**Prometheus :**

Before we begin talking about how to install [**Prometheus**](https://prometheus.io/) on Ubuntu 20.04, let’s briefly understand - **What Prometheus is**?

Prometheus is a powerful, and open-source monitoring system. It collects metrics from the services and stores them in a time-series database. Further, it offers a multi-dimensional data model, as well as flexible query language. It enables diverse visualization possibilities via tools like Grafana.

By default, Prometheus only exports metrics about itself, like the number of requests it receives, its memory consumption, etc. But, users can expand Prometheus by installing exporters, optional programs generating additional metrics.

These, exporters - both the official ones, as well as the community-contributed ones provide information from infrastructure, databases, and web servers to messaging systems, APIs, etc.

In this tutorial, you will install Prometheus on Ubuntu 20.04. We will also address a few FAQs related to Prometheus installation.

**Prerequisites**

* Have one Ubuntu 20.04 server, including a sudo non-root user and a firewall too.

**Step 1 - Creating the Service Users**

1) For security purposes, you will create two new user accounts, prome and node\_exporter. It will let you isolate the ownership of Prometheus’ core files and directories. Now, create two users. Use the --no-create-home and --shell /bin/false options, so these users cannot log in to the server:

sudo useradd --no-create-home --shell /bin/false prometheus

sudo useradd --no-create-home --shell /bin/false node\_exporter

2) Before downloading Prometheus binaries, create necessary directories for storing Prometheus files and data as well. Follow standard Linux conventions, and create a directory in /etc. It will enable Prometheus configuration files and a directory /var/lib for its data:

sudo mkdir /etc/prometheus

sudo mkdir /var/lib/prometheus

3) Next, set the user and group ownership on new directories to the Prometheus-user, by:

sudo chown prometheus:prometheus /etc/prometheus

sudo chown prometheus:prometheus /var/lib/prometheus

4) With users and directories in place, now you will download Prometheus. Then, create a minimal configuration file, to run Prometheus for the first time.

**Step 2 - Download Prometheus**

1) You will now download and unpack the current version of Prometheus into the home directory. Find the latest binaries with their checksums on the Prometheus download page:

cd ~

curl -LO https://github.com/prometheus/prometheus/releases/download/v2.0.0/prometheus-2.0.0.linux-amd64.tar.gz

2) Next, use sha256sum command. It will generate a checksum of the downloaded file, by:

sha256sum prometheus-2.0.0.linux-amd64.tar.gz

3) After that, compare the output from this command with the checksum on the Prometheus download page. You will need to ensure, that your file is genuine and not corrupt:

Output

e12917b25b32980daee0e9cf879d9ec197e2893924bd1574604eb0f550034d46 prometheus-2.0.0.linux-amd64.tar.gz

4) Also, if the checksums do not match, you will remove the downloaded file. Then, repeat the earlier steps to re-download the file. Now, unpack the downloaded archive using the below command:

tar xvf prometheus-2.0.0.linux-amd64.tar.gz

5) This will create a directory named, prometheus-2.0.0.linux-amd64. It contains two binary files (prometheus and promtool), and consoles. Also,  console\_libraries directories containing the web-interface files, license, a notice, and a few example files. Next, copy these two binaries to the /usr/local/bin directory:

sudo cp prometheus-2.0.0.linux-amd64/prometheus /usr/local/bin/

sudo cp prometheus-2.0.0.linux-amd64/promtool /usr/local/bin/

6) You will now set the user and group ownership on binaries to the **Prometheus** user:

sudo chown prometheus:prometheus /usr/local/bin/prometheus

sudo chown prometheus:prometheus /usr/local/bin/promtool

7) Then, copy the consoles and console\_libraries directories to the /etc/prometheus:

sudo cp -r prometheus-2.0.0.linux-amd64/consoles /etc/prometheus

sudo cp -r prometheus-2.0.0.linux-amd64/console\_libraries /etc/prometheus

8) Again, set the user and group ownership on directories to the Prometheus user. Use the -R flag. It will ensure the ownership is set on files inside the directory too:

sudo chown -R prometheus:prometheus /etc/prometheus/consoles

sudo chown -R prometheus:prometheus /etc/prometheus/console\_libraries

9) At last, remove the leftover files from the home directory. These files are no longer necessary:

rm -rf prometheus-2.0.0.linux-amd64.tar.gz prometheus-2.0.0.linux-amd64

10) Next, after Prometheus installation, create configuration and service files in preparation for the first run.

**Step 3 - Configuring the Prometheus**

1) In the /etc/prometheus directory, use nano or your favorite text editor. It will create a configuration file as prometheus.yml. This file will contain enough information to run Prometheus the first time:

sudo vi /etc/prometheus/prometheus.yml

**Note:** The Prometheus configuration file uses the YAML format. It strictly forbids tabs and requires 2 spaces for indentation otherwise Prometheus will fail to start and you need to make sure, the configuration file is not incorrectly formatted.

