**Installing Prometheus Monitoring Server with a Grafana Dashboard**

## Download and Install Node Exporter

As Prometheus is only capable of collecting metrics, we can extend its capabilities by adding **Node Exporter**, a tool that collects information about the system including [CPU, disk, and memory usage](https://github.com/prometheus/node_exporter#enabled-by-default) and exposes them for scraping.

**1. Download the latest version of Node Exporter:**

wget <https://github.com/prometheus/node_exporter/releases/download/v0.18.1/node_exporter-0.18.1.linux-amd64.tar.gz>

**2. Unpack the downloaded archive**

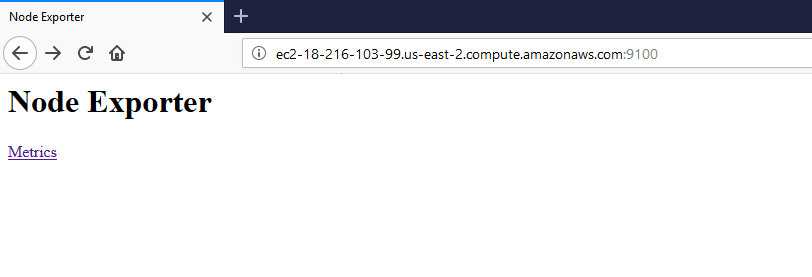
tar zxvf node\_exporter-0.18.1.linux-amd64.tar.gz

**3. Start the Node Exporter Service**

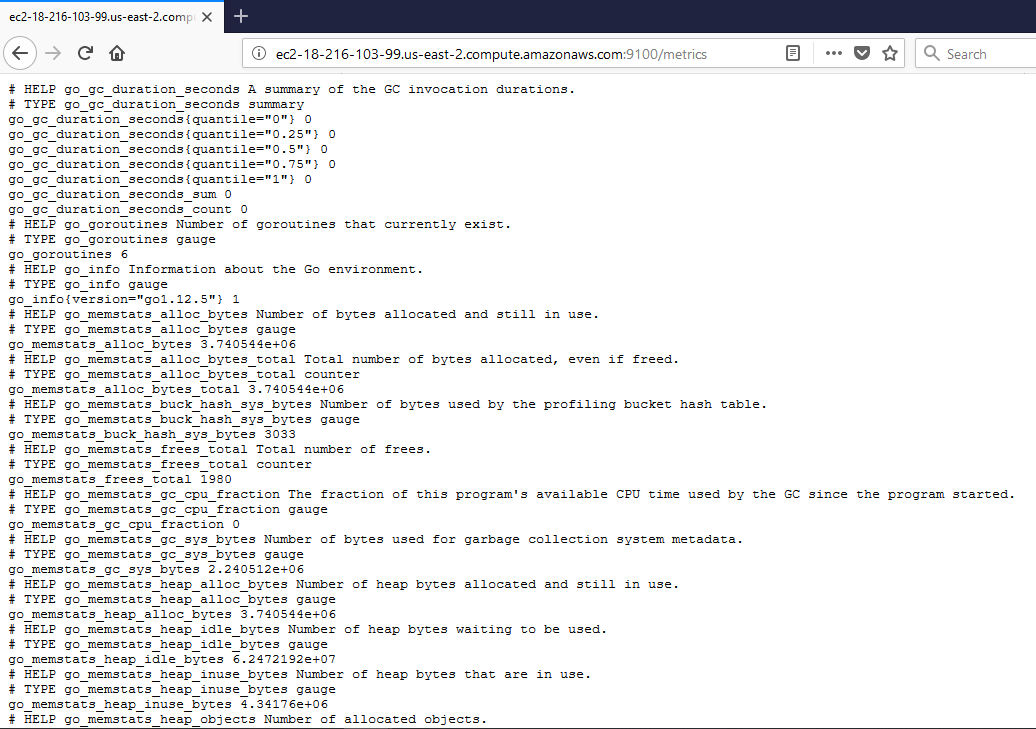
cd node\_exporter-0.18.1.linux-amd64

./node\_exporter

**4. Access the Node Exporter Service**



**5. Click on the Metrics link to read the metrics pertaining to the Node**



## Downloading and Installing Prometheus

**1. Download** [**Prometheus**](https://prometheus.io/download/) **latest release of Prometheus**

wget <https://github.com/prometheus/prometheus/releases/download/v2.13.1/prometheus-2.13.1.linux-amd64.tar.gz>

