How to Install Prometheus and Grafana on Ubuntu?

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[¶](https://antonputra.com/monitoring/install-prometheus-and-grafana-on-ubuntu/" \l "install-prometheus-on-ubuntu-2004)

First of all, let create a dedicated Linux user or 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 \

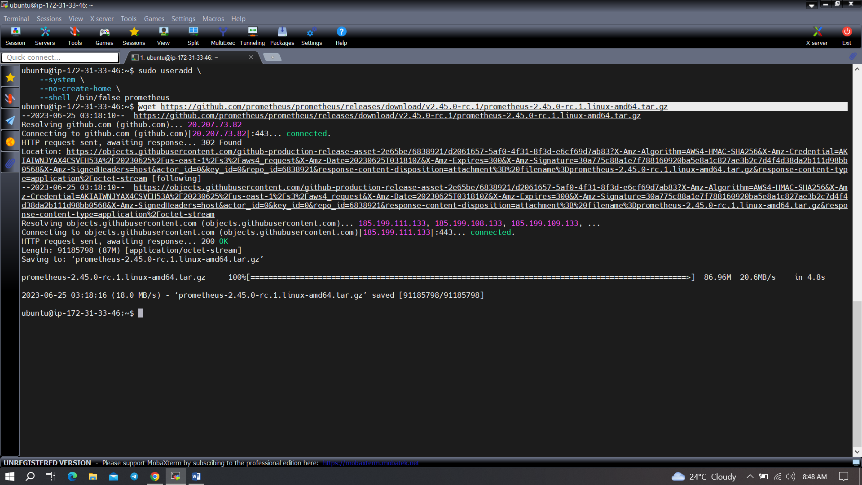
--shell /bin/false prometheus

--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 Prometheus user and a group with the exact same name.

Let's check the latest version of Prometheus from the download [page](https://prometheus.io/download/).

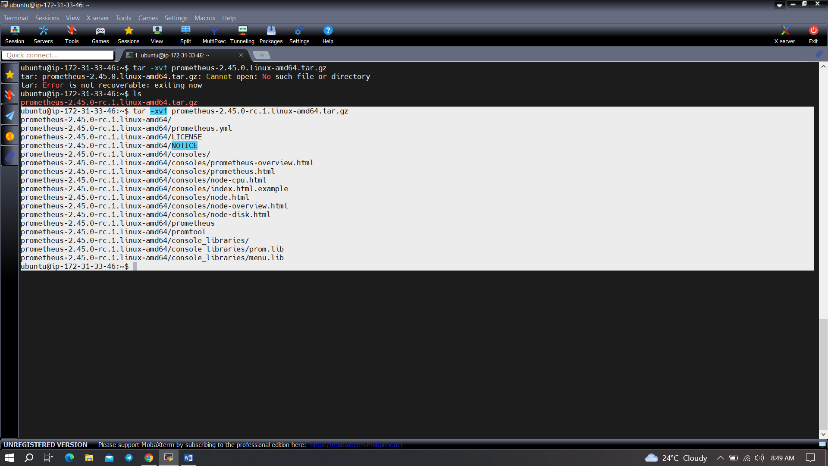
You can use the curl or wget command to download Prometheus.

wget <https://github.com/prometheus/prometheus/releases/download/v2.45.0-rc.1/prometheus-2.45.0-rc.1.linux-amd64.tar.gz>



