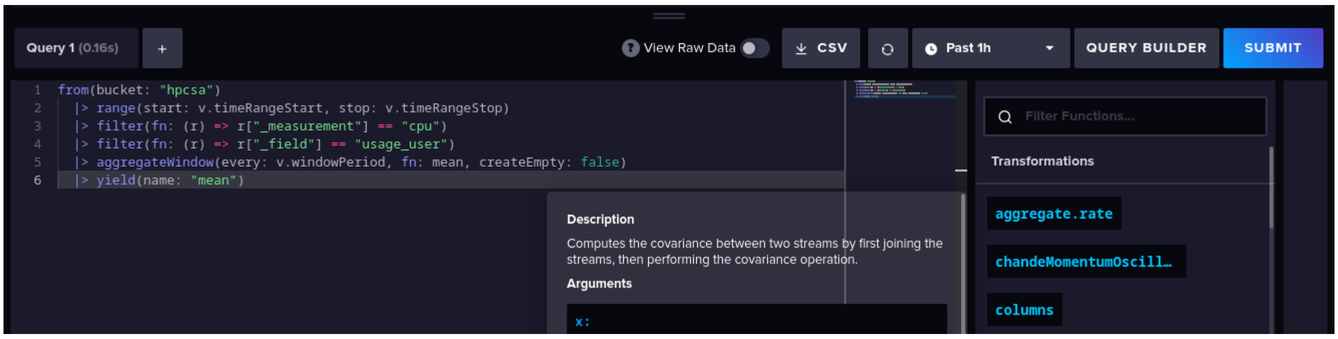


This will open the dataexplorer. At the bottom are columns letting users select, the bucket, measurements, according metrics - with this it is easy to dig down through the data in InfluxDB.



To execute a query press the "Submit" button. This will build the graph accordingly.

In order to get the script for this query to use in other tools, e.g. Grafana, the script editor has to be openened by pressing "Script Editor".



```
from(bucket: "hpcsa")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "cpu")
  |> filter(fn: (r) => r["_field"] == "usage_user")
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
  |> yield(name: "mean")
```

Description

Computes the covariance between two streams by first joining the streams, then performing the covariance operation.

Arguments

x:

To use the script displayed there in Grafana just mark the script and copy/paste it to the query editor in the panel creation in Grafana.

Optional: Install another Telegraf Plugin 5: Tutorial (0 min)

This requires a working TIG Stack as it has been installed in the previous tasks.

Check the available plugins on the according webpage https://docs.influxdata.com/telegraf/v1.20/plugins/ , select one of them, install and configure it.

Optional: Install telegraf on the worker nodes 6: Tutorial (0 min)

Requires at least the PXE booting worker nodes setup before.

Two worker nodes have been set up during previous sessions. Install telegraf on those machines and integrate them into a dashboard.

Optional: Setup Database and Grafana with specific users 7: Tutorial (0 min)

Requires a basic TIG stack running as setup in the first 3 tasks.

Recap the previous installation. Just one main user has been used to access the database and the Grafana dashboard. Usually this is considered bad practice in regards to security. Setup users for the InfluxDB and Grafana and modify all parts of the stack accordingly.

Hints

- Think about the following: Wow many users would be useful for the InfluxDB and for Grafana? What rights should they have? Why?
- Do not forget to change parts the telegraf configuration.

Optional: Central setup for telegraf 8: Tutorial (0 min)

Requires the working TIG stack from this tutorial and telegraf already integrated on the worker nodes.

In a previous lecture slurm has been setup using a central installation and configuration. Is this possible for telegraf, too? Or has something to be considered and adapted to telegraf. Implement you solution.

Hints

- HInt for tutor: you may have different hardware installed which utilize different plugins. Therefor at least the configuration has to be local for the specific machine types/hardware.

Further Reading

- https://grafana.com/docs/grafana/latest/getting-started/get-started-grafana-influxdb/
- https://grafana.com/docs/grafana/latest/introduction/
- https://docs.influxdata.com/influxdb/v2.6/install/
- https://docs.influxdata.com/telegraf/v1.20/plugins/