**2. Unpack the downloaded archive**

tar zxvf prometheus-2.13.1.linux-amd64.tar.gz

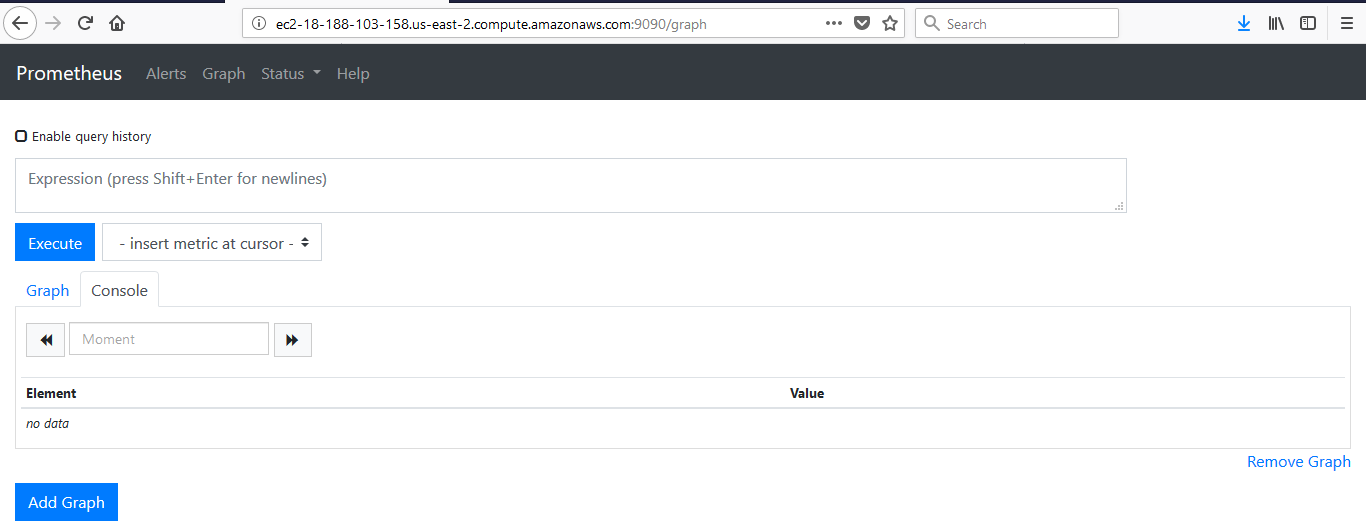
**3. Start the Prometheus Service**

cd prometheus-2.13.1.linux-amd64

./prometheus

**4. Access the Prometheus Service**

http://hostname:9090/



## Configuring Prometheus

Prior to using Prometheus, it needs basic configuring.

**Note:** The configuration file of Prometheus is written in [YAML](http://www.yaml.org/start.html) which strictly forbids to use tabs. If the yaml file is incorrectly formatted, Prometheus will not start.

**1. Stop the Prometheus Process**

netstat -anp | grep :9090 | awk -F / '{ print $1 }' | awk -F " " '{ print $7 }' | xargs kill -9

**2. Open the file prometheus.yml in a text editor:**

vi /etc/prometheus/prometheus.yml

Our file should look like this example:

global:

scrape\_interval: 15s

evaluation\_interval: 15s

rule\_files:

# - "first.rules"

# - "second.rules"

scrape\_configs:

- job\_name: 'prometheus'

scrape\_interval: 5s

static\_configs:

- targets: ['localhost:9090']

Prometheus’ configuration file is divided into three parts: **global, rule\_files, and scrape\_configs**.

In the global part we can find the general configuration of Prometheus:

**scrape\_interval** defines how often Prometheus scrapes targets

**evaluation\_interval** controls how often the software will evaluate rules.

**rules:** Are used to create new time series and for the generation of alerts.

**rule\_files:** This block contains information of the location of any rules we want the Prometheus server to load.

**scrape\_configs:** The last block of the configuration file is named scrape\_configs and contains the information which resources Prometheus monitors.

The global scrape\_interval is set to 15 seconds which is enough for most use cases.

We do not have any rule\_files yet, so the lines are commented out and start with a #.

In the scrape\_configs part we have defined our first exporter. It is Prometheus that monitors itself. As we want to have more precise information about the state of our Prometheus server we reduced the scrape\_interval to 5 seconds for this job. The parameters static\_configs and targets determine where the exporters are running. In our case it is the same server, so we use localhost and the port 9090.

