Grafana, Prometheus, and Node Exporter Installation Guide for Kali Linux

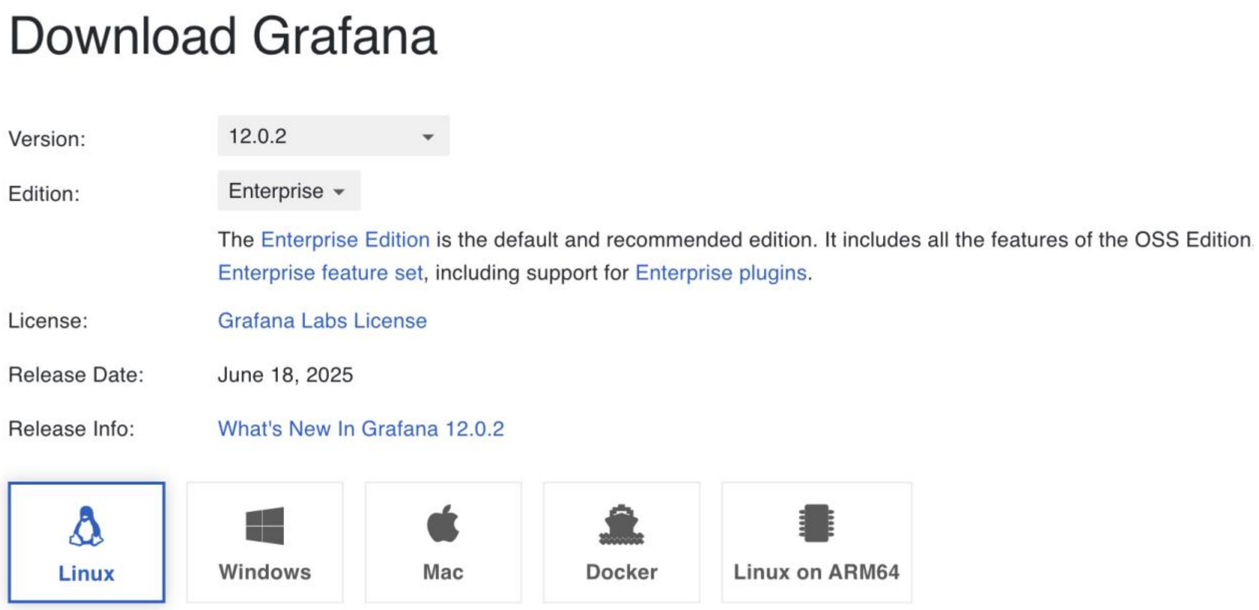
Table of Contents

Introduction

Using **Grafana** and **Prometheus** to build a monitoring system offers powerful benefits for system administrators and cybersecurity professionals. **Prometheus** efficiently collects and stores real-time metrics from servers, applications, and network devices, enabling proactive detection of performance issues or security threats. **Grafana** complements it by providing interactive and customizable dashboards that visualize these metrics clearly, making it easier to analyze trends and spot anomalies. Together, they help improve **system reliability**, **performance monitoring**, and **incident response**, while reducing downtime through real-time alerting and insights—all in an open-source, cost-effective solution.

Prerequisites

- Linux ,Mac, Windows or Docker
- Internet connection.
- Terminal access with sudo privileges.



Download Grafana

Version: 12.0.2

Edition: Enterprise

The Enterprise Edition is the default and recommended edition. It includes all the features of the OSS Edition
Enterprise feature set, including support for Enterprise plugins.

License: Grafana Labs License

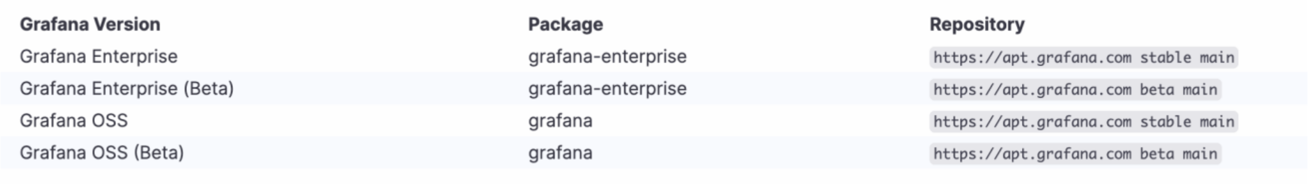
Release Date: June 18, 2025

Release Info: What's New In Grafana 12.0.2

Installation Steps

(Choose correct installation steps according to your system , this is for aarch64 / Macbook air M1/M2)

1. Install Grafana



Grafana Version	Package	Repository
Grafana Enterprise	grafana-enterprise	`https://apt.grafana.com stable main`
Grafana Enterprise (Beta)	grafana-enterprise	`https://apt.grafana.com beta main`
Grafana OSS	grafana	`https://apt.grafana.com stable main`
Grafana OSS (Beta)	grafana	`https://apt.grafana.com beta main`

https://grafana.com/docs/grafana/latest/setup-grafana/installation/debian/

1. Open a terminal and update the package lists:

```
sudo apt-get update
```

2. Add the Grafana repository:

```
sudo apt-get install -y apt-transport-https software-properties-common
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee
/etc/apt/sources.list.d/grafana.list
```

