**Prometheus**

**Prometheus** is basically used as a monitoring tool.

It helps us monitor many things like system, software, network etc.

I have been reading about this, even configuring a set of containers to test them, but I have doubts.

I understand that the architecture of Prometheus + Node exporter is based on:

* Node exporters know how to extract metrics. Those are exposed in HTTP, eg. :9201/metrics
* Prometheus queries every X seconds those HTTP endpoints (node-exporter HTTTP) and stores the metrics. It also provides another HTTP for graph/console visualization/querying.

When we download/install this software. We have two files in it prometheus.yaml and rules.

Here the yaml file consists of information about which services are running on which nodes and on which port. (here we just have one file rest we keep on adding informations about nodes , services , exporter, ports)

Other file named rules has entry of conditional statements about where to send which alerts(dir.) , where alerts will be appended(path), many other configurations can be done using condition statements.

Each node should have at least one node exporter besides that there are 1000s of exporters ,it’s upon each node according to different services/software they have there depending exporters.

**Installing prometheus :-**

wget https://github.com/prometheus/prometheus/releases/download/v2.11.1/prometheus-2.11.1.linux-amd64.tar.gz

### Extract The Tar

tar -xvzf prometheus-2.11.1.linux-amd64.tar.gz

### After Extraction Steps

$ mv prometheus-2.11.1.linux-amd64 prometheus

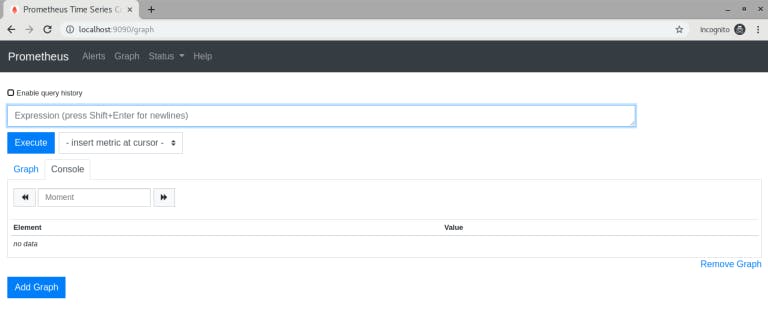
$ cd prometheus/$ ll

* prometheus: It's a binary file which is the core daemon.
* prometheus.yml: This is the config file for Prometheus service.
* promtool: This is another binary file which is used to compile the alert rules file. This will be explained in detail in the next series to this tutorial.

### Execute The Binary File Using The Below Command:

./prometheus

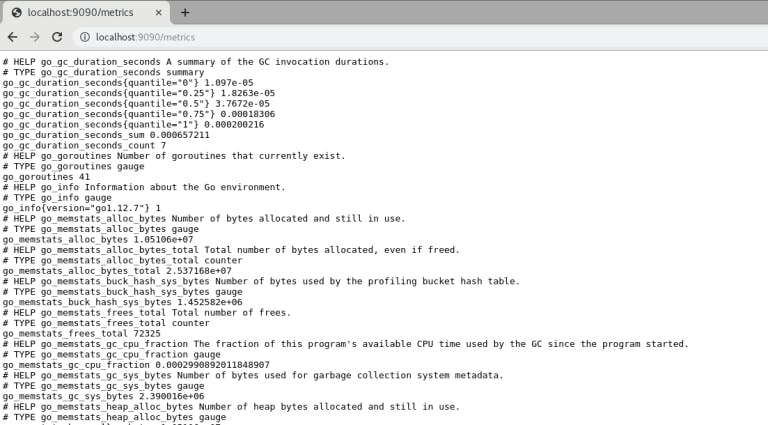
Visit localhost:9090 on your web browser:



Your Prometheus is up and running!

If you notice in prometheus/ folder, It created a folder with the name ‘data’. Prometheus starts storing metrics in this /data folder only.

Now get all metric list by hitting the URL to localhost:9090/metrics



Here is a sample production command:

~/prometheus/prometheus --storage.tsdb.path=/var/lib/prometheus/data/ --web.external-url=http://myurl.com:9090

**--storage.tsdb.path**: Specify the path where you want to save Prometheus data.

**--web.external-url**: You can use this option if you want to bind your address with your URL.

You can get below error in case of your folder don’t have appropriate permission:

level=error ts=2019-08-06T14:25:19.791Z caller=main.go:731 err="opening storage failed: lock DB directory: open /var/lib/lock: permission denied"

You can try appending “sudo” to your command OR you can give appropriate permission to your folder.

