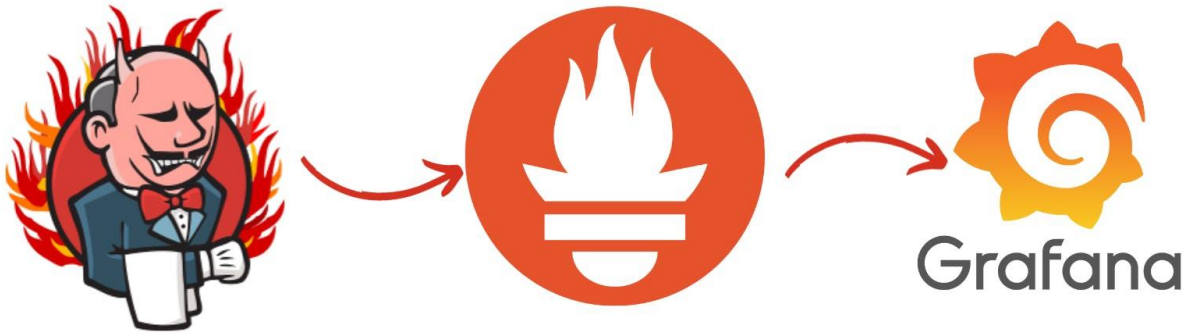
How to Monitor Jenkins Using Prometheus, Node exporter and Grafana



Prometheus:

Prometheus is an open-source monitoring and alerting system used to collect and store metrics from various sources. It is designed to monitor highly dynamic environments like cloud-native applications or microservices. Prometheus allows you to track and analyze the performance and health of your applications and infrastructure.

Use case:

Prometheus can be used to monitor the resource utilization of servers, track the response time of web services, collect metrics from databases, and measure the performance of containerized applications.

Node Exporter:

Node Exporter is a Prometheus exporter specifically designed to gather system-level metrics from a target machine. It runs on the machine you want to monitor and exposes various metrics like CPU usage, memory usage, disk utilization, network statistics, and more. These metrics are then scraped by Prometheus for further analysis.

Use case:

Node Exporter is commonly used to monitor the health and performance of individual servers or nodes in a cluster. It helps identify resource bottlenecks, detect hardware failures, and optimize resource allocation.

Grafana:

Grafana is an open-source data visualization tool that works seamlessly with Prometheus and other data sources. It allows you to create interactive and customizable dashboards to visualize metrics collected by Prometheus or other monitoring systems. Grafana provides a wide range of visualizations and supports various data sources, enabling you to monitor and analyze your data effectively.

Use case:

Grafana is useful for creating real-time monitoring dashboards, generating meaningful visualizations, and setting up alerting rules based on metric thresholds. It helps in gaining insights into system performance, identifying anomalies, and sharing visual reports with teams or stakeholders.

To summarize, Prometheus is a monitoring system that collects metrics, Node Exporter is used to gather system-level metrics from

individual machines, and Grafana helps visualize and analyze the collected data in the form of interactive dashboards. Together, these tools provide a powerful monitoring and visualization stack for tracking the performance and health of applications and infrastructure.

Install Prometheus on Ubuntu 22.04

First of all, let's create a dedicated Linux user sometimes called a system account for Prometheus. Having individual users for each service serves two main purposes:

It is a security measure to reduce the impact in case of an incident with the service.

It simplifies administration as it becomes easier to track down what resources belong to which service.

To create a system user or system account, run the following command:

```
sudo useradd \
    --system \
    --no-create-home \
    --shell /bin/false prometheus
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo useradd \
    --system \
    --no-create-home \
    --shell /bin/false prometheus
ubuntu@ip-172-31-38-156:~$
```

--system - Will create a system account.
--no-create-home - We don't need a home directory for Prometheus or any other system accounts in our case.
--shell /bin/false - It prevents logging in as a Prometheus user.

Prometheus - Will create a Prometheus user and a group with the same name.

Let's check the latest version of Prometheus from the download page.

https://prometheus.io/download/

You can use the curl or wget command to download Prometheus.

```
wget https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ wget https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz
--2023-10-06 08:51:59--  https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/6838921/2f9b7b37-63a0-428b-adb5-0294482fd743?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credentials
IAIWNJYAX4CSVEH53A%2F20231006%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20231006T085159Z&X-Amz-Expires=300&X-Amz-Signature=6962d584d4eaad7d1a082e0285936e53a7a6976aa6f71b5d51765dbfe
6a1c&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=6838921&response-content-disposition=attachment%3B%20filename%3Dprometheus-2.47.1.linux-amd64.tar.gz&response-content-type=a
lication%2Foctet-stream [following]
--2023-10-06 08:51:59--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/6838921/2f9b7b37-63a0-428b-adb5-0294482fd743?X-Amz-Algorithm=AWS4-HMAC-SHA256&X
z-Credential=AKIAIWNJYAX4CSVEH53A%2F20231006%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20231006T085159Z&X-Amz-Expires=300&X-Amz-Signature=6962d584d4eaad7d1a082e0285936e53a7a6976aa6
b5d51765dbfe36e6a1c&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=6838921&response-content-disposition=attachment%3B%20filename%3Dprometheus-2.47.1.linux-amd64.tar.gz&response
ontent-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 95713066 (91M) [application/octet-stream]
Saving to: 'prometheus-2.47.1.linux-amd64.tar.gz'

prometheus-2.47.1.linux-amd64.tar.gz          100%[===================================================================================================>]  91.28M  4.01MB/s    in 88s

2023-10-06 08:53:28 (1.03 MB/s) - 'prometheus-2.47.1.linux-amd64.tar.gz' saved [95713066/95713066]

ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ ls
prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$
```

Then, we need to extract all Prometheus files from the archive.

```
tar -xvf prometheus-2.47.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$ ls
prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ tar -xvf prometheus-2.47.1.linux-amd64.tar.gz
prometheus-2.47.1.linux-amd64/
prometheus-2.47.1.linux-amd64/LICENSE
prometheus-2.47.1.linux-amd64/NOTICE
prometheus-2.47.1.linux-amd64/prometheus.yml
prometheus-2.47.1.linux-amd64/consoles/
prometheus-2.47.1.linux-amd64/consoles/prometheus.html
prometheus-2.47.1.linux-amd64/consoles/prometheus-overview.html
prometheus-2.47.1.linux-amd64/consoles/node-cpu.html
prometheus-2.47.1.linux-amd64/consoles/index.html.example
prometheus-2.47.1.linux-amd64/consoles/node.html
prometheus-2.47.1.linux-amd64/consoles/node-disk.html
prometheus-2.47.1.linux-amd64/consoles/node-overview.html
prometheus-2.47.1.linux-amd64/promtool
prometheus-2.47.1.linux-amd64/console_libraries/
prometheus-2.47.1.linux-amd64/console_libraries/prom.lib
prometheus-2.47.1.linux-amd64/console_libraries/menu.lib
prometheus-2.47.1.linux-amd64/prometheus
ubuntu@ip-172-31-38-156:~$
```

Usually, you would have a disk mounted to the data directory. For this tutorial, I will simply create a /data directory. Also, you need a folder for Prometheus configuration files.

```
sudo mkdir -p /data /etc/prometheus
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo mkdir -p /data /etc/prometheus
ubuntu@ip-172-31-38-156:~$
```

Now, let's change the directory to Prometheus and move some files.

```
cd prometheus-2.47.1.linux-amd64/
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ cd prometheus-2.47.1.linux-amd64/
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ ls -l
total 236852
-rw-r--r-- 1 ubuntu ubuntu     11357 Oct  4 11:05 LICENSE
-rw-r--r-- 1 ubuntu ubuntu      3773 Oct  4 11:05 NOTICE
drwxr-xr-x 2 ubuntu ubuntu      4096 Oct  4 11:05 console_libraries
drwxr-xr-x 2 ubuntu ubuntu      4096 Oct  4 11:05 consoles
-rwxr-xr-x 1 ubuntu ubuntu 124158156 Oct  4 10:35 prometheus
-rw-r--r-- 1 ubuntu ubuntu       934 Oct  4 11:05 prometheus.yml
-rwxr-xr-x 1 ubuntu ubuntu 118343283 Oct  4 10:38 promtool
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

First of all, let's move the Prometheus binary and a promtool to the /usr/local/bin/. promtool is used to check configuration files and Prometheus rules.