3. Install Grafana:

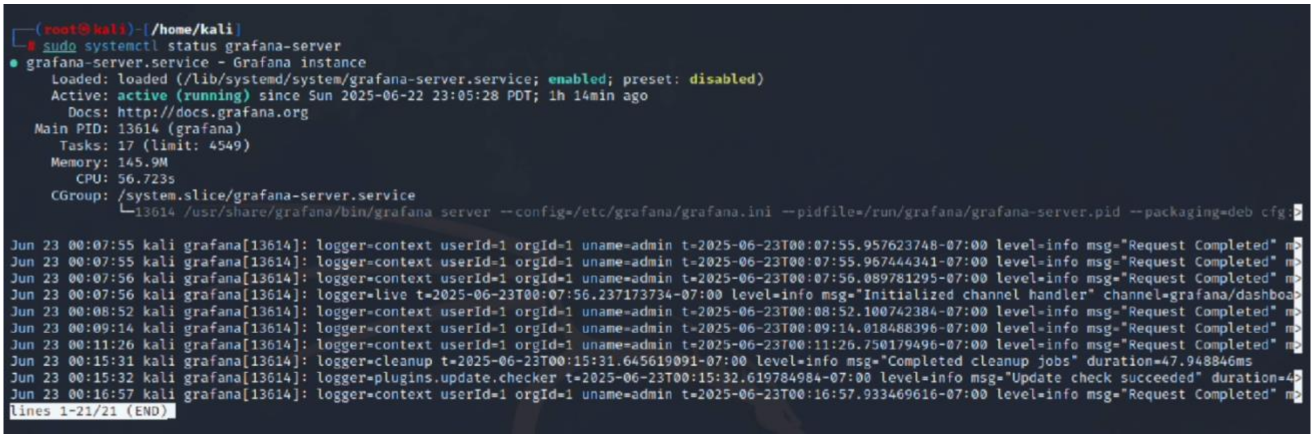
```
sudo apt-get update
sudo apt-get install Grafana
```

4. Start and enable Grafana:

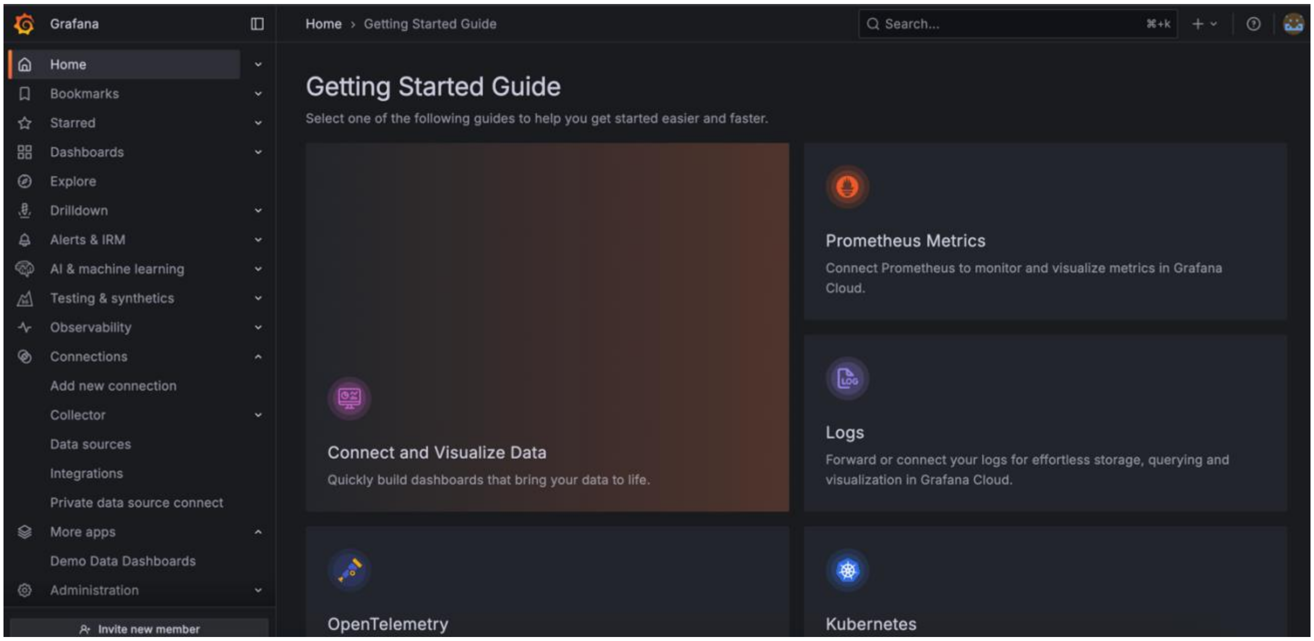
sudo systemctl start grafana-server
sudo systemctl enable grafana-server