### Step 5: Run Prometheus As A Service.

1) Create a file:

/etc/systemd/system/prometheus.service

2) Just paste below code:

[Unit]

Description=Prometheus Server

Documentation=https://prometheus.io/docs/introduction/overview/

After=network-online.target

[Service]

User=root

Restart=on-failure

#Change this line if you download the

#Prometheus on different path user

ExecStart=~/prometheus/prometheus --storage.tsdb.path=/var/lib/prometheus/data/ --web.external-url=http://myurl.com:9090

[Install]

WantedBy=multi-user.target

3) Reload the Systemctl Daemon:

sudo systemctl daemon-reload

4) Start the Prometheus service:

sudo systemctl start prometheus

## Prometheus Exporter Setup

### What Is Exporter?

Exporters can be any scripts or services which will fetch specific metrics from your system and gives data in Prometheus format. There are primarily two ways by which you can fetch metrics and store into Prometheus:

1. Via exporter, In which one service will run on a specific port. So whenever Prometheus service will hit exporter URL with the specific port it will give output in Prometheus format. We will see sample response in the below example during setting up the node exporter.
2. The second approach is you can write a script which will push data in time series to the Prometheus server. Any metric which cannot be scrape by the exporter, It can be pushed using the push method.

So now we are going to setup [node exporter](https://github.com/prometheus/node_exporter). It will fetch your server metrics which will be RAM/DISK/CPU utilization, network, io etc.

### 

### 

### Node Exporter Setup

### Step 1: Download The Binary File And Start Node Exporter:

$ wget https://github.com/prometheus/node\_exporter/releases/download/v0.18.1/node\_exporter-0.18.1.linux-amd64.tar.gz

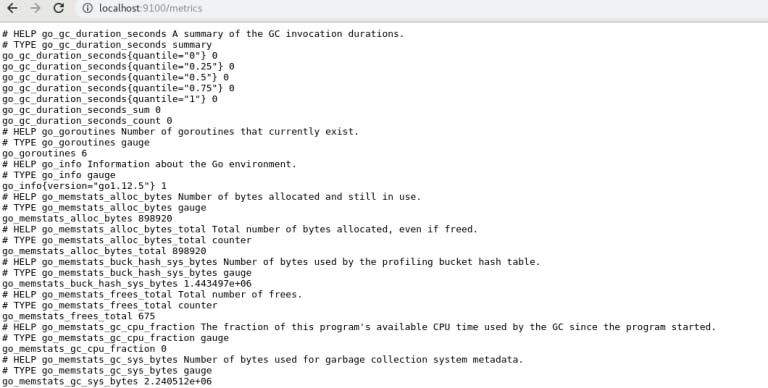
$ tar -xvzf node\_exporter-0.18.1.linux-amd64.tar.gz

$ mv node\_exporter-0.18.1.linux-amd64 node\_exporter

$ cd node\_exporter

$ ./node\_exporter

Just visit to localhost:9100/metrics



### Step 2: Let's Run Node Exporter As Service:

Create a file in below path:

/etc/systemd/system/node-exporter.service

Just paste below code:

[Unit]

Description=Node exporter

After=network-online.target

[Service]

User=root

Restart=on-failure

#Change this line if you download the

#Prometheus on different path user

ExecStart=~/node\_exporter/node\_exporter

[Install]

WantedBy=multi-user.target

Reload the systemctl daemon:

sudo systemctl daemon-reload

Start the Prometheus service:

sudo systemctl start node-exporter

### Step3: You Are Set With Node Exporter. Now In Prometheus, We Need To Configure This Node Exporter Reference So That Prometheus Can Collect Metrics From This Exporter.

Open file ~/prometheus/prometheus.yml add below configuration:

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'] # Add below block for node\_exporter - job\_name: node\_exporter scrape\_interval: 1m scrape\_timeout: 1m metrics\_path: "/metrics" static\_configs: - targets: ['localhost:9100']

Save and exit.

* **job\_name:** You can give any name to your scrape job.
* **scrape\_interval**: Interval in which Prometheus will scrape the metrics from the specified URL.
* **scrape\_timeout:** If your exporter has taken more than 1m to scrape the metrics it will be a timeout.
* **metric\_path**: This is what your endpoint’s path should be (i.e localhost:9100/metrics)
* **targets**: Here you can specify the number of servers on which node exporter is running with the same configuration.