```
sudo mv prometheus promtool /usr/local/bin/
```

```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo mv prometheus promtool /usr/local/bin/
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

Optionally, we can move console libraries to the Prometheus configuration directory. Console templates allow for the creation of arbitrary consoles using the Go templating language. You don't need to worry about it if you're just getting started.

```
sudo mv consoles/ console_libraries/ /etc/prometheus/
```

```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo mv consoles/ console_libraries/ /etc/prometheus/
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

Finally, let's move the example of the main Prometheus configuration file.

```
sudo mv prometheus.yml /etc/prometheus/prometheus.yml
```

```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo mv prometheus.yml /etc/prometheus/prometheus.yml
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

To avoid permission issues, you need to set the correct ownership for the /etc/prometheus/ and data directory.

```
sudo chown -R prometheus:prometheus /etc/prometheus/ /data/
```

```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo chown -R prometheus:prometheus /etc/prometheus/ /data/
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

You can delete the archive and a Prometheus folder when you are done.

```
cd
rm -rf prometheus-2.47.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ cd ..
ubuntu@ip-172-31-38-156:~$ ll
total 93516
drwxr-x--- 5 ubuntu ubuntu     4096 Oct  6 08:54 ./
drwxr-xr-x 3 root   root       4096 Oct  6 08:49 ../
-rw------- 1 ubuntu ubuntu       62 Oct  6 08:50 .Xauthority
-rw-r--r-- 1 ubuntu ubuntu      220 Jan  6  2022 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu     3771 Jan  6  2022 .bashrc
drwx------ 2 ubuntu ubuntu     4096 Oct  6 08:50 .cache/
-rw-r--r-- 1 ubuntu ubuntu      807 Jan  6  2022 .profile
drwx------ 2 ubuntu ubuntu     4096 Oct  6 08:49 .ssh/
-rw-r--r-- 1 ubuntu ubuntu        0 Oct  6 08:50 .sudo_as_admin_successful
-rw-rw-r-- 1 ubuntu ubuntu      165 Oct  6 08:53 .wget-hsts
drwxr-xr-x 2 ubuntu ubuntu     4096 Oct  6 08:56 prometheus-2.47.1.linux-amd64/
-rw-rw-r-- 1 ubuntu ubuntu 95713066 Oct  4 11:15 prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ rm -rf prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ ls
prometheus-2.47.1.linux-amd64
ubuntu@ip-172-31-38-156:~$
```

Verify that you can execute the Prometheus binary by running the following command:

```
prometheus --version
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ prometheus --version
prometheus, version 2.47.1 (branch: HEAD, revision: c4d1a8beff37cc004f1dc4ab9d2e73193f51aaeb)
  build user:       root@4829330363be
  build date:       20231004-10:31:16
  go version:       go1.21.1
  platform:         linux/amd64
  tags:             netgo,builtinassets,stringlabels
ubuntu@ip-172-31-38-156:~$
```

To get more information and configuration options, run Prometheus Help.

```
prometheus --help
```

We're going to use some of these options in the service definition.

We're going to use Systemd, which is a system and service manager for Linux operating systems. For that, we need to create a Systemd unit configuration file.

```
sudo vim /etc/systemd/system/prometheus.service
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/systemd/system/prometheus.service
```

Prometheus.service

```
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

StartLimitIntervalSec=500
StartLimitBurst=5

[Service]
User=prometheus
Group=prometheus
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/prometheus \
  --config.file=/etc/prometheus/prometheus.yml \
  --storage.tsdb.path=/data \
  --web.console.templates=/etc/prometheus/consoles \
  --web.console.libraries=/etc/prometheus/console_libraries \
  --web.listen-address=0.0.0.0:9090 \
  --web.enable-lifecycle

[Install]
WantedBy=multi-user.target
```

```
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

StartLimitIntervalSec=500
StartLimitBurst=5

[Service]
User=prometheus
Group=prometheus
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/prometheus \
  --config.file=/etc/prometheus/prometheus.yml \
  --storage.tsdb.path=/data \
  --web.console.templates=/etc/prometheus/consoles \
  --web.console.libraries=/etc/prometheus/console_libraries \
  --web.listen-address=0.0.0.0:9090 \
  --web.enable-lifecycle

[Install]
WantedBy=multi-user.target
```

Let's go over a few of the most important options related to Systemd and Prometheus. Restart - Configures whether the service shall be restarted when the service process exits, is killed, or a timeout is reached.
RestartSec - Configures the time to sleep before restarting a service.
User and Group - Are Linux user and a group to start a Prometheus process.
--config.file=/etc/prometheus/prometheus.yml - Path to the main Prometheus configuration file.
--storage.tsdb.path=/data - Location to store Prometheus data.
--web.listen-address=0.0.0.0:9090 - Configure to listen on all network interfaces. In some situations, you may have a proxy such as nginx to redirect requests to Prometheus. In that case, you would configure Prometheus to listen only on localhost.
--web.enable-lifecycle -- Allows to manage Prometheus, for example, to reload configuration without restarting the service.

To automatically start the Prometheus after reboot, run enable.

```
sudo systemctl enable prometheus
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl enable prometheus
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
```

Then just start the Prometheus.

```
sudo systemctl start prometheus
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl start prometheus
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
```

To check the status of Prometheus run the following command:

```
sudo systemctl status Prometheus
```

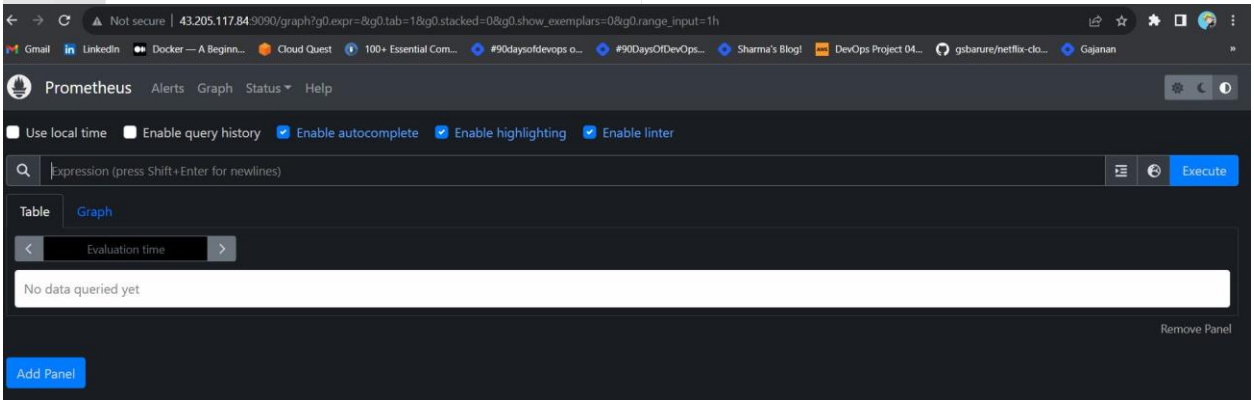
```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl status prometheus
● prometheus.service - Prometheus
     Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-10-06 09:00:39 UTC; 13s ago
   Main PID: 1941 (prometheus)
      Tasks: 6 (limit: 1141)
     Memory: 16.2M
        CPU: 64ms
     CGroup: /system.slice/prometheus.service
             └─1941 /usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/data --web.console.templates=/etc/prometheus/consoles --web.console.lib

Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=tls_config.go:274 level=info component=web msg="Listening on" address=[::]:9090
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=tls_config.go:277 level=info component=web msg="TLS is disabled." http2=false address=[::]:9090
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=head.go:760 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSegment=0
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=head.go:797 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=42.05
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.801Z caller=main.go:1045 level=info fs_type=EXT4_SUPER_MAGIC
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.801Z caller=main.go:1048 level=info msg="TSDB started"
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.801Z caller=main.go:1229 level=info msg="Loading configuration file" filename=/etc/prometheus/prometheus.yml
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.805Z caller=main.go:1266 level=info msg="Completed loading of configuration file" filename=/etc/prometheus/pr
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.805Z caller=main.go:1009 level=info msg="Server is ready to receive web requests."
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.805Z caller=manager.go:1009 level=info component="rule manager" msg="Starting rule manager..."
lines 1-20/20 (END)
```

Suppose you encounter any issues with Prometheus or are unable to start it. The easiest way to find the problem is to use the journalctl command and search for errors.

