Grafana

- Overview
- A short video
- Development

THIS IS WORK IN PROGRESS AND CAN CHANGE ANY TIME 😉

Overview

At the moment we are developing a simple plugin for odysseus.

To try the current state you could use the following docker file:

```
FROM node:14 as node-base
RUN echo "Installing required packages..." && \
        apt-get update && \
        apt-get install -yq --no-install-recommends python
RUN echo "Cloning Odysseus Grafana plugin source..." && \
        git clone https://git.swl.informatik.uni-oldenburg.de/scm/api_apps/grafana.git /var/lib/ody_grafana
WORKDIR /var/lib/ody_grafana/grafana-plugins/odysseus
RUN mkdir build
RUN echo "Building Odysseus Grafana plugin binaries..." && \
        yarn install && \
        yarn dev

FROM grafana/grafana:7.4.3 AS grafana
RUN echo "Copying Odysseus Grafana plugin binaries..."
COPY --from=node-base /var/lib/ody_grafana/grafana-plugins/odysseus /var/lib/grafana/plugins/ody-streaming-
datasource
WORKDIR /usr/share/grafana
USER grafana
ENTRYPOINT [ "/run.sh" ]
```

Use this:

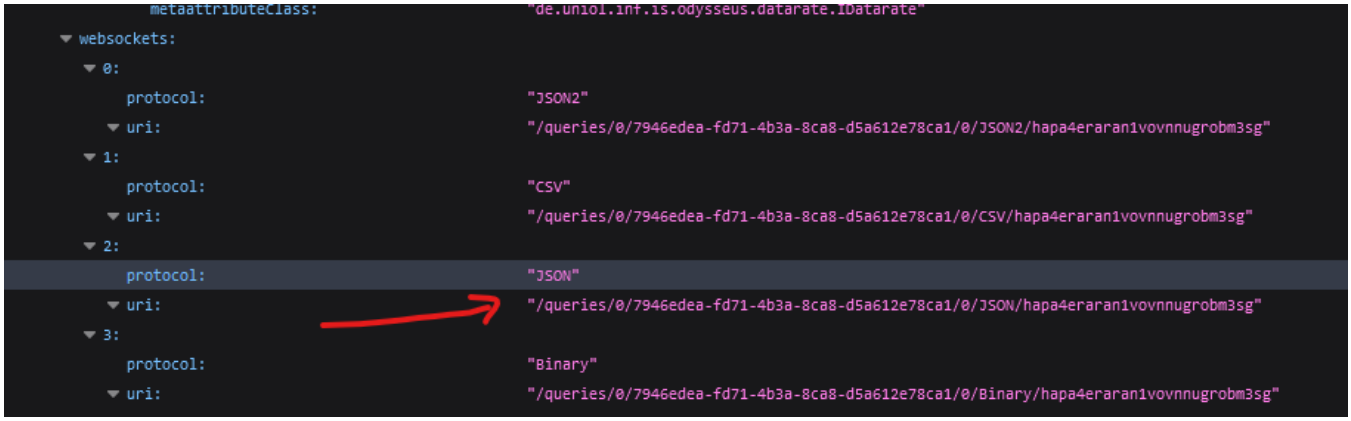
```
docker build . -t grafana
docker run -d -p 3000:3000 --name grafana grafana
```

Then you could use Odysseus in grafana as a datasource.

You will need to provide the server-adress: e.g. with ws://localhost:8888/

ATM there are some shortcommings that will be removed in future versions:

You will need to determine the websocket address manually by using a webbrowser (we recommend Firefox because of better JSON presentation) (http://localhost:8888/queries/<nr>) and find the section websockets with protocol JSON and use the entry with uri for creating panels in grafana:



The output of the connected query

- should contain an attribute named time with a milliseconds timestamp (typically this should be a current timestamp for grafana)
- all other attributes should be numeric attributes

A short video

Your browser does not support the HTML5 video element

The video can also be found here https://odysseus.informatik.uni-oldenburg.de/download/documentation/videos/grafana/odysseus_grafana_tutorial.mp4

In addition, here is a brief READ.ME.

```
We need a grafana plugin to be able to receive data from Odysseus.
Note that the current state of the plugin is experimental.

Wiki-Page: https://wiki.odysseus.informatik.uni-oldenburg.de/display/ODYSSEUS/Grafana

Requirements: docker

1. build a docker image based on the dockerfile in this folder
docker build . -t grafana

2. build and run a docker container based on the created image
docker run -d -p 3000:3000 --name grafana grafana

3. open grafana in a browser
localhost:3000
username: admin
password: admin

4. Add an Odysseus datasource
Requirements: Running Odysseus Server (or monolithic of course)
Configuration -> Data Sources
Select Others -> odysseus
Server URI: ws://localhost:8888/ (REST-port of a (here local) odysseus server)
Leave API Key empty

5. Create Dashboard
Create -> Dashboard

6. Create Panel
Requirements:
* To visualize single datastream attributes, there must be an operator that provides only the current timestamp
named "time" and that attribute
* To easily find the operator, give him a proper name
* Query(s) should be installed
Tipps:
a) At the end of the query, create for each attribute you want to visualize a projection on "time" and the
attribute
b) Grafana can display historical values but to show the results in real-time, the timestamps of the results
should be set to the current time (necessary if you, e.g., read from a CSV file with old timestamps)
6.1. select your Odysseus datasource as datasource
6.2. you need a websocket that provides the data
6.3. open in a browser localhost:8888/queries to get information about all installed queries (in chrome it is
not formatted; copy the content in, e.g., https://jsonformatter.curiousconcept.com/)
6.4. find the websocket in the json
go to the query id of your query -> rootOperators -> go to the operator providing the data to visualize ->
ports -> websockets -> copy the JSON uri
6.5. Insert the copied URI in grafana into the websocket field of the panel query
6.6. Click on apply and after that on save dashboard
```

Development

See https://grafana.com/docs/grafana/latest/developers/plugins/

and https://grafana.com/docs/grafana/latest/developers/plugins/build-a-streaming-data-source-plugin/ for further information how to extend the current implementation.

Quick start.

- Install e.g. MS Code for development
- Install anything necessary for grafana plugin development
- create a folder named /grafana-plugins
- inside this folder clone the plugin
- switch into plugin folder and run:

```
yarn install
yarn dev
```

- now start grafana e.g. with

```
docker run -d -e GF_DEFAULT_APP_MODE=development -p 3000:3000 -v "$(pwd)"/grafana-plugins:/var/lib
/grafana/plugins --name=grafana grafana/grafana:latest
```

- If you change anything inside the plugin, grafana needs to be restart. You could create a script:

```
docker stop grafana
docker rm grafana
docker run -d -e GF_DEFAULT_APP_MODE=development -p 3333:3000 -v "$(pwd)"/grafana-plugins:/var/lib
/grafana/plugins --name=grafana grafana/grafana:latest
```