As Prometheus scrapes only exporters that are defined in the scrape\_configs part of the configuration file, we have to add Node Exporter to the file, as we did for Prometheus itself.

We add the following part below the configuration for scrapping Prometheus:

- job\_name: 'node\_exporter\_hostname'

scrape\_interval: 5s

static\_configs:

- targets: ['localhost:9100']

Replace localhost with the IP address of the actual server(s). We can add any number of entries to monitor.

Our Prometheus server is ready to run for the first time.

**Sample prometheus.yaml file with 3 monitoring targets is given below:**

# 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\_exporter\_ec2-18-191-207-68.us-east-2.compute.amazonaws.com'

scrape\_interval: 5s

static\_configs:

- targets: ['ec2-18-191-207-68.us-east-2.compute.amazonaws.com:9100']

- job\_name: 'node\_exporter\_ec2-18-188-103-158.us-east-2.compute.amazonaws.com'

scrape\_interval: 5s

static\_configs:

- targets: ['ec2-18-188-103-158.us-east-2.compute.amazonaws.com:9100']

- job\_name: 'node\_exporter\_ec2-18-216-103-99.us-east-2.compute.amazonaws.com'

scrape\_interval: 5s

static\_configs:

- targets: ['ec2-18-216-103-99.us-east-2.compute.amazonaws.com:9100']

## Running Prometheus

**1. Start the Prometheus Service**

cd prometheus-2.13.1.linux-amd64

./prometheus

You must see the below message if the service startup is successful.

msg="Server is ready to receive web requests."

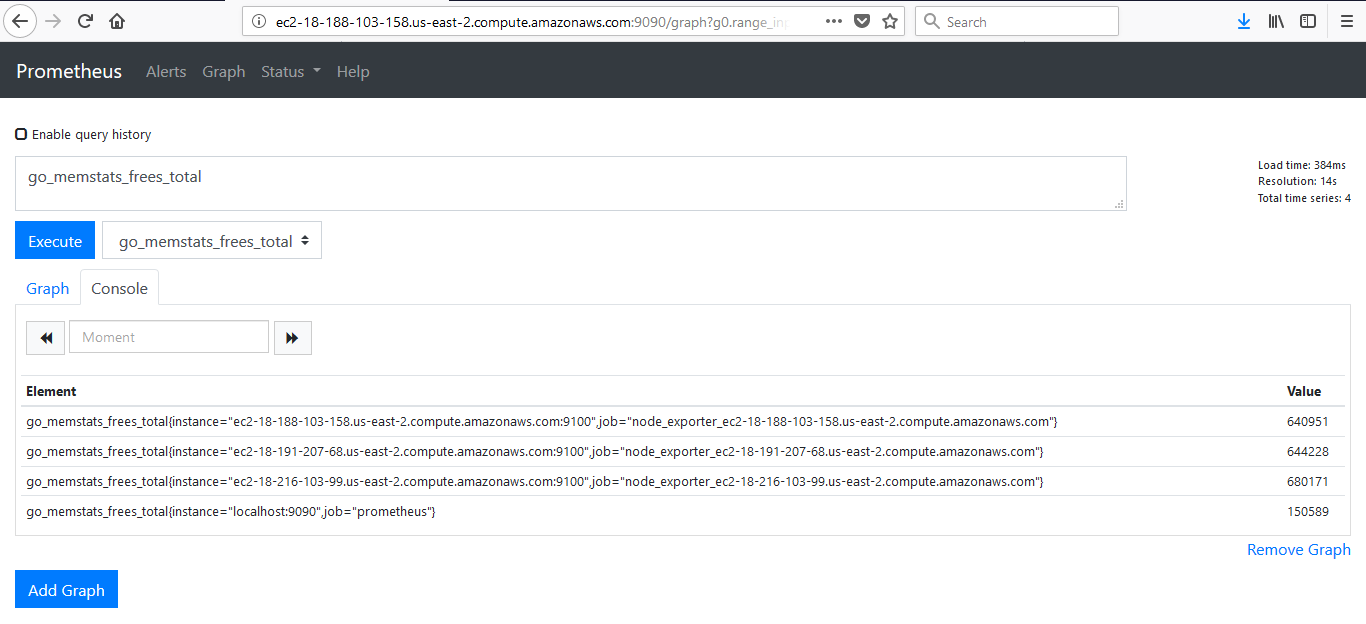
**2. Access the Prometheus Service**

http://hostname:9090/

Your Prometheus server is ready to be used.