```
journalctl -u prometheus -f --no-pager
```

Now we can try to access it via the browser. I'm going to be using the IP address of the Ubuntu server. You need to append port 9090 to the IP.

```
<public-ip:9090>
```



If you go to targets, you should see only one - Prometheus target. It scrapes itself every 15 seconds by default.

Install Node Exporter on Ubuntu 22.04

Next, we're going to set up and configure Node Exporter to collect Linux system metrics like CPU load and disk I/O. Node Exporter will expose these as Prometheus-style metrics. Since the installation process is very similar, I'm not going to cover as deep as Prometheus.

First, let's create a system user for Node Exporter by running the following command:

```
sudo useradd \
    --system \
    --no-create-home \
    --shell /bin/false node_exporter
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo useradd \
    --system \
    --no-create-home \
    --shell /bin/false node_exporter
ubuntu@ip-172-31-38-156:~$
```

You can **download Node Exporter** from here

https://prometheus.io/download/

Use the wget command to download the binary.

```
wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
--2023-10-06 09:03:19--  https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/5509b569-5c34-471e-8598-c05c0733bb7f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20231006%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20231006T090319Z&X-Amz-Expires=300&X-Amz-Signature=e11c47ad71d3d29d7e36062c68bc38d50f433ef28c4a90a73460781b5e6e3403&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=9524057&response-content-disposition=attachment%3B%20filename%3Dnode_exporter-1.6.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream [following]
--2023-10-06 09:03:19--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/5509b569-5c34-471e-8598-c05c0733bb7f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20231006%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20231006T090319Z&X-Amz-Expires=300&X-Amz-Signature=e11c47ad71d3d29d7e36062c68bc38d50f433ef28c4a90a73460781b5e6e3403&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=9524057&response-content-disposition=attachment%3B%20filename%3Dnode_exporter-1.6.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10368103 (9.9M) [application/octet-stream]
Saving to: 'node_exporter-1.6.1.linux-amd64.tar.gz'

node_exporter-1.6.1.linux-amd64.tar.gz        100%[===================================================================================>]   9.89M  --.-KB/s    in 0.07s

2023-10-06 09:03:20 (135 MB/s) - 'node_exporter-1.6.1.linux-amd64.tar.gz' saved [10368103/10368103]

ubuntu@ip-172-31-38-156:~$
```

Extract the node exporter from the archive.

```
tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ ls
node_exporter-1.6.1.linux-amd64.tar.gz  prometheus-2.47.1.linux-amd64
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
node_exporter-1.6.1.linux-amd64/
node_exporter-1.6.1.linux-amd64/NOTICE
node_exporter-1.6.1.linux-amd64/node_exporter
node_exporter-1.6.1.linux-amd64/LICENSE
ubuntu@ip-172-31-38-156:~$
```

Move binary to the /usr/local/bin.

```
sudo mv \
  node_exporter-1.6.0.linux-amd64/node_exporter \
  /usr/local/bin/
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo mv \
  node_exporter-1.6.1.linux-amd64/node_exporter \
  /usr/local/bin/
ubuntu@ip-172-31-38-156:~$
```

Clean up, and delete node_exporter archive and a folder.

```
rm -rf node_exporter*
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ ls
node_exporter-1.6.1.linux-amd64  node_exporter-1.6.1.linux-amd64.tar.gz  prometheus-2.47.1.linux-amd64
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ rm -rf node_exporter*
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ ls
prometheus-2.47.1.linux-amd64
ubuntu@ip-172-31-38-156:~$
```

Verify that you can run the binary.

```
node_exporter --version
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ node_exporter --version
node_exporter, version 1.6.1 (branch: HEAD, revision: 4a1b77600c1873a8233f3ffb55afcedbb63b8d84)
  build user:       root@586879db11e5
  build date:       20230717-12:10:52
  go version:       go1.20.6
  platform:         linux/amd64
  tags:             netgo osusergo static_build
ubuntu@ip-172-31-38-156:~$
```

Node Exporter has a lot of plugins that we can enable. If you run Node Exporter help you will get all the options.

```
node_exporter --help
```

--collector.logind We're going to enable the login controller, just for the demo.

Next, create a similar systemd unit file.

```
sudo vim /etc/systemd/system/node_exporter.service
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/systemd/system/node_exporter.service
```

node_exporter.service

```
[Unit]
Description=Node Exporter
Wants=network-online.target
After=network-online.target

StartLimitIntervalSec=500
StartLimitBurst=5

[Service]
User=node_exporter
Group=node_exporter
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/node_exporter \
    --collector.logind

[Install]
WantedBy=multi-user.target
```

```
[Unit]
Description=Node Exporter
Wants=network-online.target
After=network-online.target

StartLimitIntervalSec=500
StartLimitBurst=5

[Service]
User=node_exporter
Group=node_exporter
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/node_exporter \
    --collector.logind

[Install]
WantedBy=multi-user.target
```

Replace Prometheus user and group to node_exporter, and update the ExecStart command.

To automatically start the Node Exporter after reboot, enable the service.

```
sudo systemctl enable node_exporter
```

Then start the Node Exporter.

```
sudo systemctl start node_exporter
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl enable node_exporter
Created symlink /etc/systemd/system/multi-user.target.wants/node_exporter.service → /etc/systemd/system/node_exporter.service.
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl start node_exporter
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
```

Check the status of Node Exporter with the following command:

```
sudo systemctl status node_exporter
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl status node_exporter
● node_exporter.service - Node Exporter
     Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-10-06 09:07:39 UTC; 8s ago
   Main PID: 2030 (node_exporter)
      Tasks: 3 (limit: 1141)
     Memory: 2.0M
        CPU: 9ms
     CGroup: /system.slice/node_exporter.service
             └─2030 /usr/local/bin/node_exporter --collector.logind

Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=thermal_zone
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=time
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=timex
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=udp_queues
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=uname
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=vmstat
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=xfs
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=zfs
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.415Z caller=tls_config.go:274 level=info msg="Listening on" address=[::]:9100
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.415Z caller=tls_config.go:277 level=info msg="TLS is disabled." http2=false address=[::]:9100
ubuntu@ip-172-31-38-156:~$
```

If you have any issues, check logs with journalctl

```
journalctl -u node_exporter -f --no-pager
```

At this point, we have only a single target in our Prometheus. There are many different service discovery mechanisms built into Prometheus. For example, Prometheus can dynamically discover targets in AWS, GCP, and other clouds based on the labels. In the following tutorials, I'll give you a few examples of deploying Prometheus in a cloud-specific environment. For this tutorial, let's keep it simple and keep adding static targets. Also, I have a lesson on how to deploy and manage Prometheus in the Kubernetes cluster.

To create a static target, you need to add job_name with static_configs.

```
sudo vim /etc/prometheus/prometheus.yml
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/prometheus/prometheus.yml
ubuntu@ip-172-31-38-156:~$
```

prometheus.yml

```
  - job_name: node_export
    static_configs:
      - targets: ["localhost:9100"]
```

```
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: node_export
    static_configs:
      - targets: ["localhost:9100"]

~
```

By default, Node Exporter will be exposed on port 9100.

Since we enabled lifecycle management via API calls, we can reload the Prometheus config without restarting the service and causing downtime.

Before, restarting check if the config is valid.

```
1  promtool check config /etc/prometheus/prometheus.yml
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ promtool check config /etc/prometheus/prometheus.yml
Checking /etc/prometheus/prometheus.yml
 SUCCESS: /etc/prometheus/prometheus.yml is valid prometheus config file syntax

ubuntu@ip-172-31-38-156:~$
```

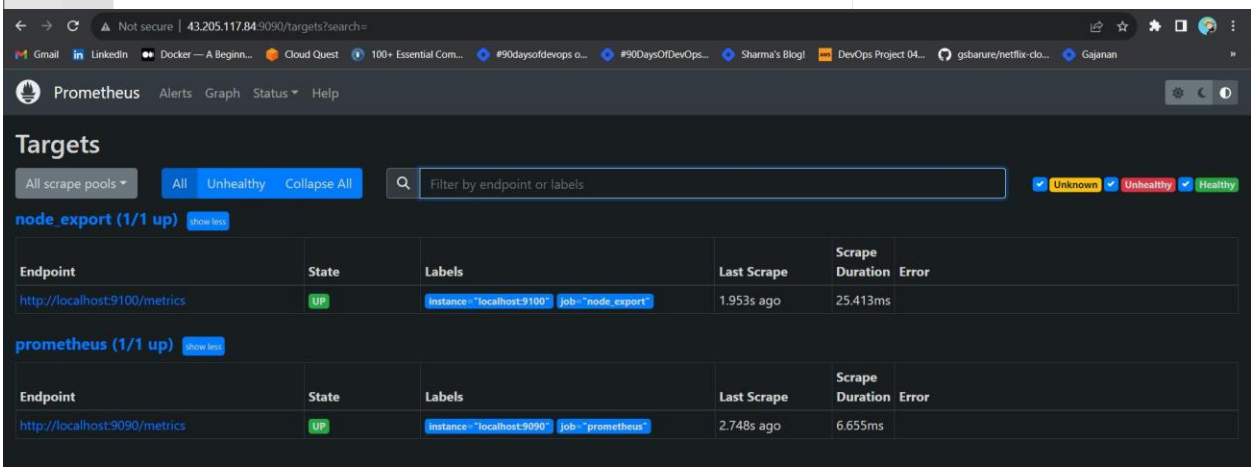
Then, you can use a POST request to reload the config.