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

tar -xvf prometheus-2.45.0-rc.1.linux-amd64.tar.gz

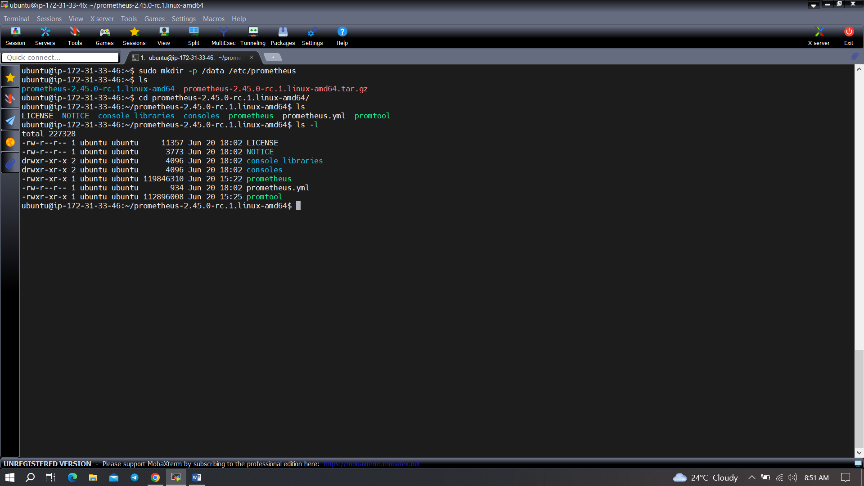


Usually, you would have a disk mounted to the data directory. For this tutorial, I will simply create a /data director. Also, you need a folder for Prometheus configuration files.

sudo mkdir -p /data /etc/prometheus

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

cd prometheus-2.45.0-rc.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/

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.

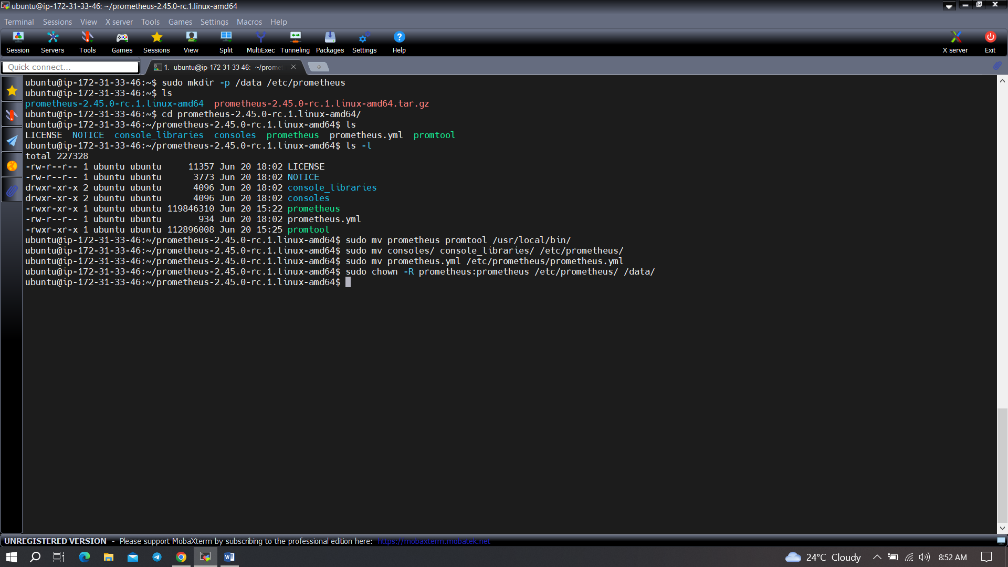
sudo mv consoles/ console\_libraries/ /etc/prometheus/

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

c

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

cd



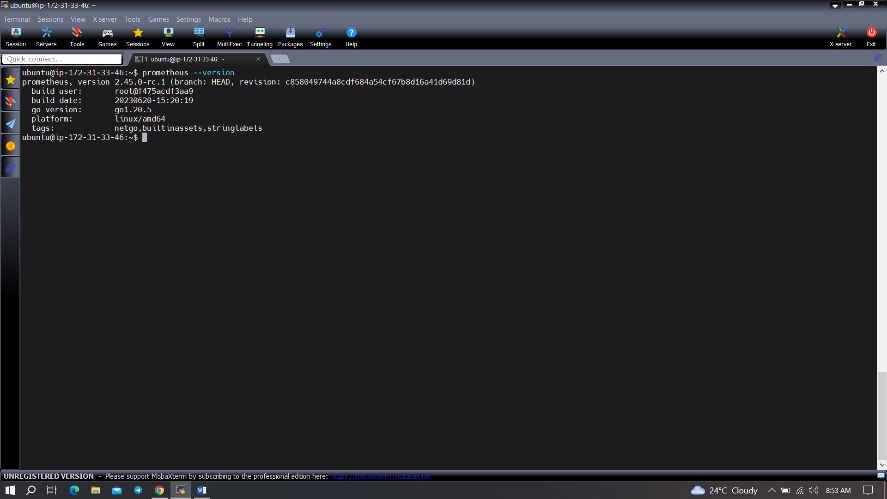
You can delete the archive and a Prometheus folder when you are done.