We have now installed Prometheus to monitor your instance.

Select the metric of your choice and click on Execute button



**Grafana Installation**

The below steps outline the steps which are needed to install Grafana in Linux.

**1. Download the software from** <https://dl.grafana.com/oss/release/grafana-6.4.2.linux-amd64.tar.gz>

**2. Extract the downloaded software**

tar -zxvf grafana-6.4.2.linux-amd64.tar.gz

**3. Check the Grafana Version**

cd ~/grafana-6.4.2/bin

./grafana-cli --version

Grafana cli version 6.4.2

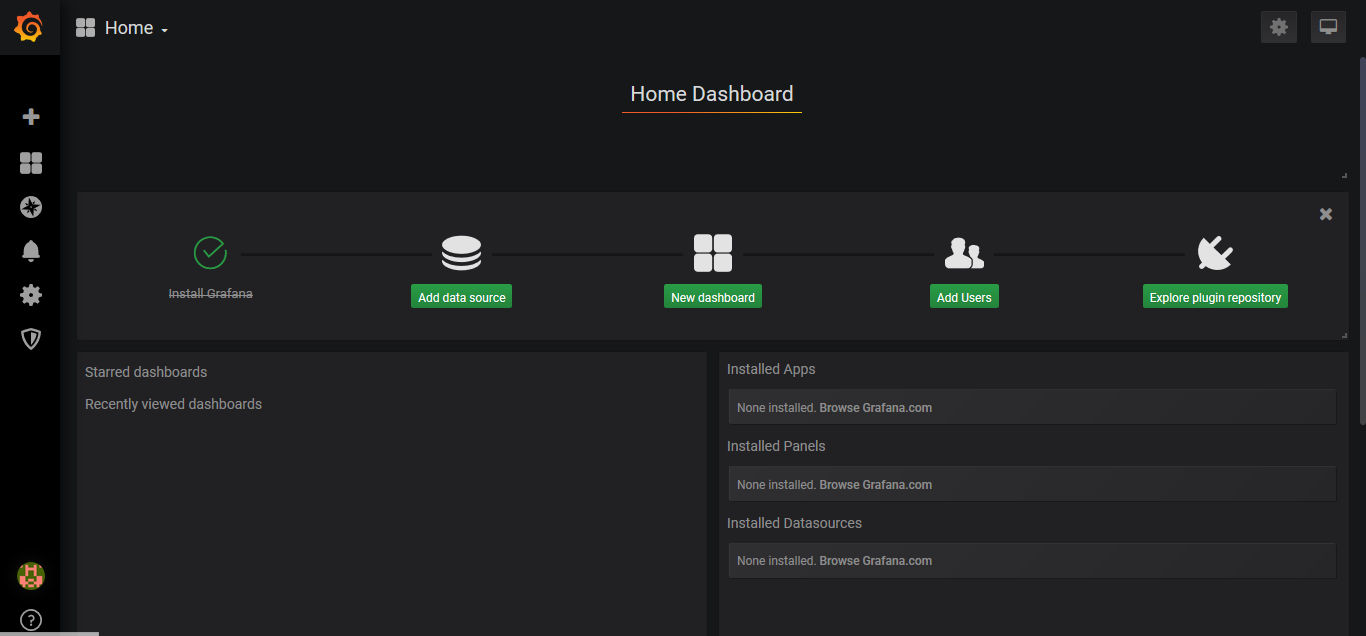
**4. Start the Grafana Server**

./grafana-server

**5. Access the Grafana UI in browser via** <http://hostname:3000/>

Replace the host with the hostname on which Grafana has been installed.

The default credentials to login to UI are admin/admin.

