Grafana Setup Documentation

Task requirement:

"Setting Up Monitoring Infrastructure with Grafana, Prometheus, and Node Exporter".

Monitoring :

Regular collection of information and data to measure progress of projects and activities so we can track performance and resources utilisation over time.

System Configuration:

- OS Name : Ubuntu 20.04.6 LTS
- Podman version:- 3.4.2
- RAM : 5.6 GiB
- CPU : 12
- STORAGE : 512.1 GB

Prerequisite tools:

- Podman
- Grafana
- Prometheus
- Node-exporter

Definition of tools

Podman:

Podman is an open source tool for developing, managing, and running containers
on your Linux® system

Grafana :

Grafana open-source software enables you to query, visualise, alert on, and explore your metrics, logs, and traces wherever they are stored.

- Query
- Visualise
- Alert

Prometheus :

Prometheus is an open source monitoring solution written in Go that collects metrics data and stores that data in a time series database.
Grafana allows to visualise the data stored in prometheus.

Node-exporter:

Node Exporter is a software tool that collects system-level metrics from servers and nodes, exposing them in a format that can be scraped by Prometheus for monitoring and analysis.

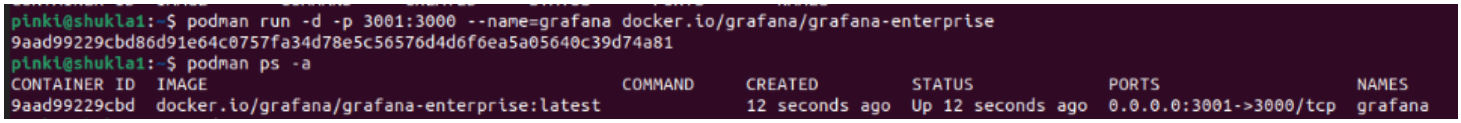
Node Exporter collects a wide range of system metrics, including CPU, usage Memory, usage Disk usage, System load,Filesystem statistics.

Installation Process:

Step-1 Create a Grafana container:

To run the latest stable version of Grafana, run the following command:

```
podman run -d -p 3001:3000 --name=grafana docker.io/grafana/grafana-enterprise
```