cd

rm -rf prometheus\*

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

prometheus –version



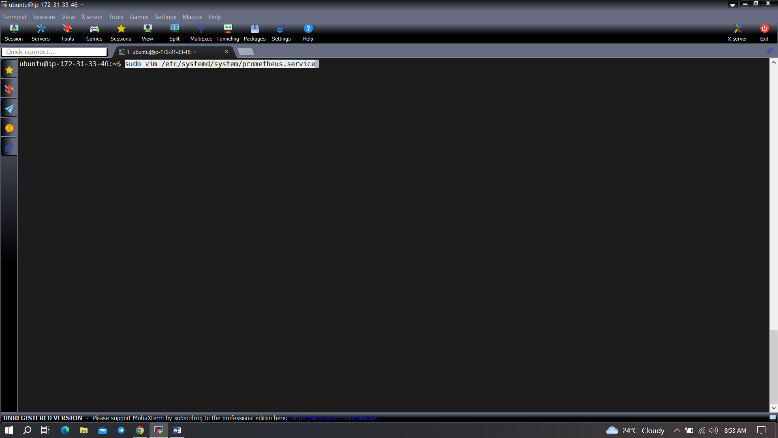
To get more information and configuration options, run Prometheus help.

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



|  |  |
| --- | --- |
| **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 |

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.