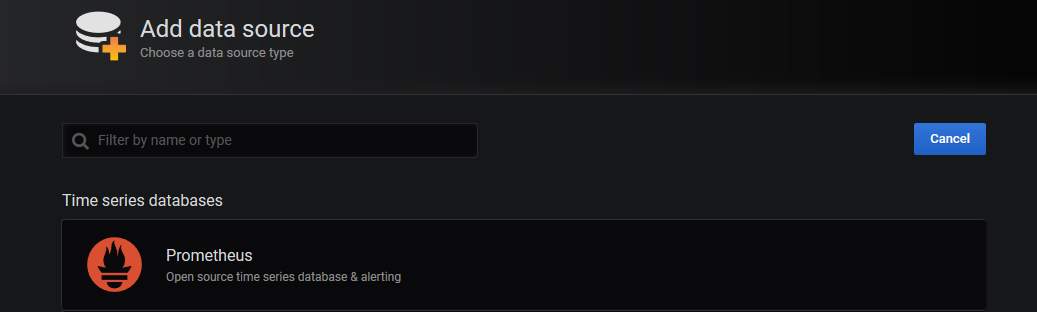


**6. Create data source in Grafana UI**

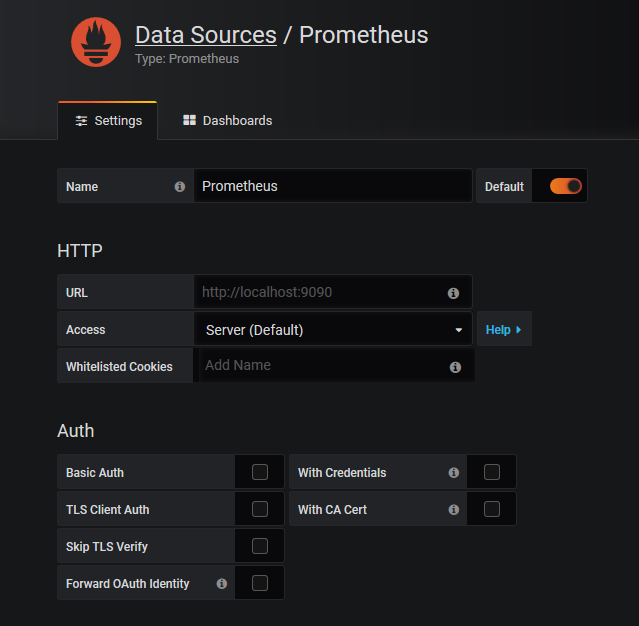
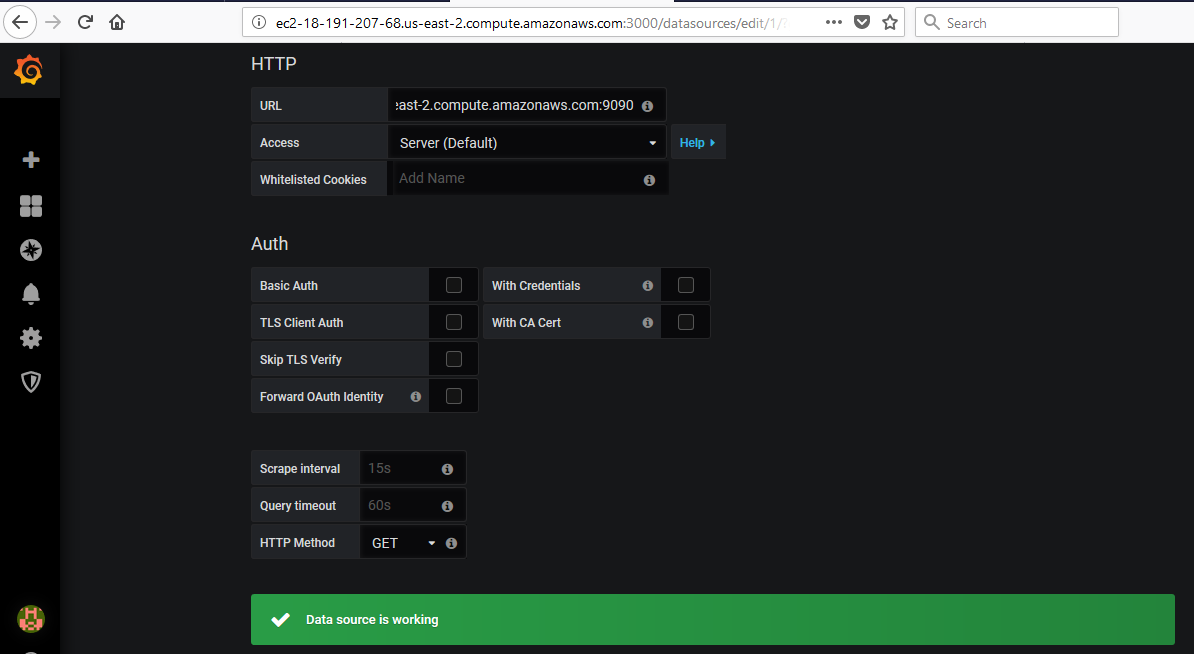
Click on the **“Add data source”** icon in the UI, choose Prometheus and click on