2) Now, in global-settings, define default intervals for the scraping metrics. Also, the Prometheus will apply these settings to every exporter. So, unless an individual exports own settings override the globals:

global:

scrape\_interval: 15s

The scrape\_interval value indicates Prometheus collects metrics from its exporters every 15-seconds. It is long enough for most of the exporters.

3) Next, add Prometheus itself to the list of exporters. It will scrape from the following scrape\_configs directive:

...

scrape\_configs:

- job\_name: 'prometheus'

scrape\_interval: 5s

static\_configs:

- targets: ['localhost:9090']

Prometheus uses job\_name to label the exporters in queries and on graphs, be sure to pick something explanatory here. Further, Prometheus exports necessary data about itself. It will let you can use it for monitoring performance and debugging. In addition, Prometheus uses both static\_configs and targets directives to know where exporters are running. Although, this particular exporter is running on the same server as Prometheus itself. So, you will use localhost instead of IP-address along with default-port, 9090.

The configuration file will now look like below:

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'prometheus'

scrape\_interval: 5s

static\_configs:

- targets: ['localhost:9090']

4) You will need to save the file and exit the text editor.

5) Set the user and group ownership on the configuration file to Prometheus user:

sudo chown prometheus:prometheus /etc/prometheus/prometheus.yml

6) After, the configuration is complete, you are ready to test Prometheus by running it for the first time.

**Step 4 - Running the Prometheus**

1) Now, start the Prometheus as a Prometheus user and provide the path to both configuration file as well as the data directory:

sudo -u prometheus /usr/local/bin/prometheus \

--config.file /etc/prometheus/prometheus.yml \

--storage.tsdb.path /var/lib/prometheus/ \

--web.console.templates=/etc/prometheus/consoles \

--web.console.libraries=/etc/prometheus/console\_libraries

The output has the information about Prometheus loading progress, configuration file, and other relating services confirms that Prometheus is listening on the port 9090:

Output

level=info ts=2017-11-17T18:37:27.474530094Z caller=main.go:215 msg="Starting Prometheus" version="(version=2.0.0, branch=HEAD, re

vision=0a74f98628a0463dddc90528220c94de5032d1a0)"

level=info ts=2017-11-17T18:37:27.474758404Z caller=main.go:216 build\_context="(go=go1.9.2, user=root@615b82cb36b6, date=20171108-

07:11:59)"

level=info ts=2017-11-17T18:37:27.474883982Z caller=main.go:217 host\_details="(Linux 4.4.0-98-generic #121-Ubuntu SMP Tue Oct 10 1

4:24:03 UTC 2017 x86\_64 prometheus-update (none))"

level=info ts=2017-11-17T18:37:27.483661837Z caller=web.go:380 component=web msg="Start listening for connections" address=0.0.0.0

:9090

level=info ts=2017-11-17T18:37:27.489730138Z caller=main.go:314 msg="Starting TSDB"

level=info ts=2017-11-17T18:37:27.516050288Z caller=targetmanager.go:71 component="target manager" msg="Starting target manager...

"

level=info ts=2017-11-17T18:37:27.537629169Z caller=main.go:326 msg="TSDB started"

level=info ts=2017-11-17T18:37:27.537896721Z caller=main.go:394 msg="Loading configuration file" filename=/etc/prometheus/promethe

us.yml

level=info ts=2017-11-17T18:37:27.53890004Z caller=main.go:371 msg="Server is ready to receive requests.

2) So, if you get an error message, you will need to double-check the used YAML syntax in your configuration file then, follow the on-screen instructions to solve the problem.

3) Now, halt the Prometheus by pressing CTRL+C and open a new systemd service file:

sudo nano /etc/systemd/system/prometheus.service

4) The service file indicates systemd to run Prometheus as Prometheus user. The location of the configuration file is in the /etc/prometheus/prometheus.yml directory. It stores its data in the /var/lib/prometheus directory. Now, copy the following content into the file:

Prometheus service file - /etc/systemd/system/prometheus.service

[Unit]

Description=Prometheus

Wants=network-online.target

After=network-online.target

[Service]

User=prometheus

Group=prometheus

Type=simple

ExecStart=/usr/local/bin/prometheus \

--config.file /etc/prometheus/prometheus.yml \

--storage.tsdb.path /var/lib/prometheus/ \

--web.console.templates=/etc/prometheus/consoles \

--web.console.libraries=/etc/prometheus/console\_libraries

[Install]

WantedBy=multi-user.target

5) Next, save the file and close the text editor. To use the new service, reload systemd using the below command:

sudo systemctl daemon-reload

6) You will now start Prometheus using the below command:

sudo systemctl start prometheus

7) You will make sure Prometheus is running. Check service’s status, by:

sudo systemctl status prometheus

The output will tell you the Prometheus’ status. Also, the main process identifier (PID), memory use, and much more.