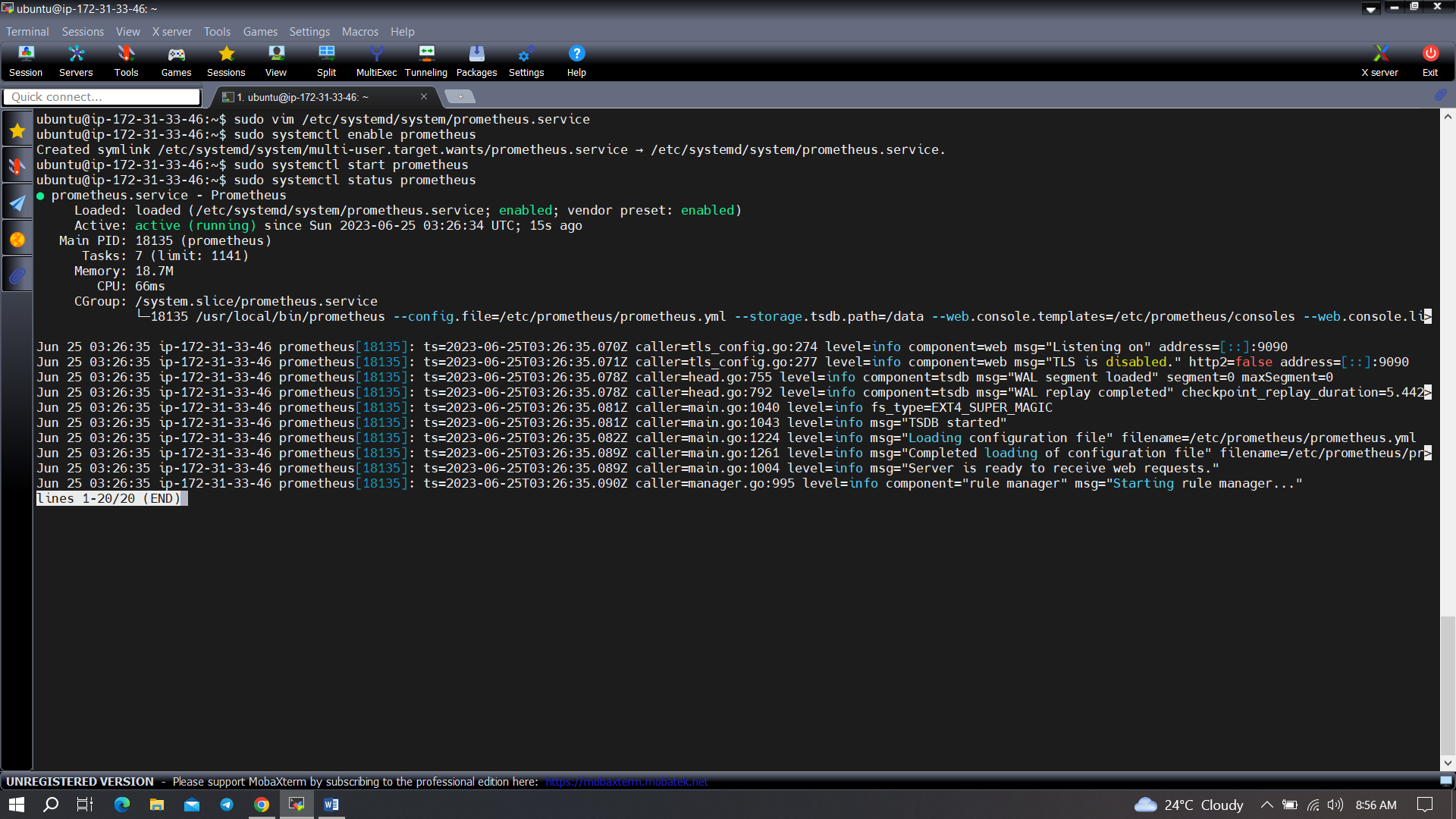
sudo systemctl enable prometheus

Then just start the Prometheus.

sudo systemctl start prometheus

To check the status of Prometheus run following command:

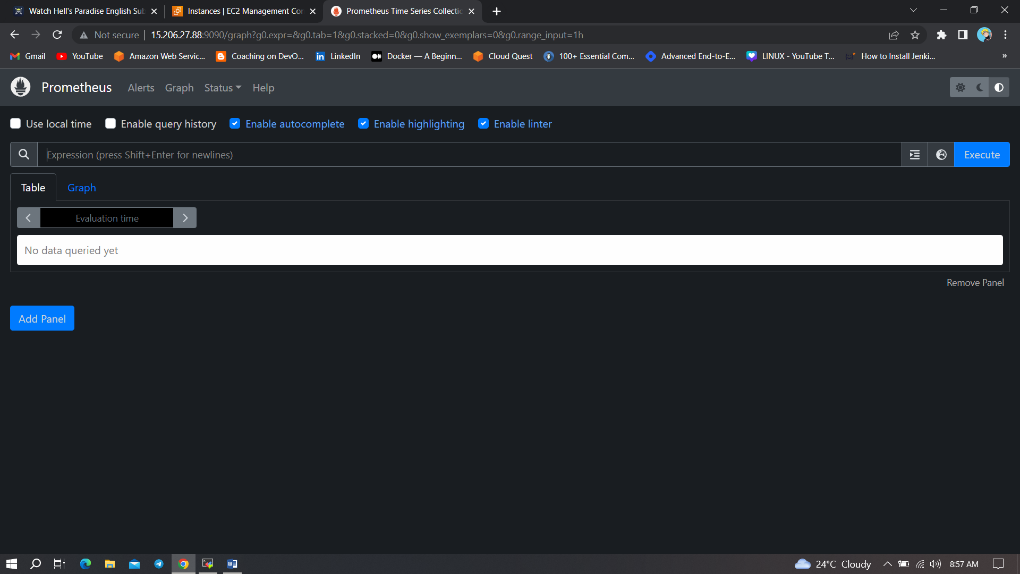
sudo systemctl status Prometheus



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 browser. I'm going to be using the IP address of the Ubuntu server. You need to append port 9090 to the IP. For example http://<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[¶](https://antonputra.com/monitoring/install-prometheus-and-grafana-on-ubuntu/" \l "install-node-exporter-on-ubuntu-2004)