```
curl -X POST http://localhost:9090/-/reload
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ curl -X POST http://localhost:9090/-/reload
ubuntu@ip-172-31-38-156:~$
```

Check the targets section

```
http://<ip>:9090/targets
```



Install Grafana on Ubuntu 22.04

To visualize metrics we can use Grafana. There are many different data sources that Grafana supports, one of them is Prometheus.

First, let's make sure that all the dependencies are installed.

```
sudo apt-get install -y apt-transport-https software-properties-common
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo apt-get install -y apt-transport-https software-properties-common
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  python3-software-properties
The following NEW packages will be installed:
  apt-transport-https
The following packages will be upgraded:
  python3-software-properties software-properties-common
2 upgraded, 1 newly installed, 0 to remove and 127 not upgraded.
Need to get 44.4 kB of archives.
After this operation, 169 kB of additional disk space will be used.
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.10 [1510 B]
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 software-properties-common all 0.99.22.7 [14.1 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 python3-software-properties all 0.99.22.7 [28.8 kB]
Fetched 44.4 kB in 0s (2002 kB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 64295 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.4.10_all.deb ...
Unpacking apt-transport-https (2.4.10) ...
Preparing to unpack .../software-properties-common_0.99.22.7_all.deb ...
Unpacking software-properties-common (0.99.22.7) over (0.99.22.6) ...
Preparing to unpack .../python3-software-properties_0.99.22.7_all.deb ...
Unpacking python3-software-properties (0.99.22.7) over (0.99.22.6) ...
Setting up apt-transport-https (2.4.10) ...
Setting up python3-software-properties (0.99.22.7) ...
Setting up software-properties-common (0.99.22.7) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

Next, add the GPG key.

```
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
```

Add this repository for stable releases.

```
echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
deb https://packages.grafana.com/oss/deb stable main
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
```

After you add the repository, update and install Garafana.

```
sudo apt-get update
```

```
sudo apt-get -y install grafana
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo apt-get -y install grafana
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core libfontconfig1 musl
The following NEW packages will be installed:
  fontconfig-config fonts-dejavu-core grafana libfontconfig1 musl
0 upgraded, 5 newly installed, 0 to remove and 127 not upgraded.
Need to get 104 MB of archives.
After this operation, 379 MB of additional disk space will be used.
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 fonts-dejavu-core all 2.37-2build1 [1041 kB]
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 fontconfig-config all 2.13.1-4.2ubuntu5 [29.1 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libfontconfig1 amd64 2.13.1-4.2ubuntu5 [131 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 musl amd64 1.2.2-4 [407 kB]
Get:5 https://packages.grafana.com/oss/deb stable/main amd64 grafana amd64 10.1.4 [102 MB]
17% [5 grafana 0 B/102 MB 0%]
```

To automatically start the Grafana after reboot, enable the service.

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

Then start the Grafana.

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

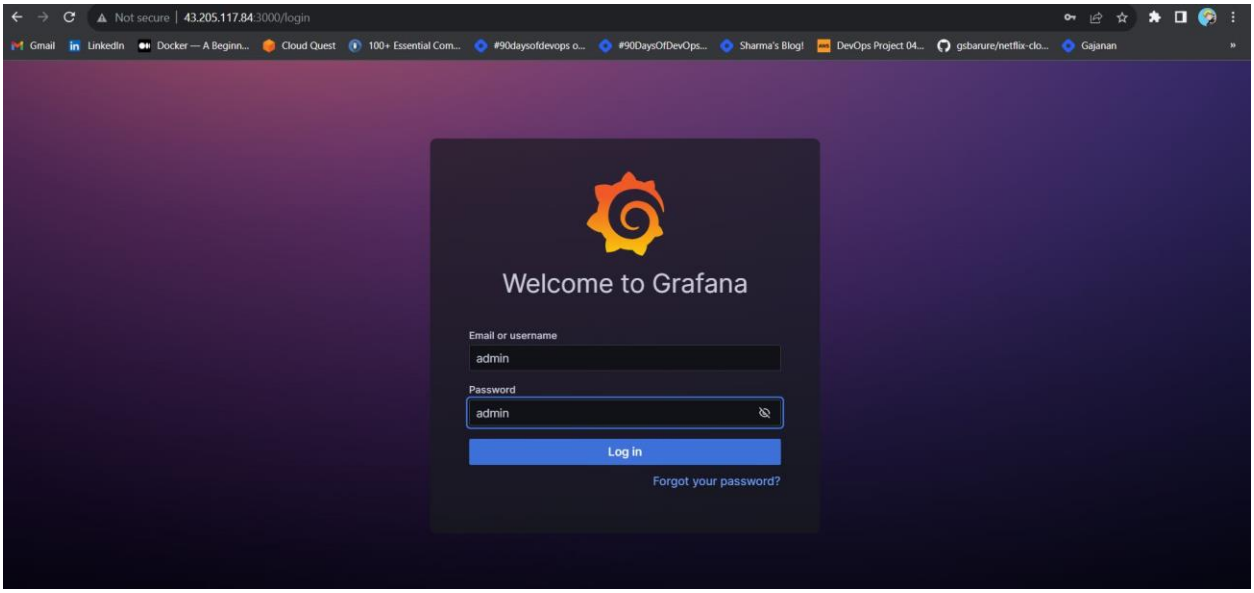
```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl enable grafana-server
Synchronizing state of grafana-server.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable grafana-server
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /lib/systemd/system/grafana-server.service.
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl start grafana-server
ubuntu@ip-172-31-38-156:~$
```

To check the status of Grafana, run the following command:

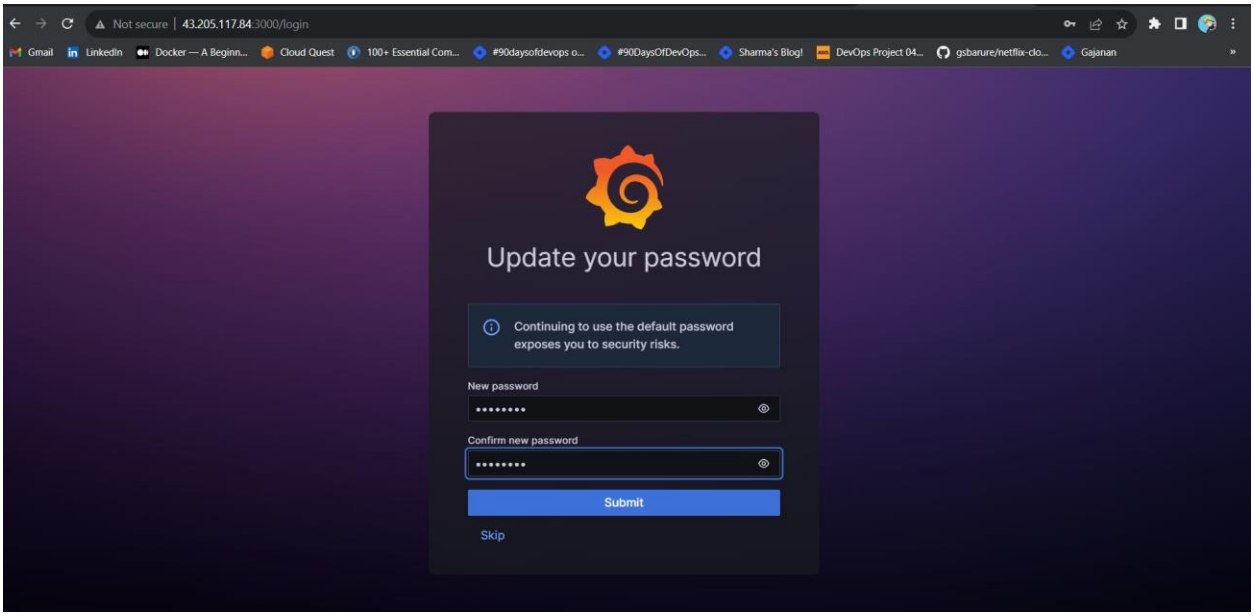
```
sudo systemctl status grafana-server
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo systemctl status grafana-server
● grafana-server.service - Grafana instance
     Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-10-06 09:14:01 UTC; 7s ago
       Docs: http://docs.grafana.org
   Main PID: 3263 (grafana)
      Tasks: 5 (limit: 1141)
     Memory: 175.9M
        CPU: 4.330s
     CGroup: /system.slice/grafana-server.service
             └─3263 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packaging=deb cfg:default.paths.logs=/var/log/g

Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=migrator t=2023-10-06T09:14:06.750063027Z level=info msg="Executing migration" id="Add unique index for folder.title and folder.
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=migrator t=2023-10-06T09:14:06.757199101Z level=info msg="migrations completed" performed=497 skipped=0 duration=4.148747644s
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=sqlstore t=2023-10-06T09:14:06.774714527Z level=info msg="Created default admin" user=admin
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=sqlstore t=2023-10-06T09:14:06.775163061Z level=info msg="Created default organization"
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=secrets t=2023-10-06T09:14:06.782069114Z level=info msg="Envelope encryption state" enabled=true currentprovider=secretKey.v1
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=plugin.loader t=2023-10-06T09:14:06.883193007Z level=warn msg="Plugin missing module.js" pluginID=input warning="Missing module.
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=plugin.loader t=2023-10-06T09:14:06.883523114Z level=info msg="Plugin registered" pluginID=input
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=local.finder t=2023-10-06T09:14:06.883678317Z level=warn msg="Skipping finding plugins as directory does not exist" path=/var/li
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=query_data t=2023-10-06T09:14:06.890278741Z level=info msg="Query Service initialization"
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=live.push_http t=2023-10-06T09:14:06.897367677Z level=info msg="Live Push Gateway initialization"
lines 1-21/21 (END)
```

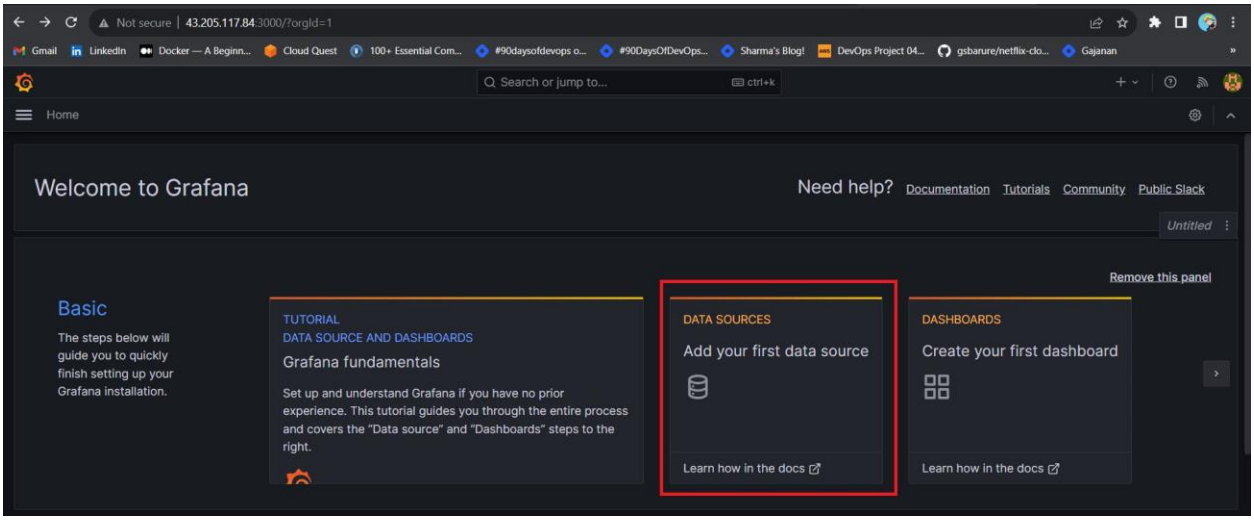
Go to `http://<ip>:3000` and log in to the Grafana using default credentials. The username is admin, and the password is admin as well.



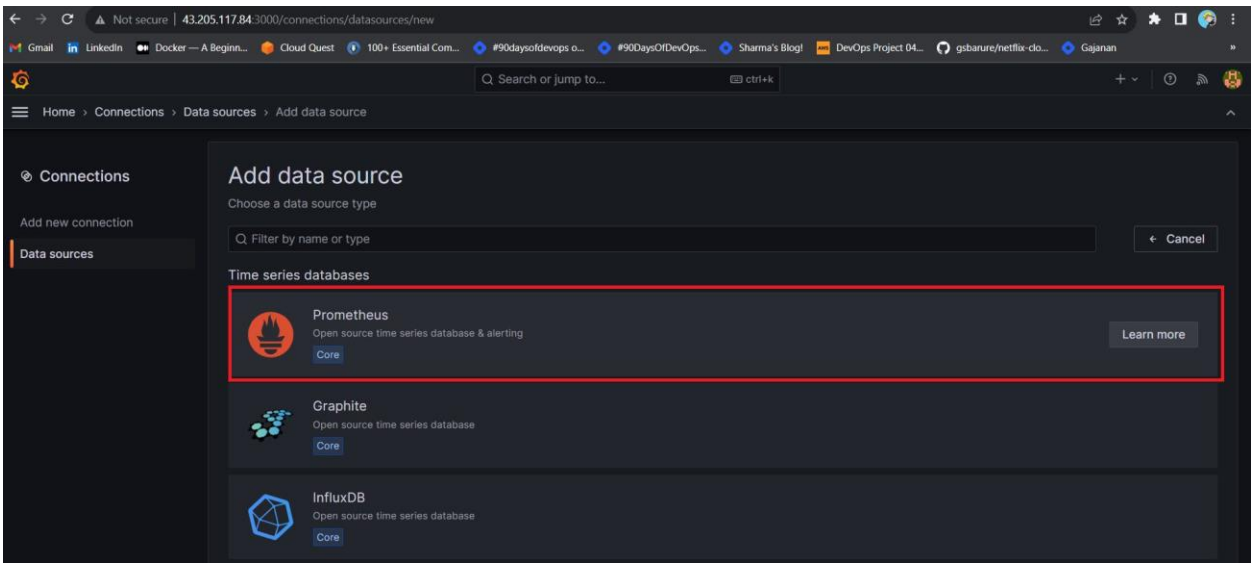
When you log in for the first time, you get the option to change the password.



To visualize metrics, you need to add a data source first.