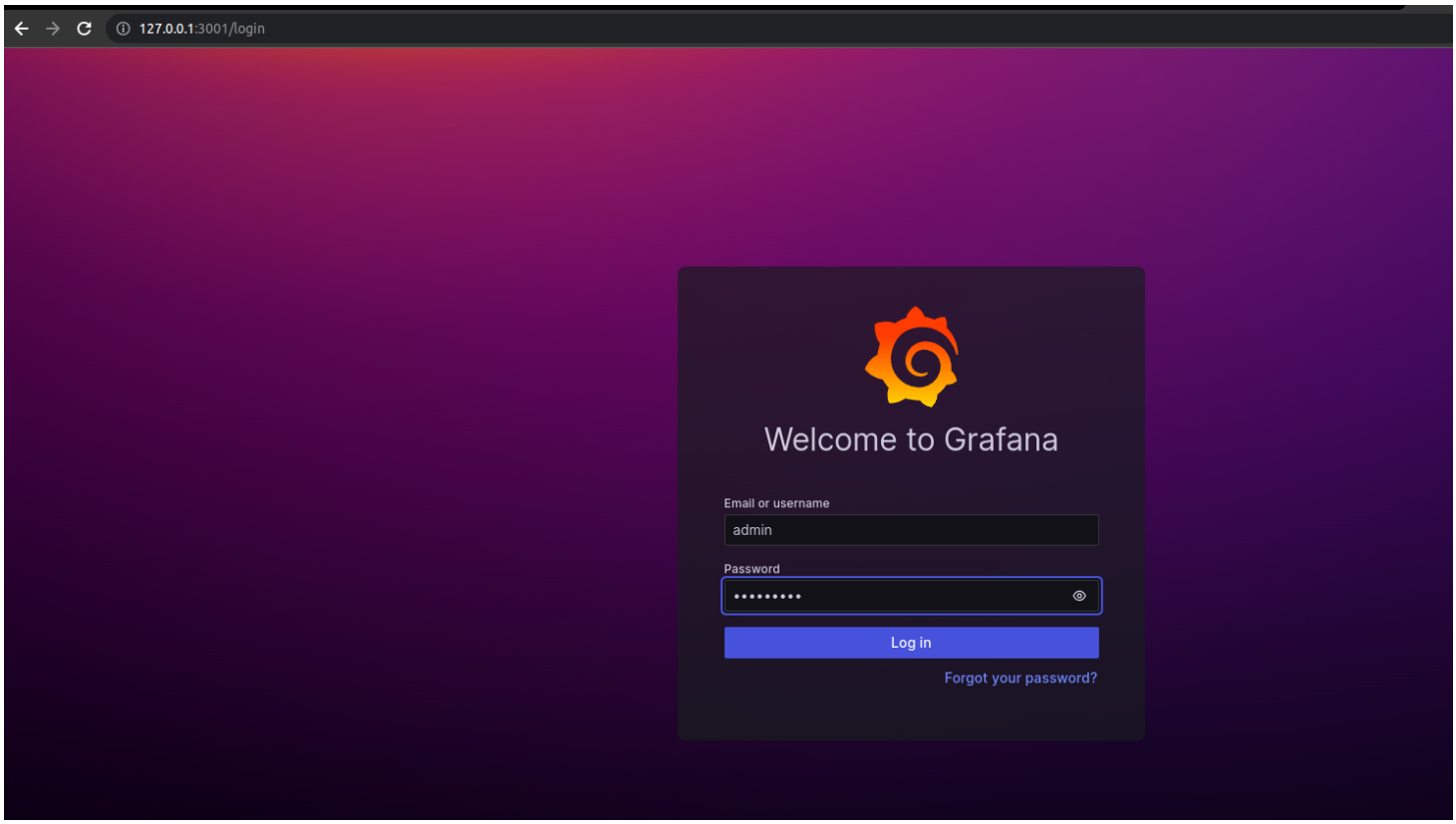


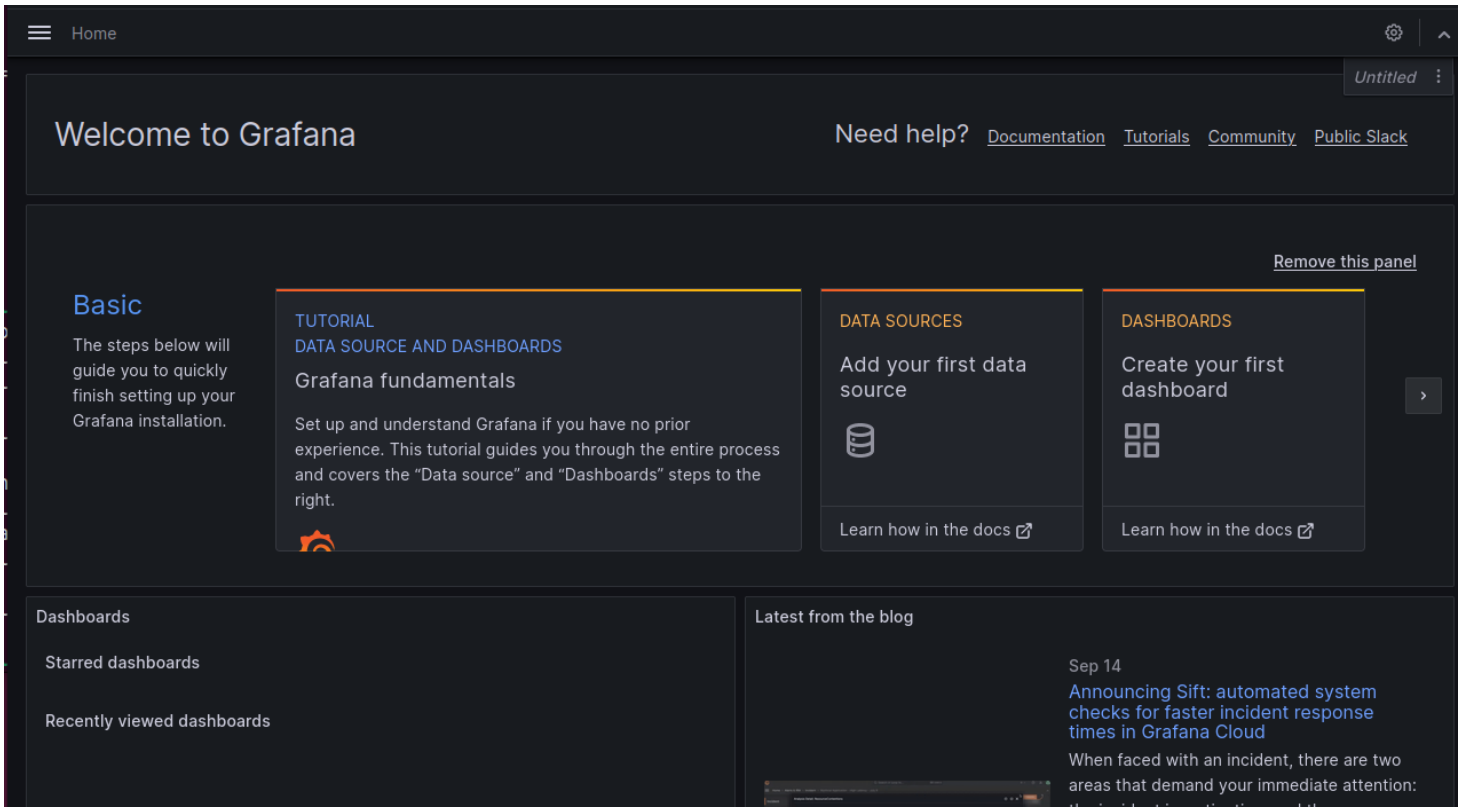
Where:

run = run directly from the command line
d = run in the background
p = assign the port number, which in this case is 3001
name = assign a name to the container, for example, grafana

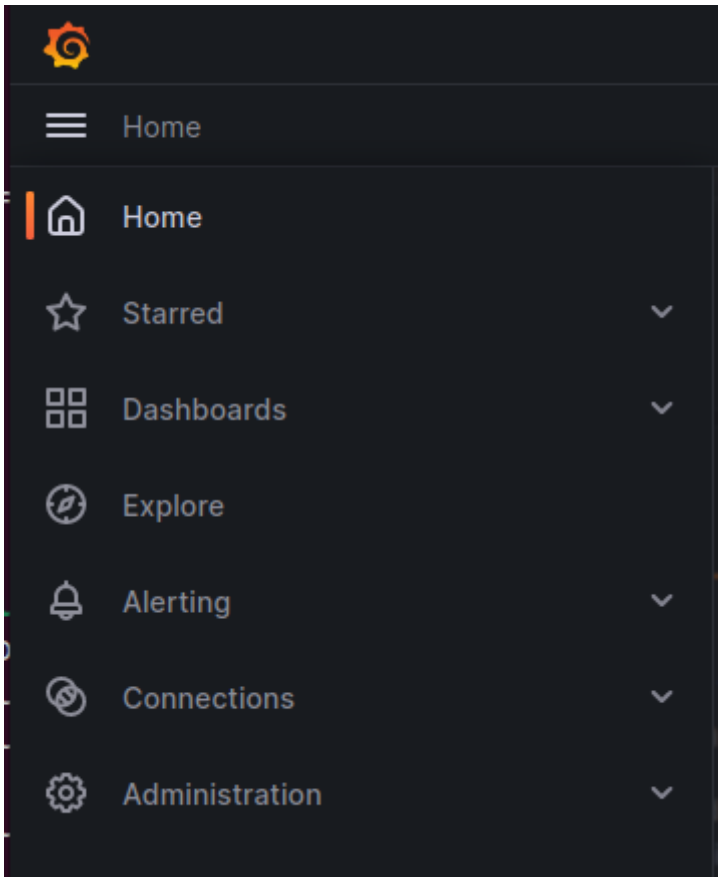
docker.io/grafana/grafana-enterprise: This is the name of the image we want to run inside the container. It's called 'grafana-enterprise,' and it's stored in a special place on the internet called 'Docker Hub.'

http://localhost:3001 hit on web browser to see Grafana login page in your web browser.