8) So, if the service’s status is not active then, follow the on-screen instructions. Further, re-trace the previous steps to solve the problem:

Output

● prometheus.service - Prometheus

Loaded: loaded (/etc/systemd/system/prometheus.service; disabled; vendor preset: enabled)

Active: active (running) since Fri 2017-07-21 11:40:40 UTC; 3s ago

Main PID: 2104 (prometheus)

Tasks: 7

Memory: 13.8M

CPU: 470ms

CGroup: /system.slice/prometheus.service

...

9) When ready to move on, then press Q to quit the status command. Lastly, enable service to start on the boot, by:

sudo systemctl enable prometheus

Now, the Prometheus is running. You can install an additional exporter, it will generate metrics about the server’s resources.

**Step 5 - Downloading the Node Exporter**

1) You will need to expand Prometheus beyond metrics about itself only. Install an additional exporter known as Node-Exporter. It provides details about the system, the CPU, disk, and memory usage. Now, download the current version of node-exporter into your home directory. You will find the latest binaries along with their checksums on the Prometheus download page:

cd ~

curl -LO https://github.com/prometheus/node\_exporter/releases/download/v0.15.1/node\_exporter-0.15.1.linux-amd64.tar.gz

2) Next, use the sha256sum command, It will generate a checksum of the downloaded file:

sha256sum node\_exporter-0.15.1.linux-amd64.tar.gz

3) Verify the downloaded file’s integrity. Do it by comparing its checksum with the one on download page:

Output

7ffb3773abb71dd2b2119c5f6a7a0dbca0cff34b24b2ced9e01d9897df61a127 node\_exporter-0.15.1.linux-amd64.tar.gz

4) If the checksums do not match, you will need to remove the downloaded file and repeat the steps. Next, unpack the downloaded archive using the below command:

tar xvf node\_exporter-0.15.1.linux-amd64.tar.gz

The above will create a directory known as node\_exporter-0.15.1.linux-amd64. It contains a binary file named, node\_exporter, a license, and a notice.

5) Then, copy the binary to /usr/local/bin directory. You will set the user and group ownership to node\_exporter user:

sudo cp node\_exporter-0.15.1.linux-amd64/node\_exporter /usr/local/bin

sudo chown node\_exporter:node\_exporter /usr/local/bin/node\_exporter

6) At last, remove the leftover files from your home directory as they are no longer needed:

rm -rf node\_exporter-0.15.1.linux-amd64.tar.gz node\_exporter-0.15.1.linux-amd64

7) After, installing Node Exporter, test it by running it. Do it before creating a service file for it, so that it starts on-boot.

**Step 6 - Running the Node Exporter**

1) The steps for running Node-Exporter are similar to running Prometheus. Now, start by creating systemd service file for the Node-Exporter:

sudo nano /etc/systemd/system/node\_exporter.service

2) This service enables the system to run Node Exporter. Now, copy the below content into the service file:

Node Exporter service file - /etc/systemd/system/node\_exporter.service

[Unit]

Description=Node Exporter

Wants=network-online.target

After=network-online.target

[Service]

User=node\_exporter

Group=node\_exporter

Type=simple

ExecStart=/usr/local/bin/node\_exporter

[Install]

WantedBy=multi-user.target

3) The Collectors define which metrics the Node-Exporter will generate. You will then see Node-Exporter’s complete list of collectors. It also includes those enabled by default and also which are deprecated, in the Node-Exporter README file. If you need to override the default list of the collectors, use the--collectors.enabled flag:

Node Exporter service file part - /etc/systemd/system/node\_exporter.service

...

ExecStart=/usr/local/bin/node\_exporter --collectors.enabled meminfo,loadavg,filesystem

...

4) The above example tells Node-Exporter to generate metrics. It is basically using meminfo, loadavg, and filesystem collectors. It enables you to limit the collectors to a few or many. But, also note that there are no blank spaces before or after commas.

Next, save the file and close the text editor.