**“select”** button and then click on **“Save and Test”**

I chose **Prometheus.**

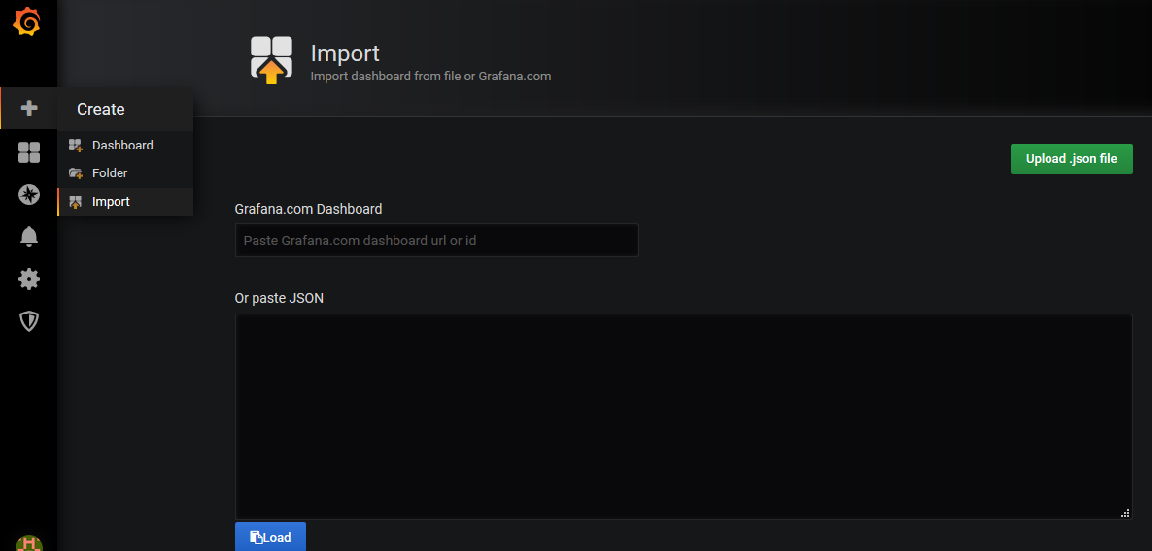


Key in the values in the below screen accordingly and click save and test.