5. Verify:

Open http://localhost:3000 in a browser. Log in with admin/admin (change password on first login).



```
┌──(root㉿kali)-[/home/kali]
└─# sudo systemctl status grafana-server
● grafana-server.service - Grafana instance
     Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; preset: disabled)
     Active: active (running) since Sun 2025-06-22 23:05:28 PDT; 1h 14min ago
       Docs: http://docs.grafana.org
   Main PID: 13614 (grafana)
      Tasks: 17 (limit: 4549)
     Memory: 145.9M
        CPU: 56.723s
     CGroup: /system.slice/grafana-server.service
             └─13614 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packaging=deb cfg:>

Jun 23 00:07:55 kali grafana[13614]: logger=context userId=1 orgId=1 uname=admin t=2025-06-23T00:07:55.957623748-07:00 level=info msg="Request Completed" m>
Jun 23 00:07:55 kali grafana[13614]: logger=context userId=1 orgId=1 uname=admin t=2025-06-23T00:07:55.967444341-07:00 level=info msg="Request Completed" m>
Jun 23 00:07:56 kali grafana[13614]: logger=context userId=1 orgId=1 uname=admin t=2025-06-23T00:07:56.089781295-07:00 level=info msg="Request Completed" m>
Jun 23 00:07:56 kali grafana[13614]: logger=live t=2025-06-23T00:07:56.237173734-07:00 level=info msg="Initialized channel handler" channel=grafana/dashboa>
Jun 23 00:08:52 kali grafana[13614]: logger=context userId=1 orgId=1 uname=admin t=2025-06-23T00:08:52.100742384-07:00 level=info msg="Request Completed" m>
Jun 23 00:09:14 kali grafana[13614]: logger=context userId=1 orgId=1 uname=admin t=2025-06-23T00:09:14.018488396-07:00 level=info msg="Request Completed" m>
Jun 23 00:11:26 kali grafana[13614]: logger=context userId=1 orgId=1 uname=admin t=2025-06-23T00:11:26.750179496-07:00 level=info msg="Request Completed" m>
Jun 23 00:15:31 kali grafana[13614]: logger=cleanup t=2025-06-23T00:15:31.645619091-07:00 level=info msg="Completed cleanup jobs" duration=47.948846ms
Jun 23 00:15:32 kali grafana[13614]: logger=plugins.update.checker t=2025-06-23T00:15:32.619784984-07:00 level=info msg="Update check succeeded" duration=4>
Jun 23 00:16:57 kali grafana[13614]: logger=context userId=1 orgId=1 uname=admin t=2025-06-23T00:16:57.933469616-07:00 level=info msg="Request Completed" m>
lines 1-21/21 (END)
```



Getting Started Guide

Select one of the following guides to help you get started easier and faster.

Connect and Visualize Data

Quickly build dashboards that bring your data to life.

Prometheus Metrics

Connect Prometheus to monitor and visualize metrics in Grafana Cloud.

Logs

Forward or connect your logs for effortless storage, querying and visualization in Grafana Cloud.

OpenTelemetry

Kubernetes

2. Install Prometheus

1) Download the aarch64 version (e.g., 2.54.1): download correct version according to yours

```
VERSION=2.54.1
wget https://github.com/prometheus/prometheus/releases/download/v${VERS
tar xvfz prometheus-${VERSION}.linux-arm64.tar.gz
sudo mv prometheus-${VERSION}.linux-arm64 /opt/prometheus
```

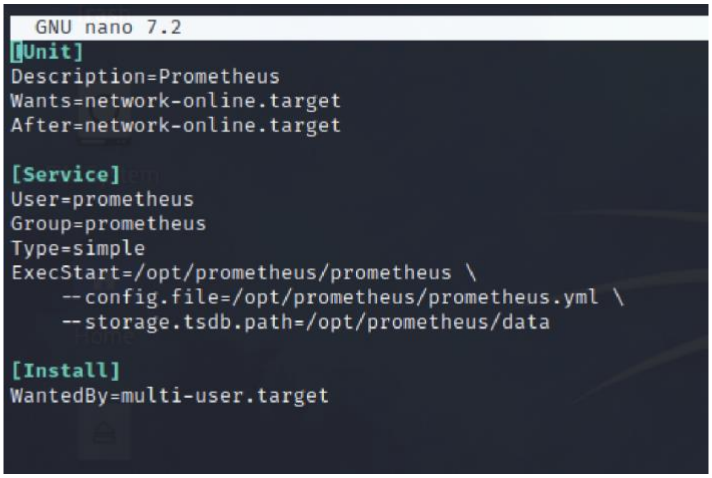
2) Create a user and set permissions:

```
sudo useradd --no-create-home --shell /bin/false prometheus
sudo chown -R prometheus:prometheus /opt/prometheus
```

3) Create a systemd service file:

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

Add this:



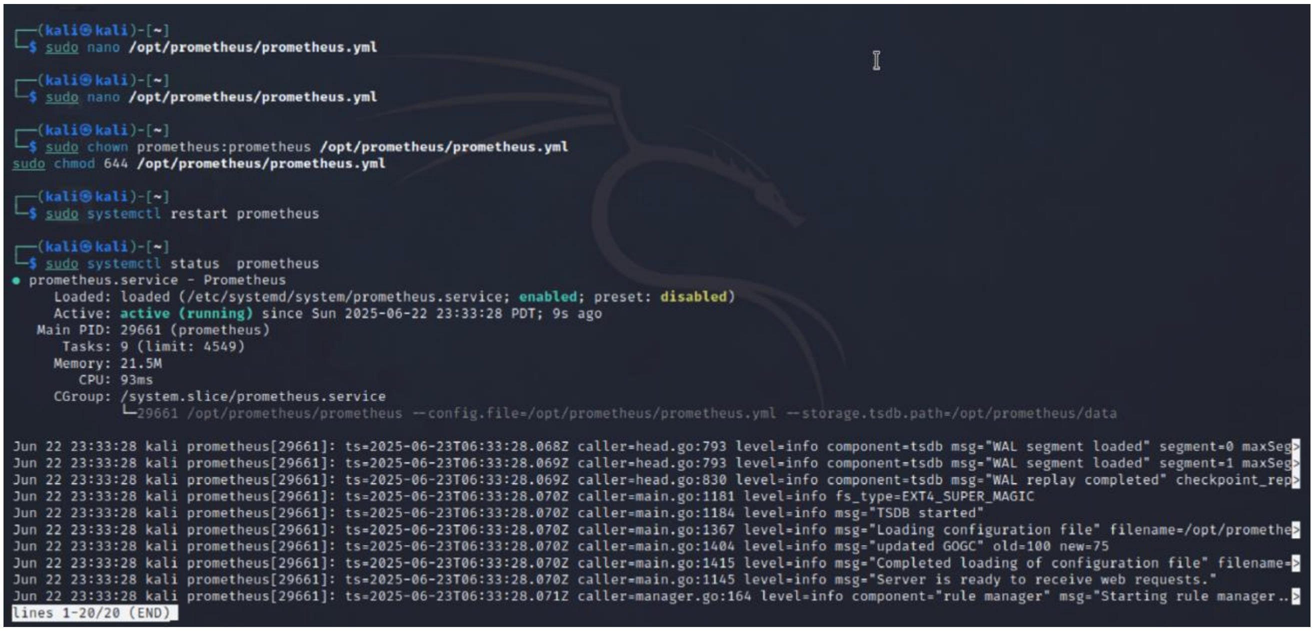
```
  GNU nano 7.2
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/opt/prometheus/prometheus \
    --config.file=/opt/prometheus/prometheus.yml \
    --storage.tsdb.path=/opt/prometheus/data

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

```
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target
[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/opt/prometheus/prometheus \
--config.file=/opt/prometheus/prometheus.yml \
--storage.tsdb.path=/opt/prometheus/data
[Install]
WantedBy=multi-user.target
```

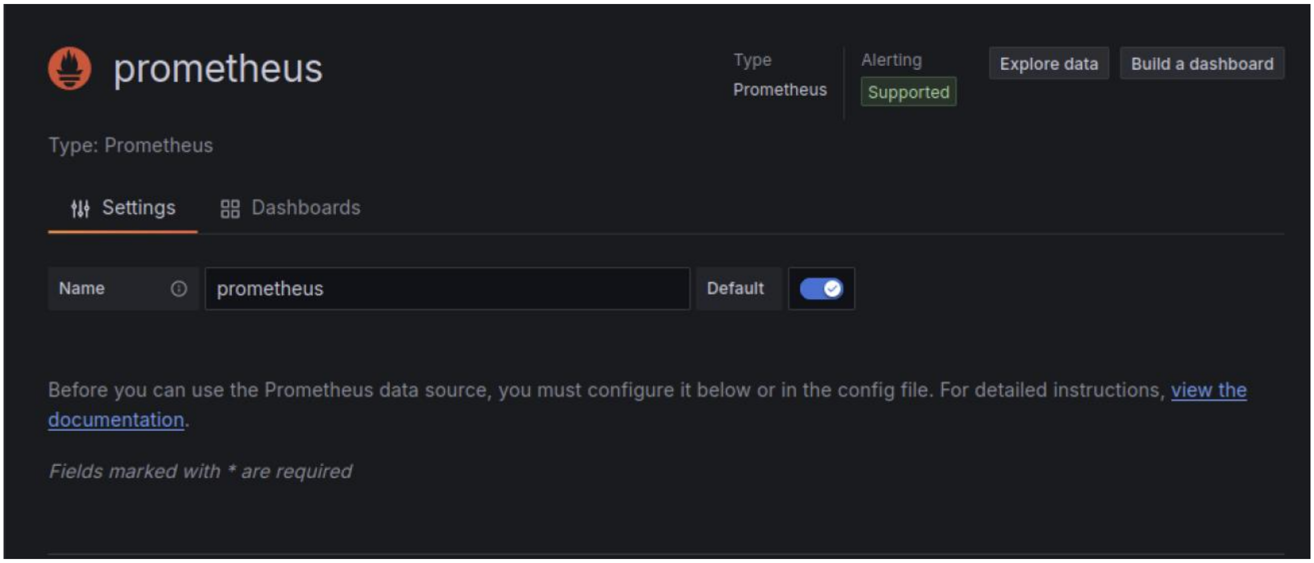
4) Start and enable Prometheus:

```
sudo systemctl daemon-reload
sudo systemctl start prometheus
sudo systemctl enable Prometheus
```

5) Verify: Open http://localhost:9090 or run curl http://localhost:9090.