127.0.0.1:3001/login
Welcome to Grafana
Email or username
admin
Password
Log in
Forgot your password?


Home
Untitled
Welcome to Grafana
Need help?
Documentation
Tutorials
Community
Public Slack
Remove this panel
Basic
The steps below will guide you to quickly finish setting up your Grafana installation.
TUTORIAL
DATA SOURCE AND DASHBOARDS
Grafana fundamentals
Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.
DATA SOURCES
Add your first data source
Learn how in the docs
DASHBOARDS
Create your first dashboard
Learn how in the docs
Dashboards
Starred dashboards
Recently viewed dashboards
Latest from the blog
Sep 14
Announcing Sift: automated system checks for faster incident response times in Grafana Cloud
When faced with an incident, there are two areas that demand your immediate attention:

Use these options to create dashboards and alerts.



Step-2 Create Prometheus container on Podman:

Create directory :-

```
mkdir prometheus
```

Create file:-

```
vim prometheus.yml
```

prometheus.yml is a configuration file of prometheus.

(push all the data in your prometheus.yml file which has been given below)

```
global:
  scrape_interval: 5s
  external_labels:
        monitor: 'node'

scrape_configs:
  - job_name: 'prometheus'
        static_configs:
        - targets: ['192.168.1.57:9090']

  - job_name: 'node-exporter'
        static_configs:
        - targets: ['192.168.1.57:9100']
```

```
podman run -d --name prometheus -p 9090:9090 -v /home/pinki/prometheus/prometheus.yml:/etc/prometheus
```

```
pinki@shukla1:~/prometheus$ podman run -d --name prometheus -p 9090:9090 -v /home/pinki/prometheus/prometheus.yml:/etc/prometheus/prometheus.yml docker.io/prom/prometheus
f25c542682245621fb392fcc4a5142d68847f4625c8dbb190d83a9239755fbd11
```

podman run: This part of the command instructs podman to run a container.

d: This flag stands for "detached" mode. It runs the container in the background

--name prometheus: This flag assigns a name to the container. In this case, the container is named "prometheus."

-p 9090:9090: This flag specifies port mapping. It tells podman to map port 9090 on the host to port 9090 inside the container.