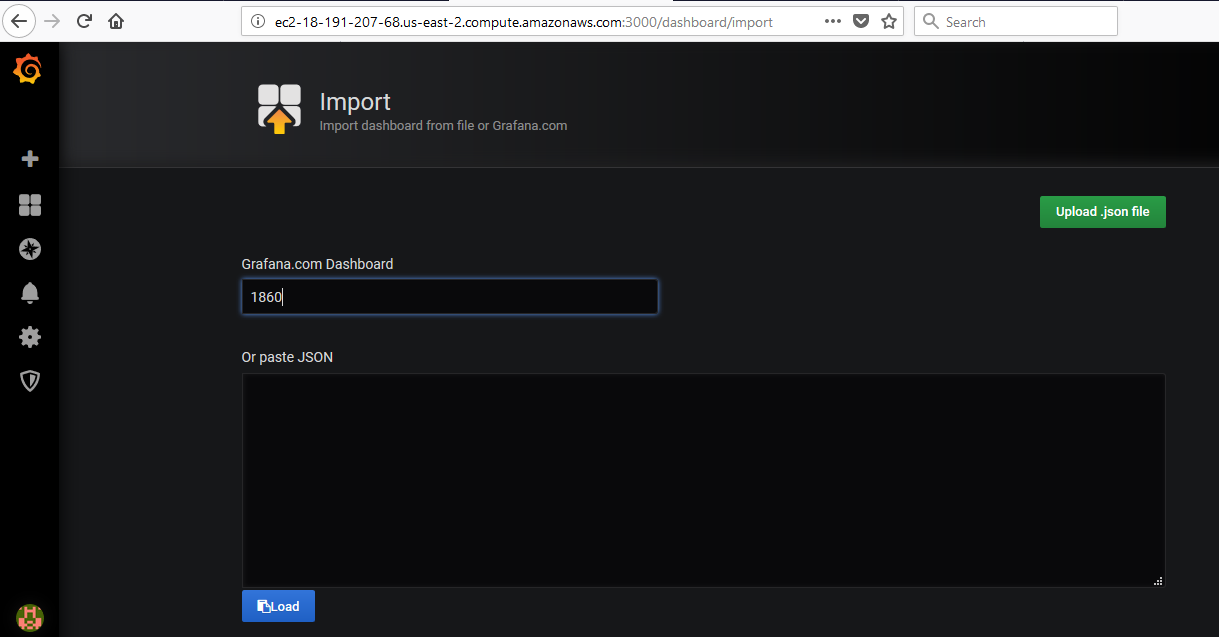
  


**7. Import Grafana Dashboard**

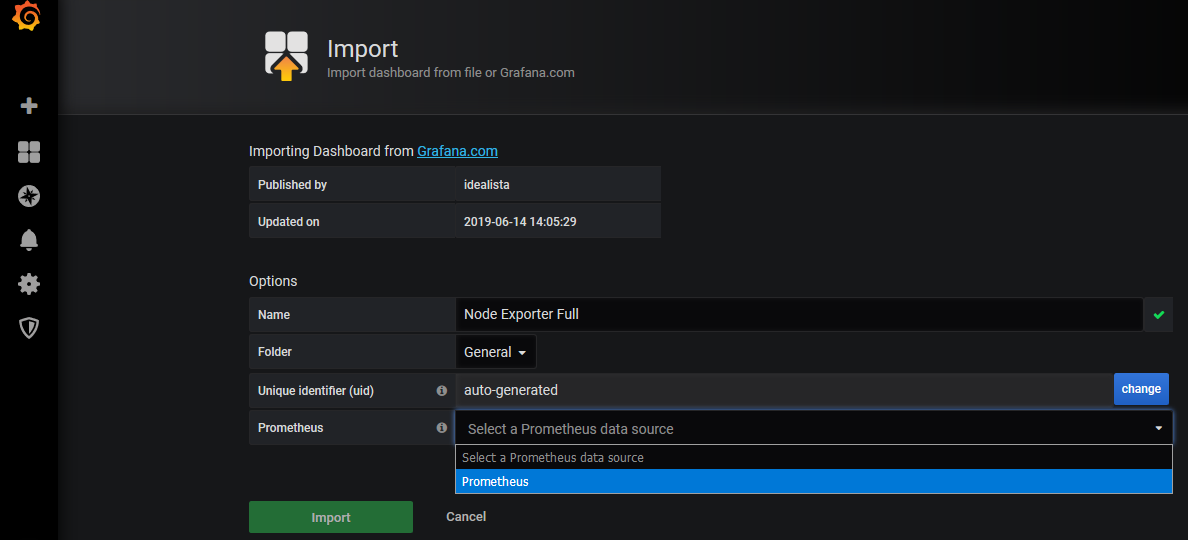
Click on Import button to import a Grafana Dashboard

****

Key in 1860 in the Grafana.com Dashboard Section and click the Load button.