```
┌──(kali㉿kali)-[~]
└─$ sudo nano /opt/prometheus/prometheus.yml

┌──(kali㉿kali)-[~]
└─$ sudo nano /opt/prometheus/prometheus.yml

┌──(kali㉿kali)-[~]
└─$ sudo chown prometheus:prometheus /opt/prometheus/prometheus.yml
sudo chmod 644 /opt/prometheus/prometheus.yml

┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart prometheus

┌──(kali㉿kali)-[~]
└─$ sudo systemctl status  prometheus
● prometheus.service - Prometheus
     Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; preset: disabled)
     Active: active (running) since Sun 2025-06-22 23:33:28 PDT; 9s ago
   Main PID: 29661 (prometheus)
      Tasks: 9 (limit: 4549)
     Memory: 21.5M
        CPU: 93ms
     CGroup: /system.slice/prometheus.service
             └─29661 /opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml --storage.tsdb.path=/opt/prometheus/data

Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.068Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSeg>
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.069Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=1 maxSeg>
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.069Z caller=head.go:830 level=info component=tsdb msg="WAL replay completed" checkpoint_rep>
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.070Z caller=main.go:1181 level=info fs_type=EXT4_SUPER_MAGIC
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.070Z caller=main.go:1184 level=info msg="TSDB started"
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.070Z caller=main.go:1367 level=info msg="Loading configuration file" filename=/opt/promethe>
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.070Z caller=main.go:1404 level=info msg="updated GOGC" old=100 new=75
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.070Z caller=main.go:1415 level=info msg="Completed loading of configuration file" filename=>
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.070Z caller=main.go:1145 level=info msg="Server is ready to receive web requests."
Jun 22 23:33:28 kali prometheus[29661]: ts=2025-06-23T06:33:28.071Z caller=manager.go:164 level=info component="rule manager" msg="Starting rule manager..>
lines 1-20/20 (END)
```



Type: Prometheus

Alerting: Supported

Type: Prometheus

Name: prometheus

Default: ●

Before you can use the Prometheus data source, you must configure it below or in the config file. For detailed instructions, view the documentation.

*Fields marked with * are required*

3. Install Node Exporter

1. Download the aarch64 version (e.g., 1.8.2): **Choose your one**

```
VERSION=1.8.2  # Replace with the latest version if different wget
https://github.com/prometheus/node_exporter/releases/download/v${VERSION}
/node_exporter-${VERSION}.linux-amd64.tar.gz
tar xvfz node_exporter-${VERSION}.linux-amd64.tar.gz
sudo mv node_exporter-${VERSION}.linux-amd64/node_exporter /usr/local/bin/
```

2. Create a user and set permissions:

```
sudo useradd --no-create-home --shell /bin/false node_exporter
sudo chown node_exporter:node_exporter /usr/local/bin/node_exporter
```

3. Create a systemd service file:

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

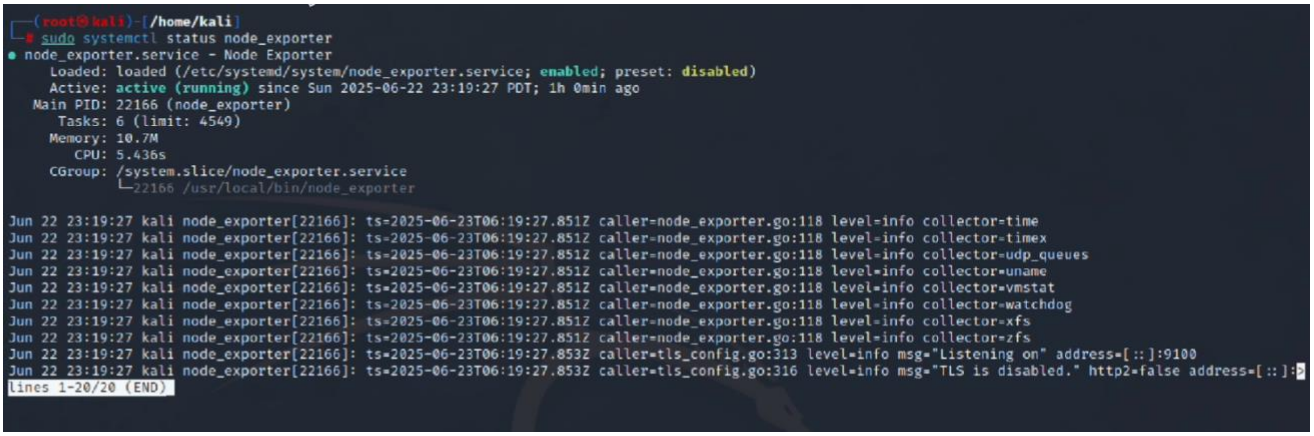
Add this:

```
[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
```

4. Save , exit and Enable Node Exporter

```
sudo systemctl daemon-reload
sudo systemctl start node_exporter
sudo systemctl enable node_exporter
```

Verify:
Run curl http://localhost:9100/metrics.