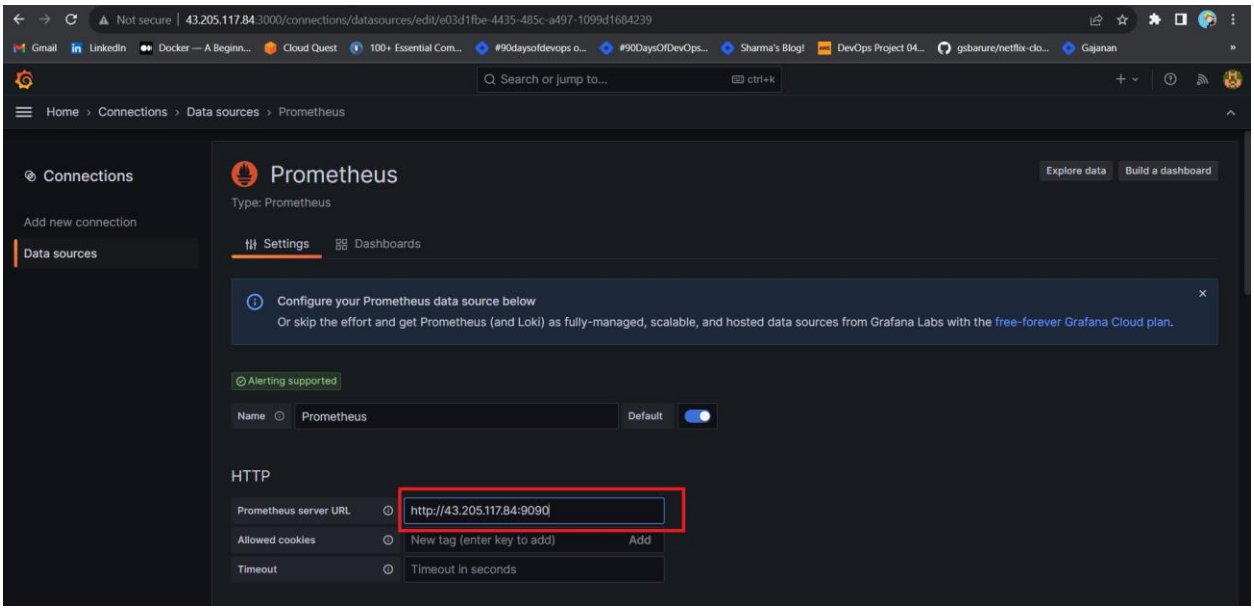


Click Add data source and select Prometheus.

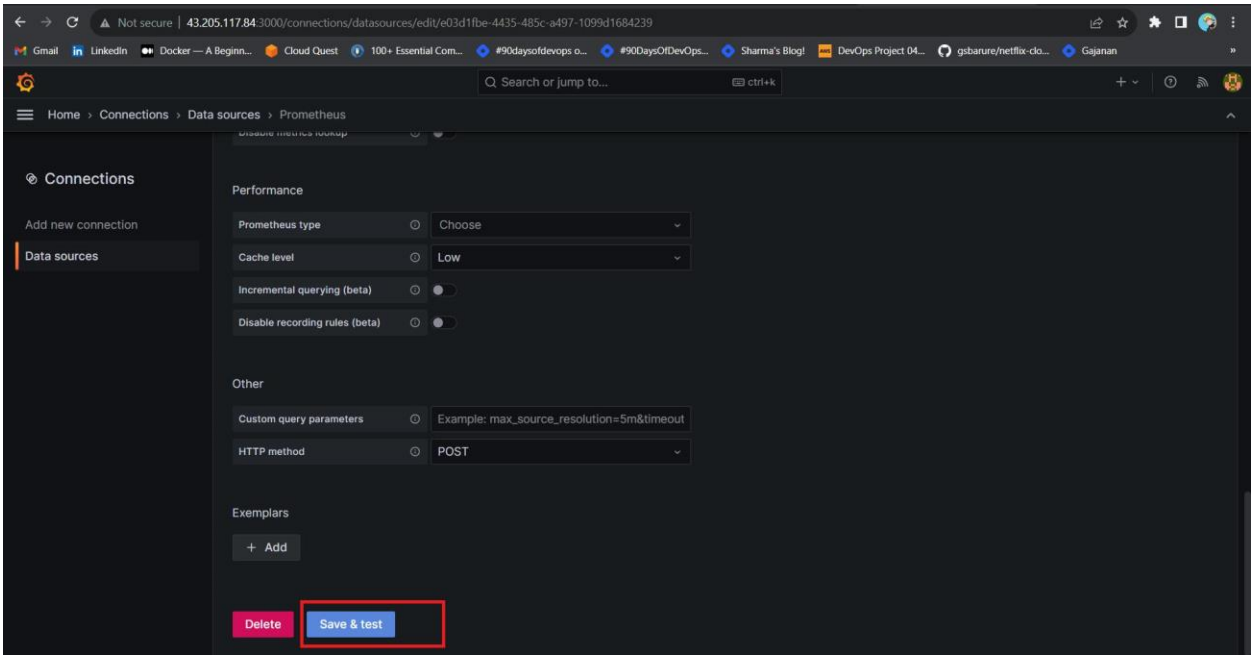


For the URL, enter http://localhost:9090 and click Save and test. You can see Data source is working.

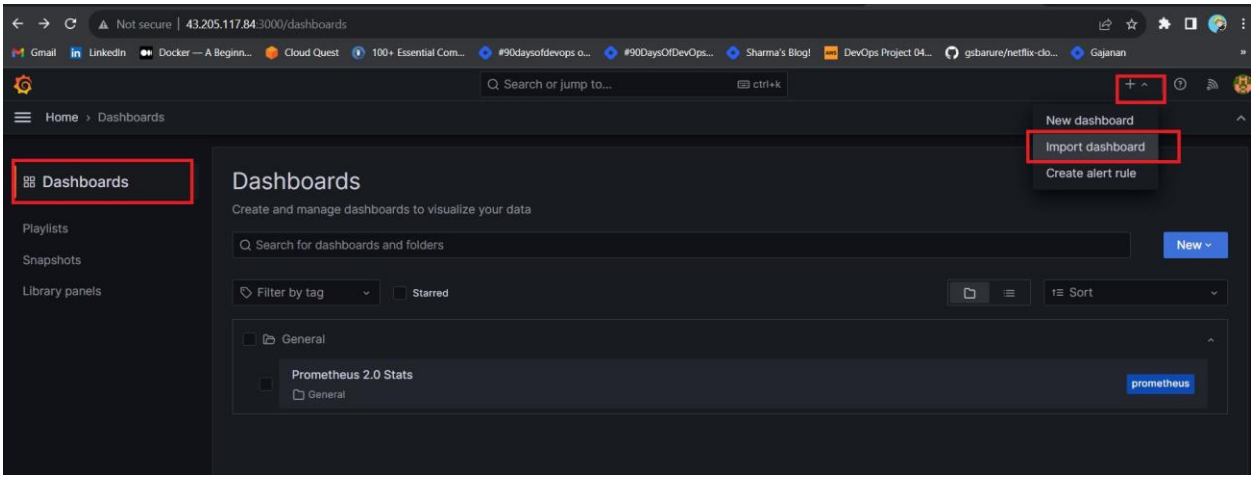
```
<public-ip:9090>
```



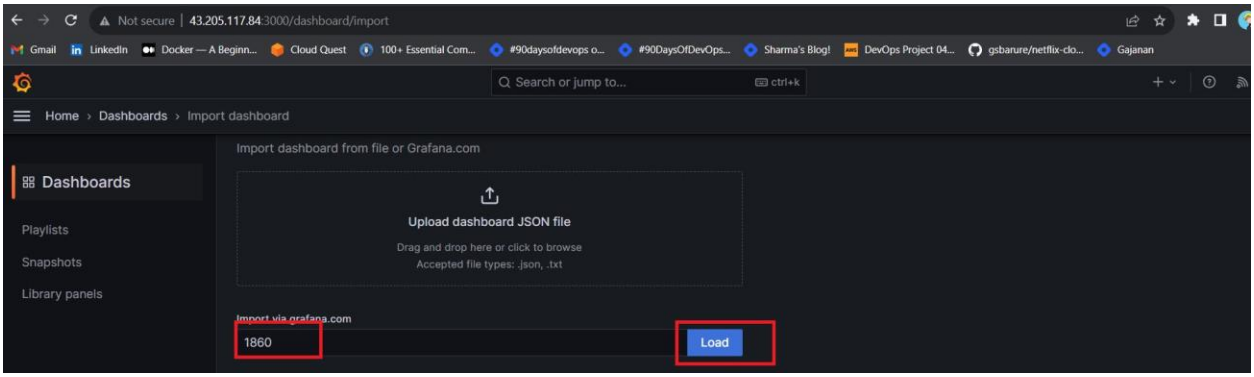
Click on Save and Test.