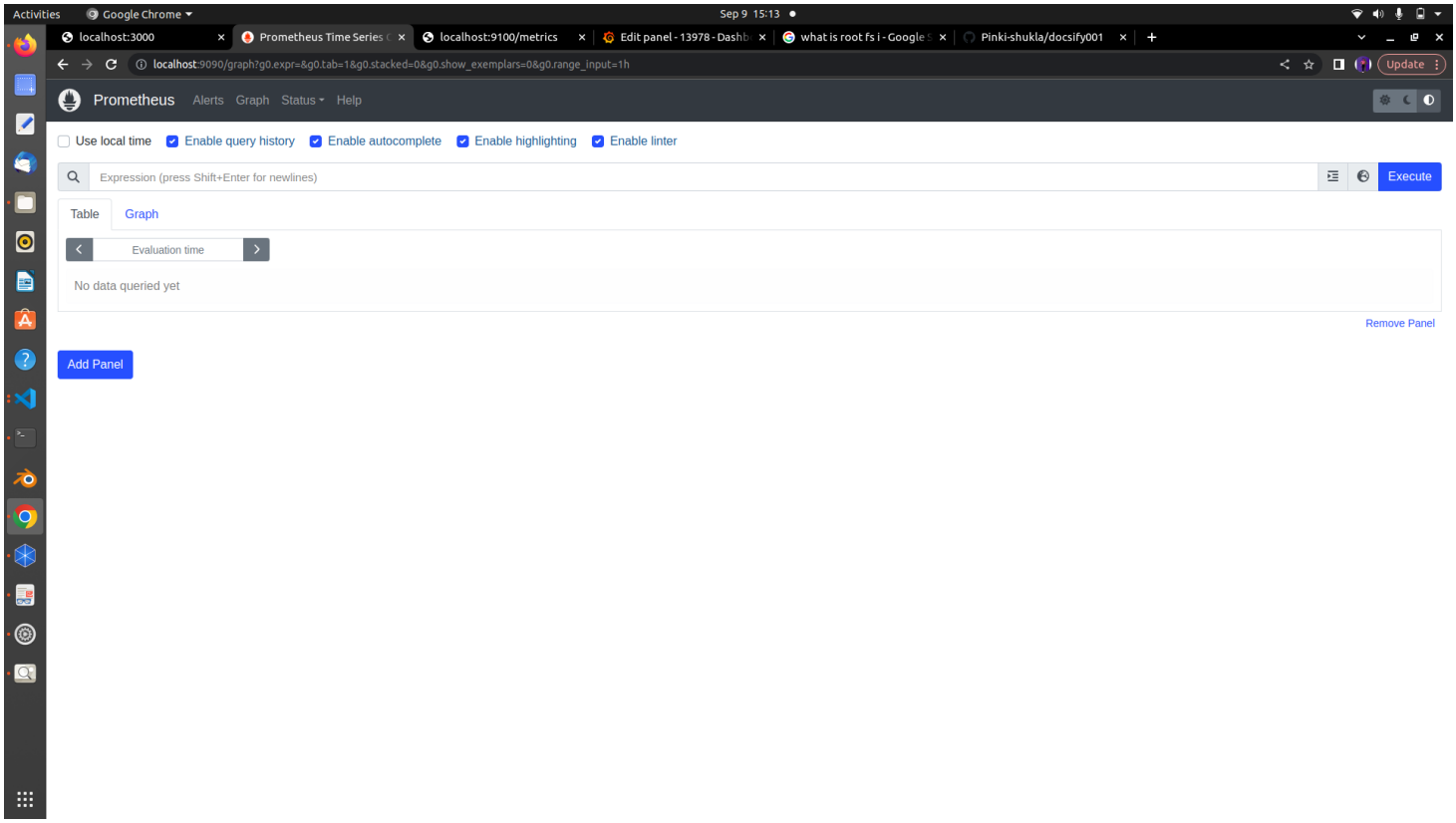
-v /home/amit/prometheus/prometheus.yml:/etc/prometheus/prometheus.yml: This flag specifies a volume mount. It connects a directory or file on your host system to a location inside the container.

In this case, it's mounting the file /home/pinki/prometheus/prometheus.yml from your host into the container at /etc/prometheus/prometheus.yml

- **" /home/pinki/prometheus/prometheus.yml":** This is the path to the Prometheus configuration file on your host. It's being shared with the container.
- **"/etc/prometheus/prometheus.yml":** This is where Prometheus expects its configuration file to be inside the container.

docker.io/prom/prometheus: This is the name of the Docker image you want to run as a container. It specifies the image's repository and name. In this case, you are running the "prometheus" image from the "prom" repository on Docker Hub

http://localhost:9090 on web browser to see prometheus.



Step-3 Create Node exporter container on podman

```
podman run -d --name=node-exporter -p 9100:9100 -v"/:/host:ro,rslave" quay.io/prometheus/node-exporter:
```

```
valid_lft forever preferred_lft forever
pinki@shukla1:~/prometheus$ podman run -d --name=node-exporter -p 9100:9100 -v"/:/host:ro,rslave" quay.io/prometheus/node-exporter:latest --path.rootfs=/host
Trying to pull quay.io/prometheus/node-exporter:latest...
Getting image source signatures
Copying blob d5c4df21b127 done
Copying blob 2b6642e6c59e done
Copying blob 2f5f7d8898a1 done
Copying config 458e026e6a done
Writing manifest to image destination
Storing signatures
2d7fdbbd57fa82d23995be1742baaf5ca82ebc45a8f5f2e0609a18209322abe5
```

podman run: This part of the command instructs podman to run a container.