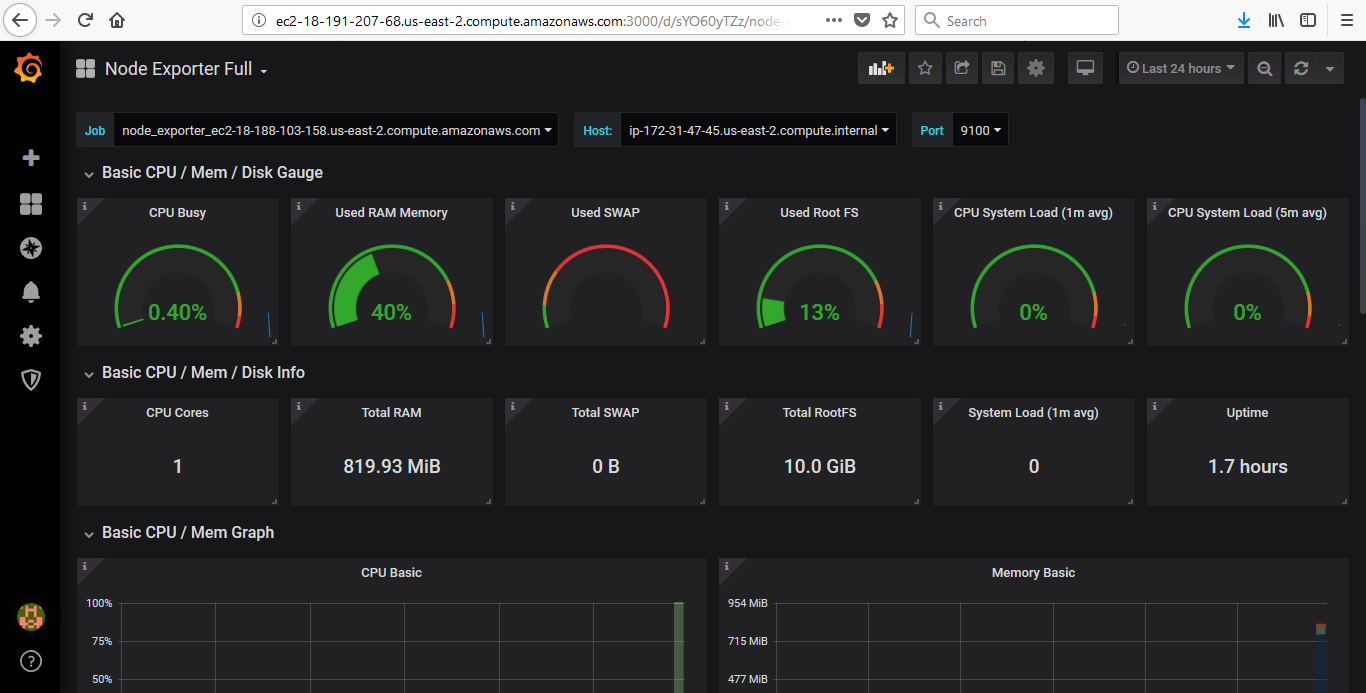


**Select a Prometheus data source from the drop down and click on Import**

****

After clicking import the dashboard would be created with the metrics fetched from the hosts configured under prometheus.yaml under the entry scrape\_configs.

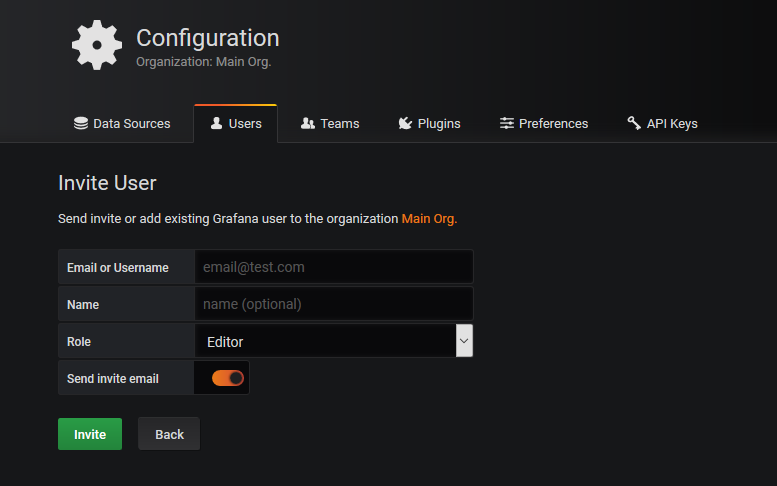
You can choose the job name or host from the below seen drop downs to read the metrics pertaining to the corresponding host(s).



This completes the setup!!!

**8. Add Users in Grafana**

Click on **“Add Users”** icon, fill in the below details and click the button **“Invite”**



**9. Configuration Files**

The Grafana Configuration files reside under the path “/root/grafana-6.4.2/conf”

**10. Grafana Logs**

The Grafana logs can be found under the below path

/root/grafana-6.4.2/data/log/grafana.log

**11. Database**

Grafana needs a database to store users and dashboards (and other things). By default it is configured to use **sqlite3** which is an embedded database. However, we can also use MySQL or Postgres as the Grafana database.

The below is the path to the grafana.db file:

**/root/grafana-6.4.2/data/grafana.db**

If you want to use other databases like MySQL or Postgres, make the changes under the [database] section in the below file.

**/root/grafana-6.4.2/conf/defaults.ini**

**url**

Use either URL or the other fields below to configure the database

**Example:** mysql://user:secret@host:port/database

**type**

Either mysql, postgres or sqlite3, it’s your choice.