5) Then, reload systemd to use this new service:

sudo systemctl daemon-reload

6) You will now run Node-Exporter using the below command:

sudo systemctl start node\_exporter

7) Then, verify that Node Exporter is running correctly with status command:

sudo systemctl status node\_exporter

8) Again, the output tells you the Node-Exporter’s status. Along with the main process identifier (PID), memory usage, and much more, if the service’s status is not active then, follow the on-screen messages. Further, re-trace the previous steps to resolve the problem before continuing:

Output

● node\_exporter.service - Node Exporter

Loaded: loaded (/etc/systemd/system/node\_exporter.service; disabled; vendor preset: enabled)

Active: active (running) since Fri 2017-07-21 11:44:46 UTC; 5s ago

Main PID: 2161 (node\_exporter)

Tasks: 3

Memory: 1.4M

CPU: 11ms

CGroup: /system.slice/node\_exporter.service

9) At last, enable Node-Exporter to start on boot using the below command:

sudo systemctl enable node\_exporter

The Node Exporter configuration is now complete and running. You will then tell Prometheus to start scraping the new metrics.

**Step 7 - Configuring the Prometheus to Scrape Node-Exporter**

1) The Prometheus only scrapes exporters, defined in the scrape\_configs portion of its configuration file. You will need to add an entry for Node-Exporter. So, open the configuration file using the below command:

sudo nano /etc/prometheus/prometheus.yml

2) Then, at the end of the scrape\_configs block, add a new entry as node\_exporter:

...

- job\_name: 'node\_exporter'

scrape\_interval: 5s

static\_configs:

- targets: ['localhost:9100']

3) Here, the exporter is also running on the same server as Prometheus. You can use localhost instead of an IP address. Again, along with the Node-Exporter's default port, 9100. The whole configuration file will look like below:

Prometheus config file - /etc/prometheus/prometheus.yml

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'prometheus'

scrape\_interval: 5s

static\_configs:

- targets: ['localhost:9090']

- job\_name: ‘microservice-1’

scrape\_interval: 5s

static\_configs:

- targets: ['localhost:9100']

4) Next, save the file and exit the text editor, when ready to continue. Finally, you can restart Prometheus for the changes to take action:

sudo systemctl restart prometheus

5) Again, verify that everything is running correctly. Do it with the status command:

sudo systemctl status prometheus

6) If the service status is not set to active, follow the on-screen instructions then, re-trace previous steps before moving on:

Output

● prometheus.service - Prometheus

Loaded: loaded (/etc/systemd/system/prometheus.service; disabled; vendor preset: enabled)

Active: active (running) since Fri 2017-07-21 11:46:39 UTC; 6s ago

Main PID: 2219 (prometheus)

Tasks: 6

Memory: 19.9M

CPU: 433ms

CGroup: /system.slice/prometheus.service

7) Now, the installation and configuration of Prometheus and Node-Exporter is complete and running. Also, for a final precaution before connecting to the web interface, enhance the installation’s security with basic HTTP authentication. It will ensure that unauthorized users cannot access your metrics.

**Grafana :**

Grafana is a monitoring and visualization software. With the help of a dashboard and charts, you can visualize, query and monitor your data over a period of time. You can connect with complex data sources like MySQL, ElasticSearch, Graphite, Prometheus, etc.

In today's post, we will guide you through the steps of installing Grafana on the operating system. [Ubuntu](https://codepre.com/en/tips/ubuntu/). You can install Grafana on Ubuntu OS in different ways:

Here, we will cover:

* Installation of Grafana using the APT repository

**Installation of Grafana using the APT repository**In the following procedure, we will install Grafana using the APT repository. Although you can install Grafana using the official Ubuntu repository, but it is not the latest version. To get the latest version of Grafana, we can use the official Grafana APT repository. With this method, it will be easier for you to update Grafana using apt-get update whenever a newer version is available. Let us begin.

1. Add the software-properties-common package to your system. It will allow you to easily manage the PPAs in your system:

$ sudo apt install software-properties-common

The installation will now start and if during the installation you will be prompted **y/n**write **y** and press **Enter**.

2. Now, you will need to import the Grafana GPG key into the APT source key set. Run the following command to do so:

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

This command will download the Grafana GPG key and add it to the APT source key set.

3. Next, use the following command to add the Grafana repository to your system source list:

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

4. After adding the Grafana repository, update the system font list using the following command:

$ sudo apt update

5. Now you can install Grafana as follows:

$ sudo apt install Grafana

The installation will now start and if during the installation you will be prompted **y/n**write **y** and press **Enter**.

Text

Description automatically generated

6. To verify the installation, use the following command:

$ grafana-server -v

The following output verifies that the version of Grafana **8.0.5** has been installed on your system.

Text

Description automatically generated

After installation, you can start the Grafana service using the following command:

$ sudo systemctl start grafana-server

Use the following command to make the service start automatically on boot:

$ sudo systemctl enable grafana-server

To check if the Grafana service is running, use the following command:

$ sudo systemctl status grafana-server

You should see the following output showing that the Grafana service is running without errors.

Text

Description automatically generated