-d: This flag stands for "detached" mode.

--name=node-exporter: This flag assigns a name to the container. In this case, the container is named "node-exporter."

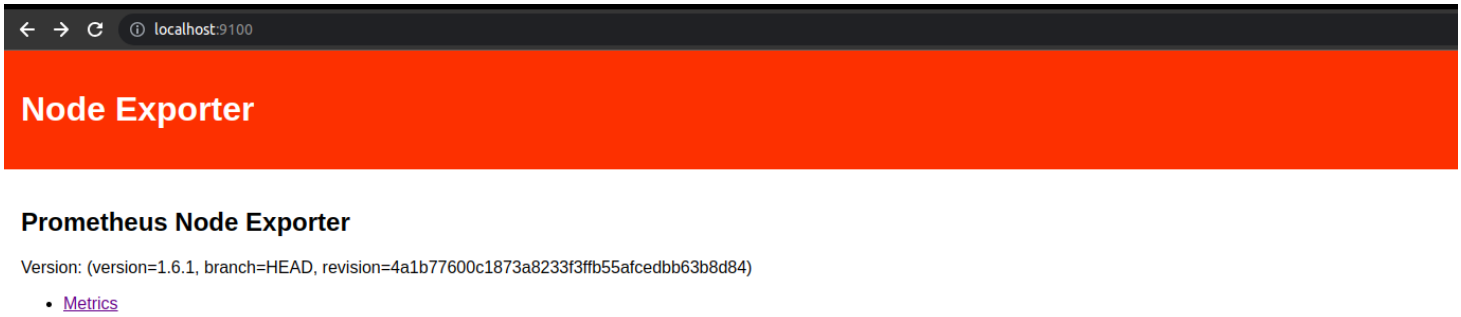
-p 9100:9100: This flag specifies port mapping. it allows you to access the service inside the container via port 9100 on your host.

"/:/host:ro,rslave": This part specifies the volume configuration. It tells podman to mount the root directory of your host (represented by **"/"**) to the **"/host"** directory inside the container. The **"ro"** option stands for "read-only," which means the container can read the files on the host but can't modify them. The rslave option is related to mount propagation, allowing mounted file systems to be shared among containers.

quay.io/prometheus/node-exporter:latest: In this case, you are running the "node-exporter" image from the "prometheus" repository on Quay.io. The ":latest" tag indicates that you want to use the latest version of this image.

--path.rootfs=/host: This is an additional command passed to the container. It specifies the root file system path as "/host" inside the container. This can be important for some containerized applications to correctly access system resources.

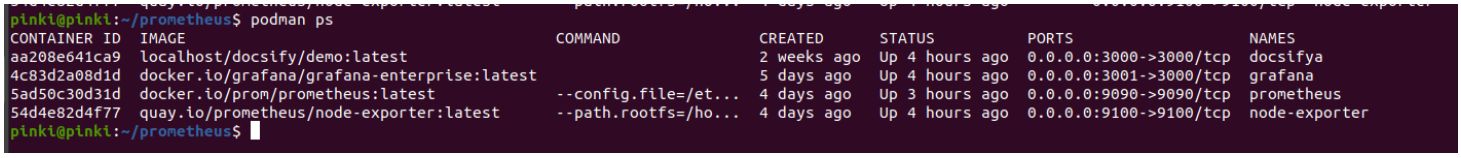
http://localhost:9100 on web browser to see Node-expoerter.



Now you can see our all containers are ready grafana,prometheus and node-exporter

```
podman ps
```

The **podman ps** command is used to list the currently running containers on your system. It provides information about the containers that are actively running and includes details such as the container ID, names, status, and other relevant information.



Grafana setup has been ready with Dashboard.