Next, we're going to set up and configured 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 \

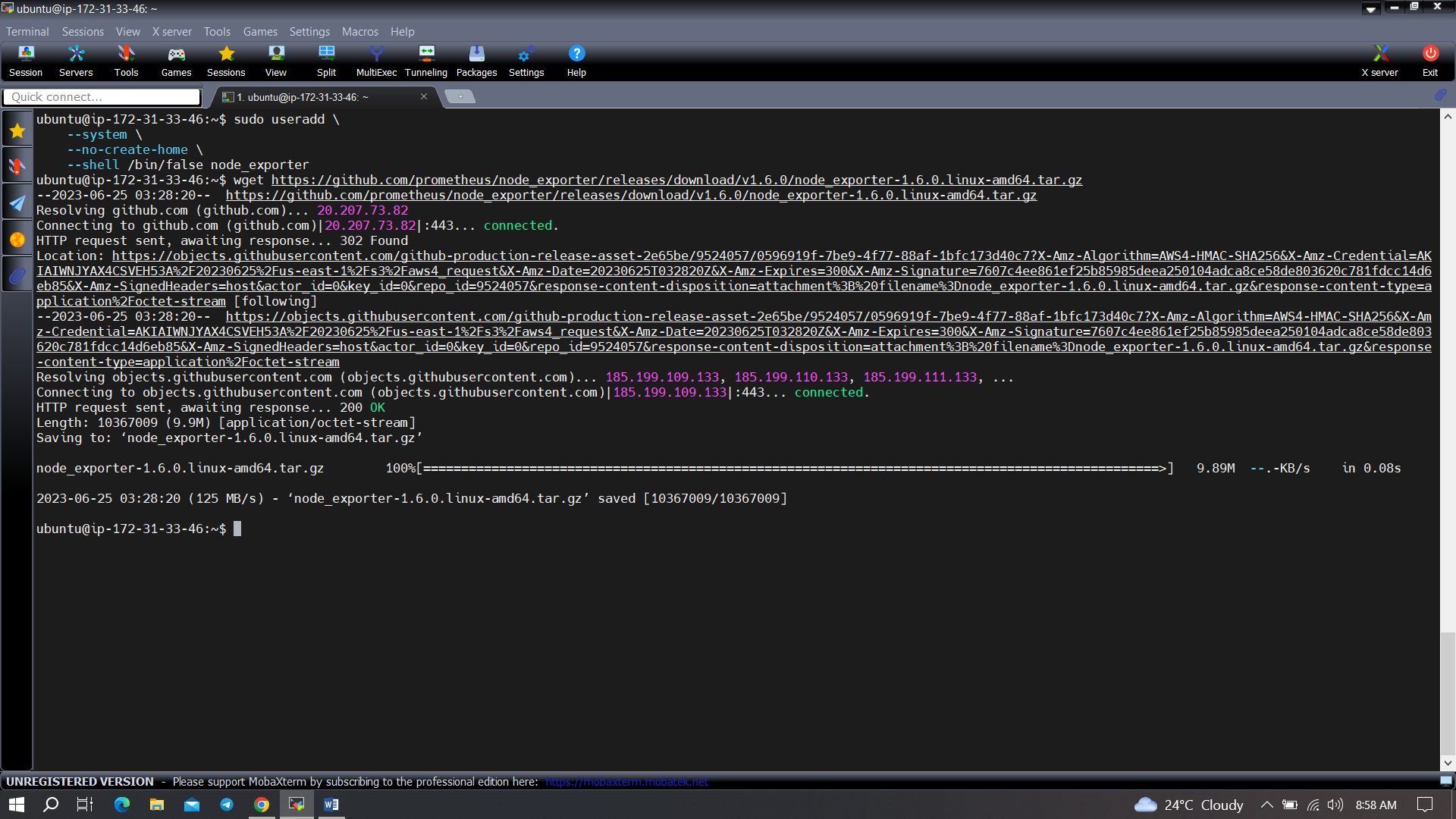
--no-create-home \

--shell /bin/false node\_exporter

You can download Node Exporter from the same [page](https://prometheus.io/download/).

Use wget command to download binary.

wget <https://github.com/prometheus/node_exporter/releases/download/v1.6.0/node_exporter-1.6.0.linux-amd64.tar.gz>



Extract node exporter from the archive.

tar -xvf node\_exporter-1.6.0.linux-amd64.tar.gz

Move binary to the /usr/local/bin.