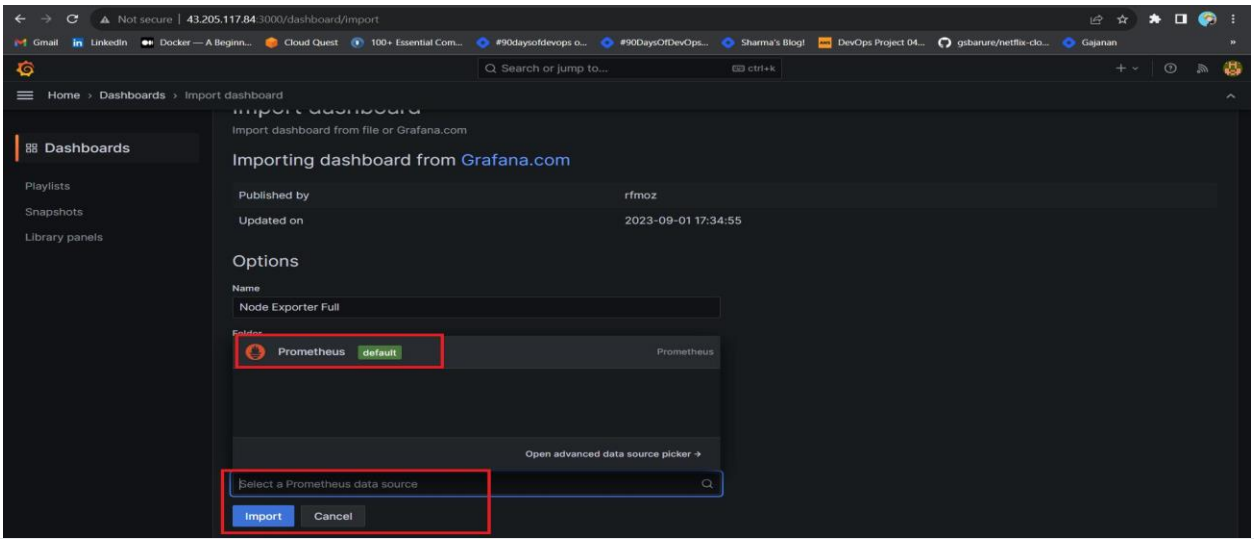
Let's add Dashboard for a better view



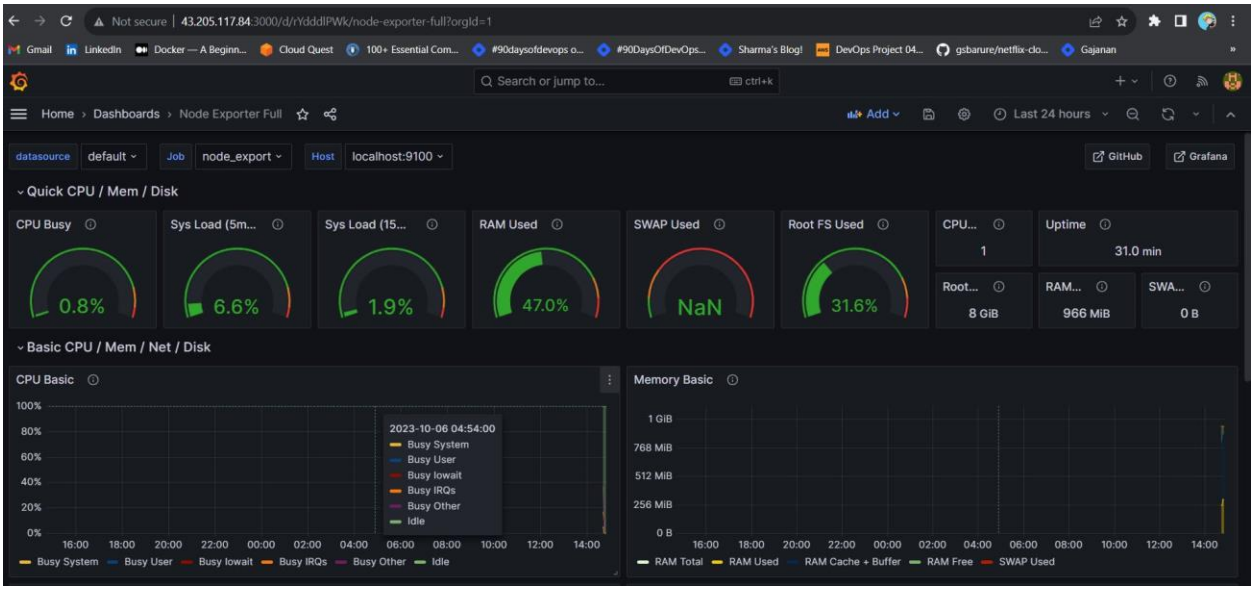
Click on Import Dashboard paste this code 1860 and click on load



Select the Datasource and click on Import



You will see this output

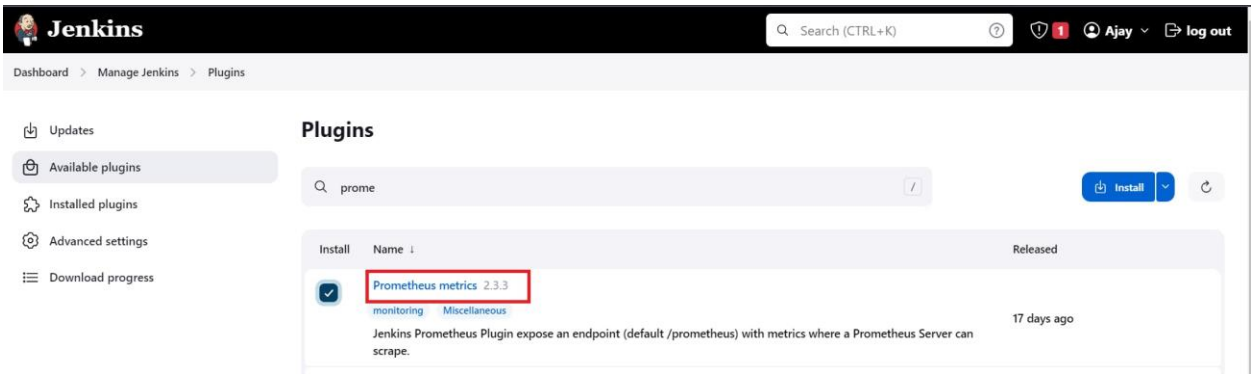


Let's Monitor JENKINS SYSTEM

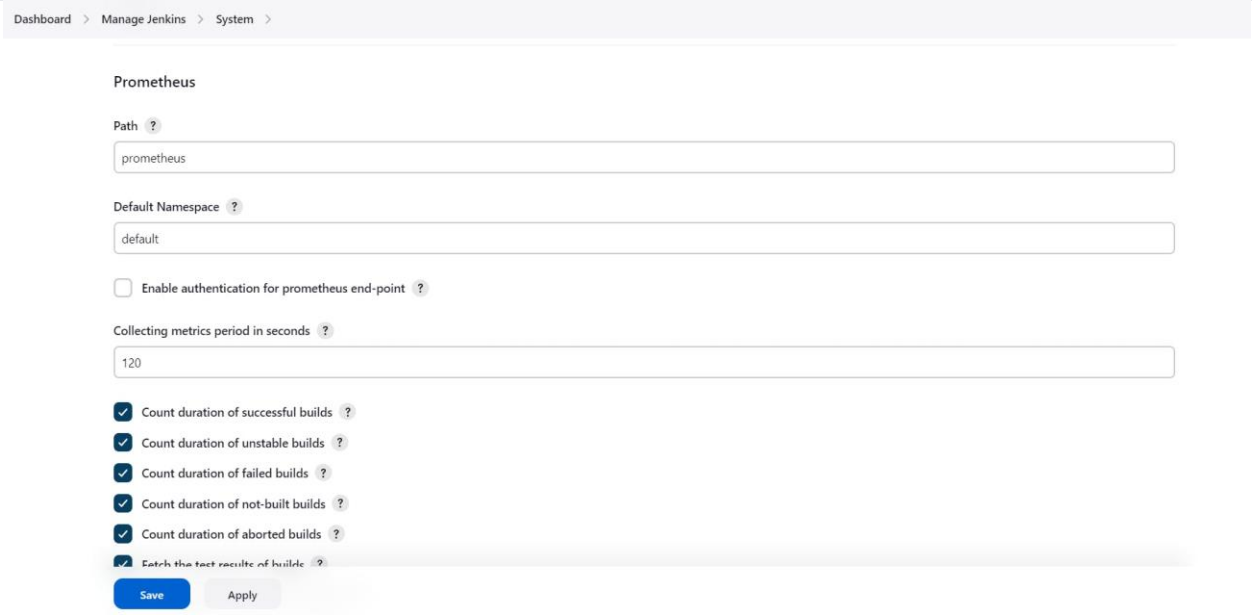
Need Jenkins up and running machine

Goto Manage Jenkins --> Plugins --> Available Plugins

Search for `Prometheus and install it`



Once that is done you will Prometheus is set to `/Prometheus` path in system configurations



Nothing to change click on apply and save

To create a static target, you need to add job_name with static_configs.

```
sudo vim /etc/prometheus/prometheus.yml
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/prometheus/prometheus.yml
```

Paste below code

```
  - job_name: 'jenkins'
    metrics_path: '/prometheus'
    static_configs:
      - targets: ['<jenkins-ip>:8080']
```

```yaml
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: node_export
    static_configs:
      - targets: ["localhost:9100"]

  - job_name: 'jenkins'
    metrics_path: '/prometheus'
    static_configs:
      - targets: ['3.111.170.92:8080']
```

Before, restarting check if the config is valid.

```
promtool check config /etc/prometheus/prometheus.yml
```

Then, you can use a POST request to reload the config.