### Step 4: Here's The Command To Execute Prometheus:

~/prometheus/prometheus --storage.tsdb.path=/var/lib/prometheus/data/ --config.file=~/prometheus/prometheus.yml --web.external-url=http://myurl.com:9090

Also, don’t forget to make the same changes in your service file:/etc/systemd/system/prometheus.service

### Step 5: Run This Code

sudo systemctl restart prometheus

### Step 6: Visiting Localhost:9090 Again

Now visit the URL localhost:9090. In the Expression field you can search for “node\_filesystem\_size\_bytes” by clicking on the “Execute” button.

**There are already lots of exporters available on the internet like Nginx exporter, MongoDB exporter, MySQL server exporter, etc. Just download them and start using it.**

**Alert Manager**

The Alertmanager **handles alerts sent by client applications such as the Prometheus server**. It takes care of deduplicating, grouping, and routing them to the correct receiver integration such as email, PagerDuty, or OpsGenie. It also takes care of silencing and inhibition of alerts.

It shows alerts in graphical interface , which is produced by prometheus and configured in its two files {yaml and roles}.

**First, we need to download the latest binary of AlertManager**

sudo su

cd /opt/wget <https://github.com/prometheus/alertmanager/releases/download/v0.11.0/alertmanager-0.11.0.linux-amd64.tar.gz>

tar -xvzf alertmanager-0.11.0.linux-amd64.tar.gz

mv alertmanager-0.11.0.linux-amd64/alertmanager /usr/local/bin/

**AlertManager Configuration**

The AlertManager uses a configuration file named alertmanager.ymlThis file is contained in the extracted directory. However, it is not of our use. That’s why we need to create our own alertmanager.yml

mkdir /etc/alertmanager/

sudo vi /etc/alertmanager/alertmanager.yml

[Prometheus email alerting | Prometheus alertmanager email configuration](https://youtu.be/x2qTvTN8YKI)

Then put the following :

global:

smtp\_from:

smtp\_smarthost:

smtp\_auth\_username:

smtp\_auth\_password:

templates:

- '/etc/alertmanager/template/\*.tmpl'

route:

group\_by: ['alertname']

group\_wait: 3s

group\_interval: 5s

repeat\_interval: 1h

receiver: mail-slack-receiver

receivers:

- name: 'mail-slack-receiver'

slack\_configs:

- api\_url: put your url here

channel: 'put your channel name here'

send\_resolved: true

icon\_url: https://avatars3.githubusercontent.com/u/3380462

text: >-

{{ range .Alerts -}}

\*Alert:\* {{ .Annotations.title }}{{ if .Labels.severity }} - `{{ .Labels.severity }}`{{ end }}

\*Description:\* {{ .Annotations.description }}

\*Details:\*

{{ range .Labels.SortedPairs }} • \*{{ .Name }}:\* `{{ .Value }}`

{{ end }}

{{ end }}

email\_configs:

- to: 'emails of the ones that need to be notified'

send\_resolved: true

Finally, we create the AlertManager systemd service :

vi /etc/systemd/system/alertmanager.service

Put the following :

[Unit]

Description=AlertManager Server Service

Wants=network-online.target

After=network-online.target

[Service]

User=root

Group=root

Type=simple

ExecStart=/usr/local/bin/alertmanager --config.file /etc/alertmanager/alertmanager.yml -web.external-url=http://x.x.x.x:9093

[Install]

WantedBy=multi-user.target

*Using -web.external-url=http://x.x.x.x:9093 allow the notification URL to be redirected to the prometheus AlertManager web interface.* x.x.x.x correspond*s* to the prometheus server public ip.

Then reload the daemon and start the alertmanager service :

systemctl daemon-reload

systemctl start alertmanager

systemctl enable alertmanager

systemctl status alertmanager`

**Grafana**

It's a tool which displays the output/information in a graphical format on the dashboard.

It has a user-friendly interface.

You can also install the Prometheus Alertmanager Plugin in Grafana. Head to the instance where grafana is installed and install the plugin:

| grafana-cli plugins install camptocamp-prometheus-alertmanager-datasource |
| --- |

Once the plugin is installed, restart Grafana:

service grafana-server restart

Access to the URL : *x.x.x.x:3030* and configure an AlertManager Prometheus datasource.