sudo mv \

node\_exporter-1.6.0.linux-amd64/node\_exporter \

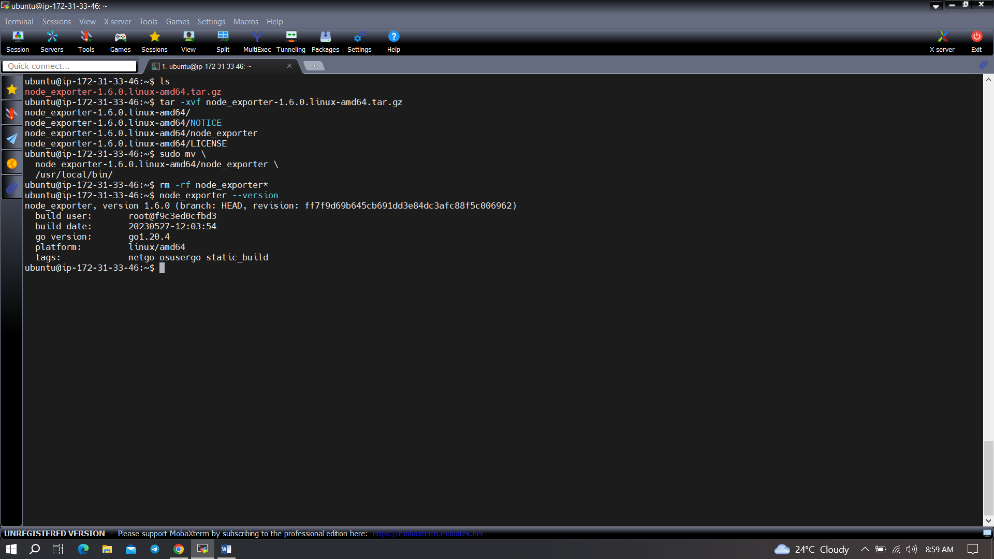
/usr/local/bin

Clean up, delete node\_exporter archive and a folder.

rm -rf node\_exporter\*

Verify that you can run the binary.

node\_exporter --version



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 login controller, just for the demo.

Next, create similar systemd unit file.

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 |

Replace Prometheus user and group to node\_exporter, and update ExecStart command.