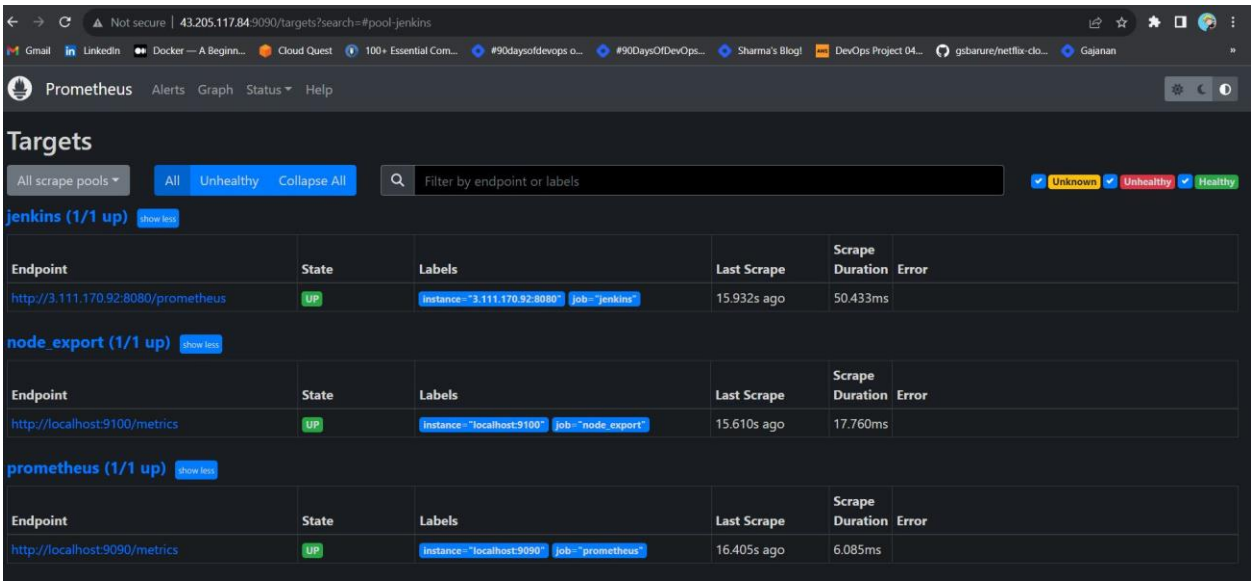
```
curl -X POST http://localhost:9090/-/reload
```

```
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ promtool check config /etc/prometheus/prometheus.yml
Checking /etc/prometheus/prometheus.yml
 SUCCESS: /etc/prometheus/prometheus.yml is valid prometheus config file syntax

ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$ curl -X POST http://localhost:9090/-/reload
ubuntu@ip-172-31-38-156:~$
```

Check the targets section

```
http://<ip>:9090/targets
```

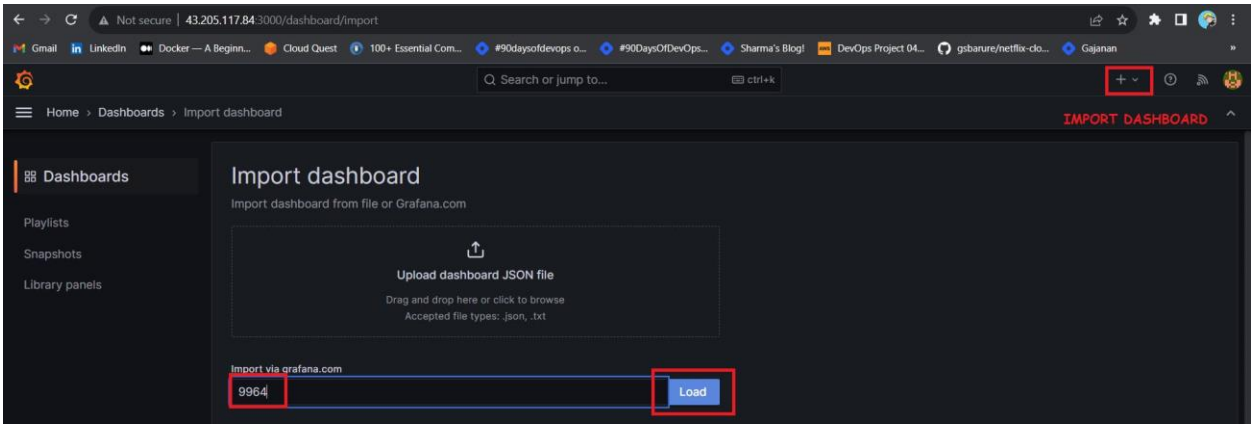
You will see Jenkins is added to it



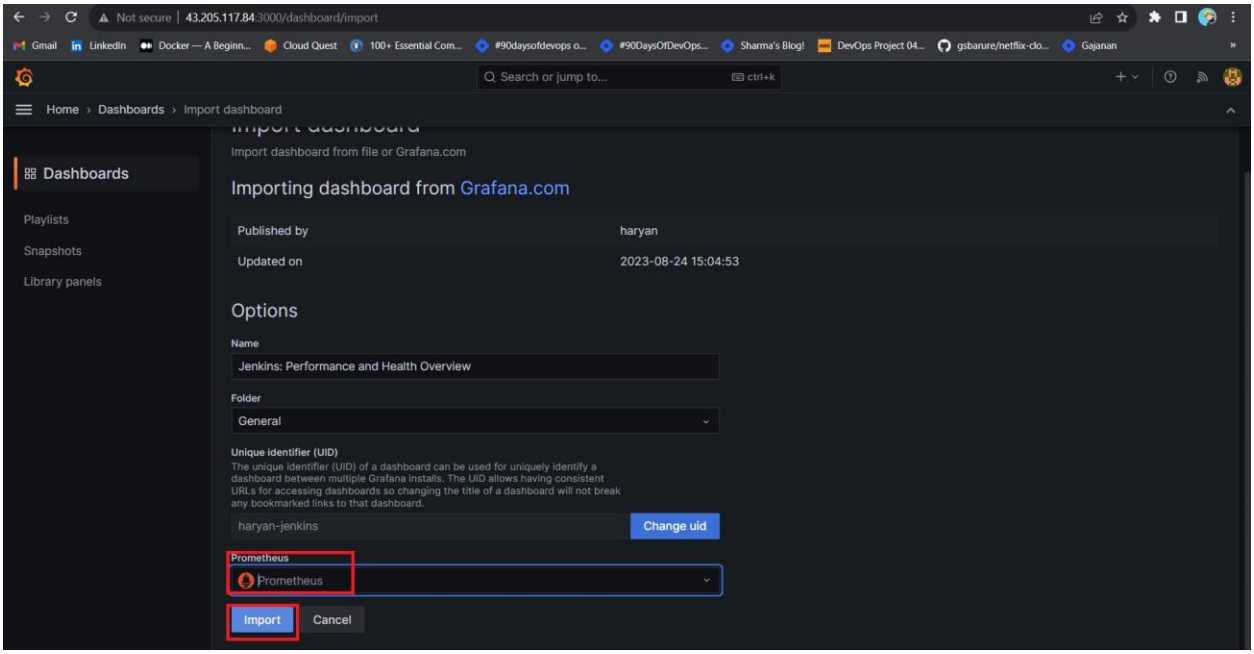
Let's add Dashboard for a better view in Grafana

Click On Dashboard --> + symbol --> Import Dashboard

Use Id `9964` and click on load



Select the data source and click on Import



Now you will see the Detailed overview of Jenkins