```
┌──(root㉿kali)-[/home/kali]
└─# sudo systemctl status node_exporter
● node_exporter.service - Node Exporter
     Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; preset: disabled)
     Active: active (running) since Sun 2025-06-22 23:19:27 PDT; 1h 0min ago
   Main PID: 22166 (node_exporter)
      Tasks: 6 (limit: 4549)
     Memory: 10.7M
        CPU: 5.436s
     CGroup: /system.slice/node_exporter.service
             └─22166 /usr/local/bin/node_exporter

Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=time
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=timex
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=udp_queues
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=uname
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=vmstat
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=watchdog
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=xfs
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.851Z caller=node_exporter.go:118 level=info collector=zfs
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.853Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9100
Jun 22 23:19:27 kali node_exporter[22166]: ts=2025-06-23T06:19:27.853Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:>
lines 1-20/20 (END)
```

4.Configure Prometheus

1. Edit the configuration file:

```
sudo nano /opt/prometheus/prometheus.yml
```

Edit and add this under Scrape_Configs: (Just add this)

```
- job_name: 'node_exporter'
static_configs:
- targets: ['localhost:9100']
```

2. Restart Prometheus:

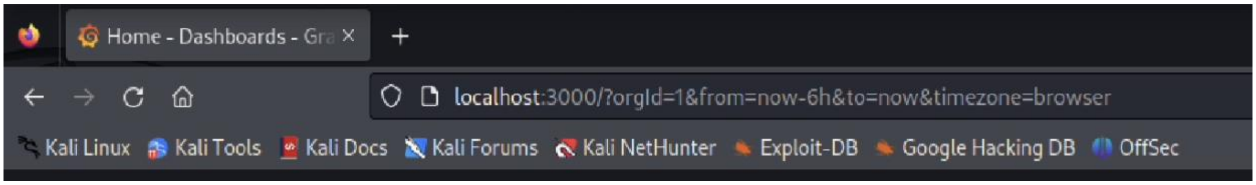
```
sudo systemctl restart Prometheus
```

3. Verify:

Open http://localhost:9090/targets

Configure Grafana

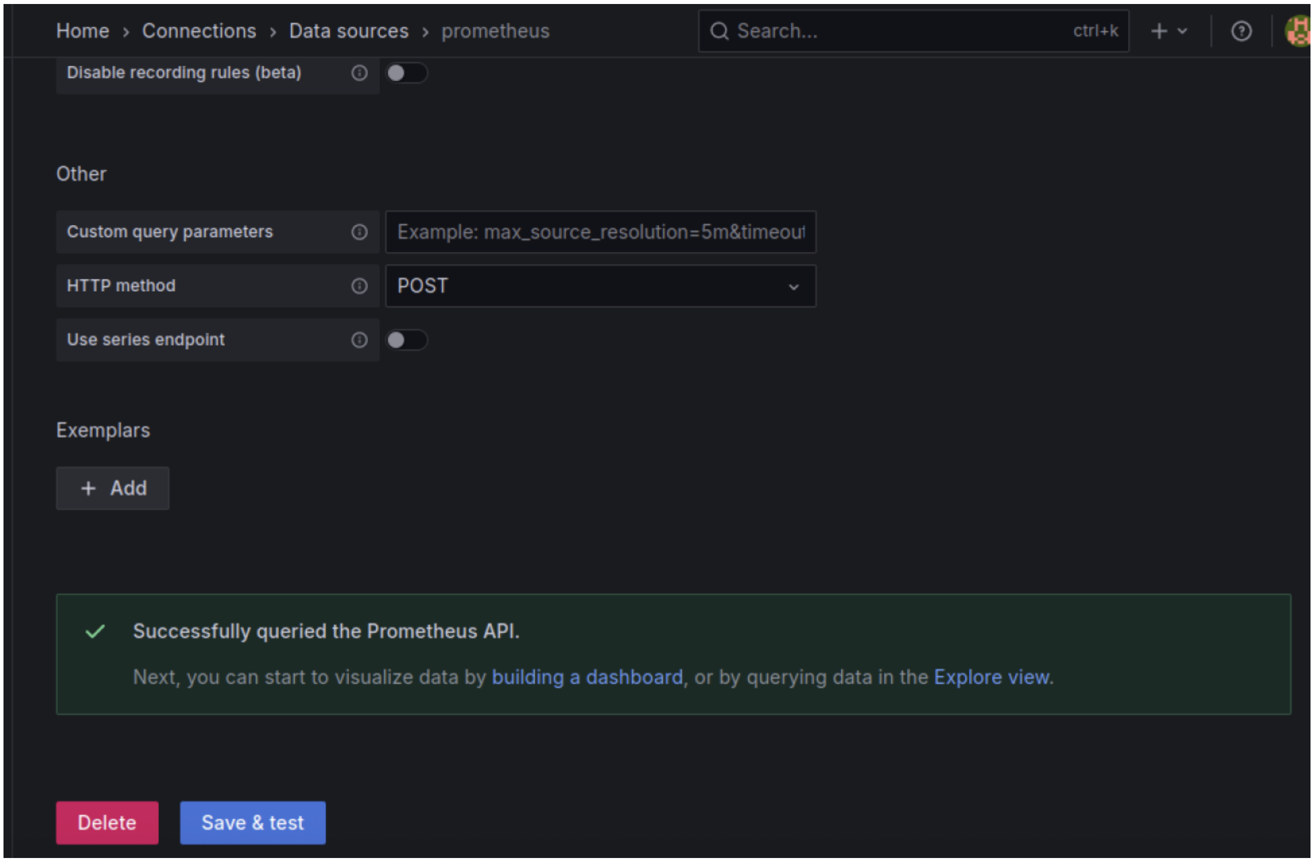
1. Log in to Grafana at http://localhost:3000.



localhost:3000/?orgId=1&from=now-6h&to=now&timezone=browser

2. Add Prometheus as a data source: -

Go to Settings > Data Sources > Add data source. - Select Prometheus, set URL to http://localhost:9090. - Click Save & Test.



Disable recording rules (beta) ○

Other

Custom query parameters: Example: max_source_resolution=5m&timeout

HTTP method: POST

Use series endpoint ○

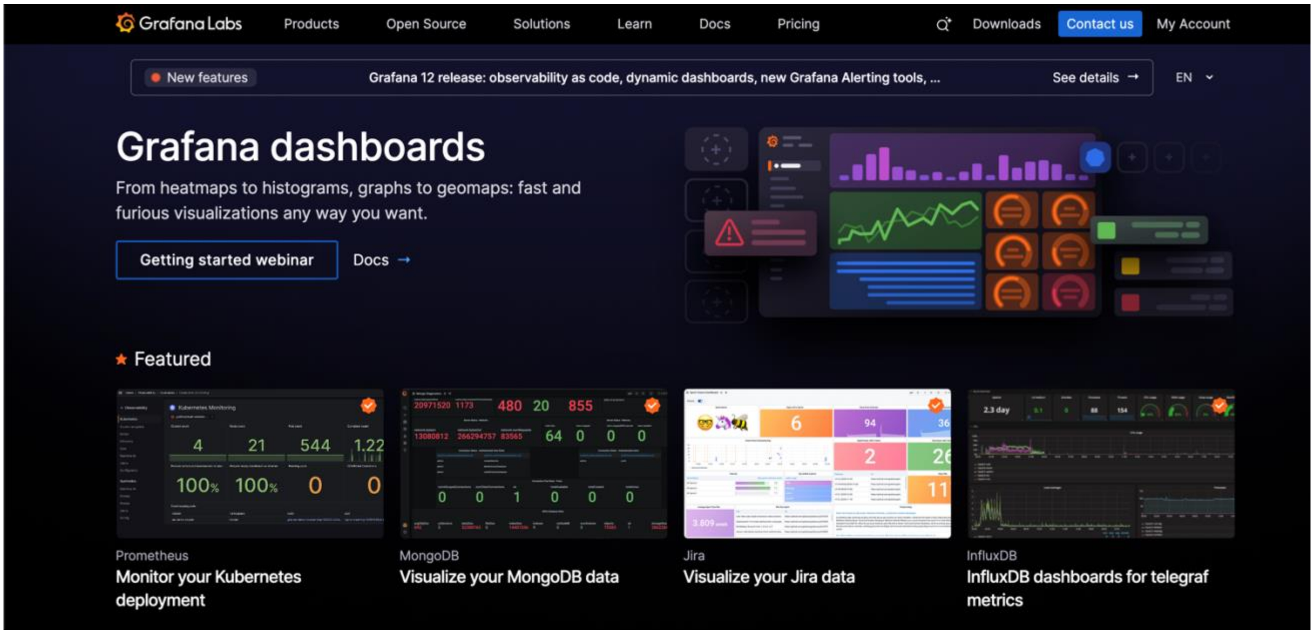
Exemplars

✓ Successfully queried the Prometheus API.

Next, you can start to visualize data by building a dashboard, or by querying data in the Explore view.

3. Import a dashboard: - Click + > Dashboard > New > Import.

https://grafana.com/grafana/dashboards/



Grafana 12 release: observability as code, dynamic dashboards, new Grafana Alerting tools, ...

Grafana dashboards

From heatmaps to histograms, graphs to geomaps: fast and furious visualizations any way you want.

Prometheus
Monitor your Kubernetes deployment

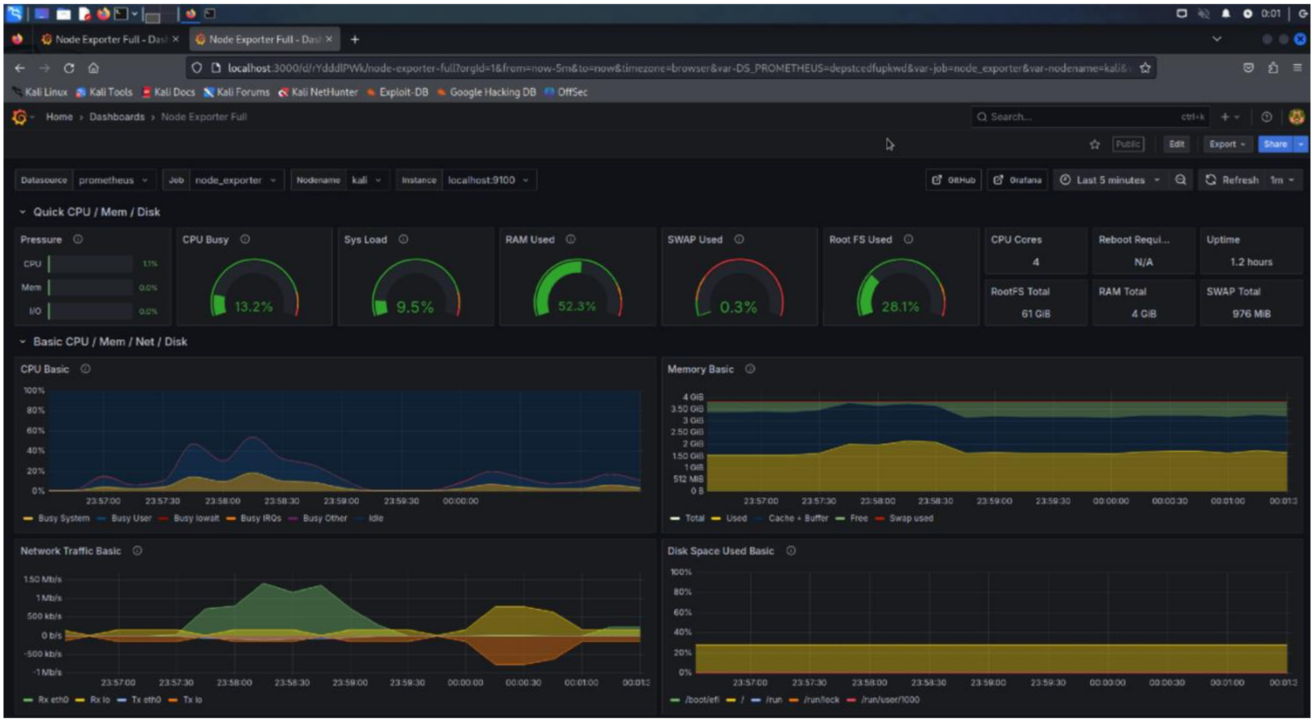
MongoDB
Visualize your MongoDB data

Jira
Visualize your Jira data

InfluxDB
InfluxDB dashboards for telegraf metrics