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

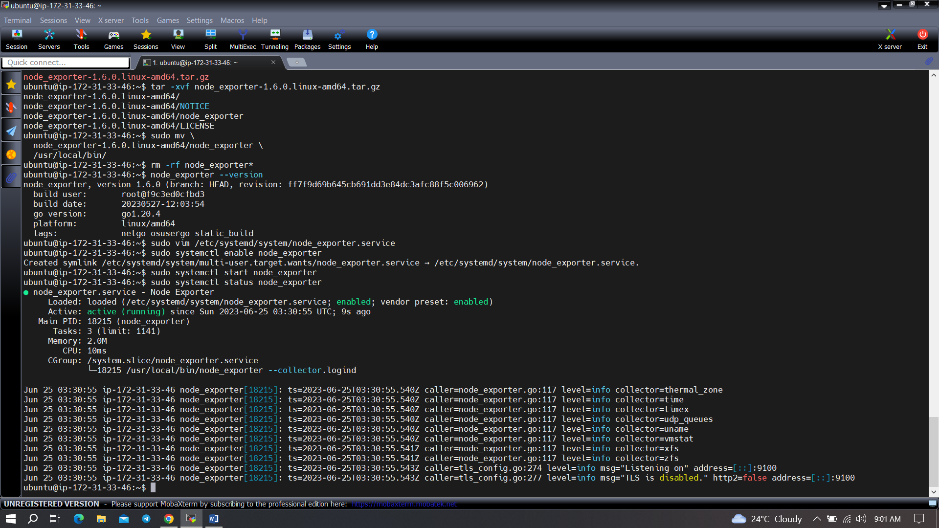
sudo systemctl enable node\_exporter

Then start the Node Exporter.

sudo systemctl start node\_exporter

Check the status of Node Exporter with the following command:

sudo systemctl status node\_exporter



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

|  |  |
| --- | --- |
| **prometheus.yml** | |
|  | ...  - 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 Prometheus config without restarting the service and causing the downtime.

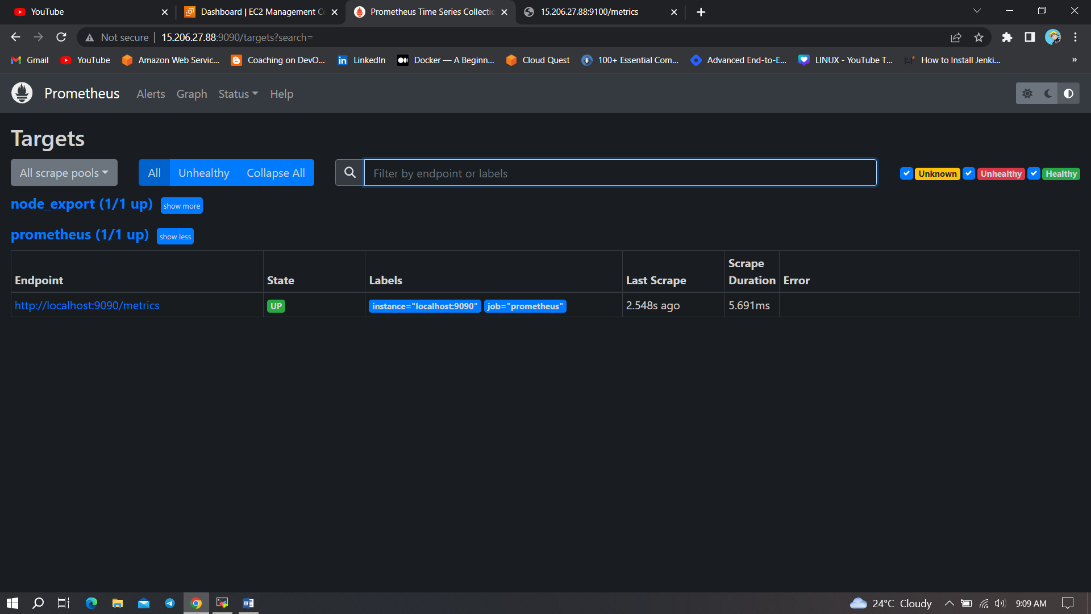
Before, restarting check if the config is valid.

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

Check the targets section http://<ip>:9090/targets



Install Grafana on Ubuntu 22.04[¶](https://antonputra.com/monitoring/install-prometheus-and-grafana-on-ubuntu/" \l "install-grafana-on-ubuntu-2004)

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

Next, add GPG key.

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

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

After you add the repository, update and install Garafana.

sudo apt-get update

sudo apt-get -y install grafana

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

sudo systemctl enable grafana-server

Then start the Grafana.

sudo systemctl start grafana-server

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

sudo systemctl status grafana-server

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. Let's use devops123 for the new 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. 