Select the Prometheus data source, click Load, then Import.

4. Verify: Check the dashboard for metrics.



localhost:3000/d/rYdddlPWk/node-exporter-full?orgId=1&from=now-5m&to=now&timezone=browser&var-DS_PROMETHEUS=depstcedfupkwd&var-job=node_exporter&var-nodename=kali&

Datasource prometheus | Job node_exporter | Nodename kali | Instance localhost:9100

Pressure: CPU 1.1%, Mem 0.0%, I/O 0.0%
CPU Busy 13.2%
Sys Load 9.5%
RAM Used 52.3%
SWAP Used 0.3%
Root FS Used 28.1%
CPU Cores 4
Reboot Requi... N/A
Uptime 1.2 hours
RootFS Total 61 GiB
RAM Total 4 GiB
SWAP Total 976 MiB

Troubleshooting

• Grafana Won’t Start: Check status with sudo systemctl status grafana-server. Ensure port 3000 is open (sudo ufw allow 3000).

• Prometheus Fails with ”Exec format error”: Verify you downloaded the aarch64 binary. Reinstall with the correct version.

• Prometheus ”Invalid argument” Error: Check prometheus.yml for syntax errors (use 2-space indentation).

• Connection Refused on Port 9090: Ensure Prometheus is running (sudo systemctl status prometheus). Check port with sudo netstat -tuln | grep 9090. Open port if needed (sudo ufw allow 9090).

• No Data in Dashboard: Confirm Node Exporter is running and scraped by Prometheus (http://localhost:9090/targets). Restart services if needed.

• APT Key Errors: Add missing keys with sudo apt-key adv --keyserver keyserver.ubuntu --recv-keys <key>.

END OF THE DOCUMENT