

**Name: Ruchi Pandey**

**Semester: 2nd**

**Project -3**

**Topic:**

**Integrate Grafana with Linux Server for high cpu utilization and create a graph in Grafana.**



**Table of contents**

1. Install Grafana Agent on Ubuntu
2. Install Linux Server Integration for Grafana Cloud
3. Modify the Grafana agent yaml file
4. Install Dashboards and alerts on Grafana Cloud
5. **Install Grafana Agent on Ubuntu**

The Grafana Agent is a lightweight, open-source agent that collects metrics, logs, and traces from your Linux servers and sends them to Grafana Cloud.

So first create the directory for the apt keyrings

sudo mkdir -p

/etc/apt/keyrings/

C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\872488F88D1B2DB54D55BC8BBA2FAD1B\WhatsApp Image 2024-06-20 at 18.09.51_ce8dfc83.jpg

Download and import the Grafana GPG key using following command.

wget -q -O – https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\2823F4797102CE1A1AEC05359CC16DD9\WhatsApp Image 2024-06-20 at 18.12.06_732c0c72.jpg

Add Grafana package repository using following command.

echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee /etc/apt/sources.list.d/grafana.list

C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\D840CC5D906C3E9C84374C8919D2074E\WhatsApp Image 2024-06-20 at 18.15.19_5103529a.jpg

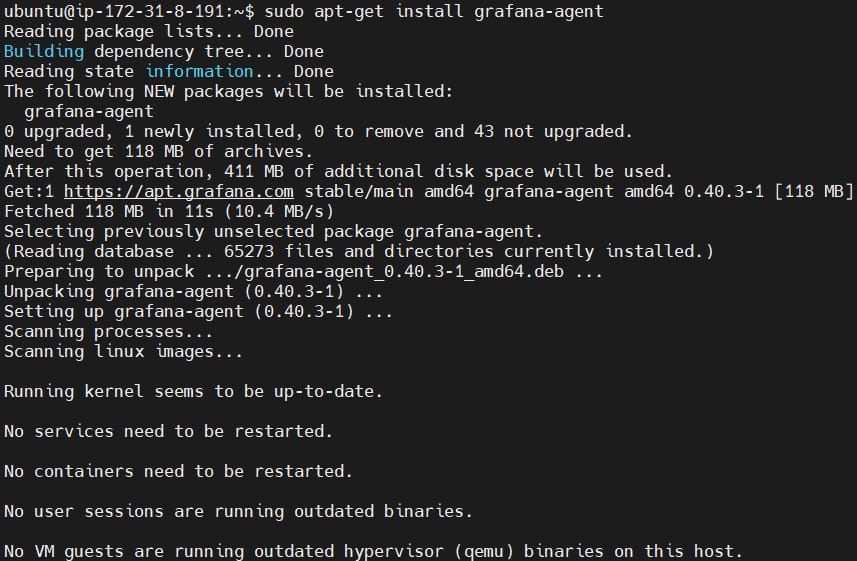
Update the repositories

sudo apt-get update

C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\4EA06FBC83CDD0A06020C35D50E1E89A\WhatsApp Image 2024-06-20 at 18.19.01_61447a55.jpg

After updating the repository. Install the Grafana Agent

sudo apt-get install grafana-agent



Start the grafana agent by running the following command

sudo systemctl start grafana-agent

C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\E07413354875BE01A996DC560274708E\WhatsApp Image 2024-06-20 at 18.26.05_a94cc77f.jpg

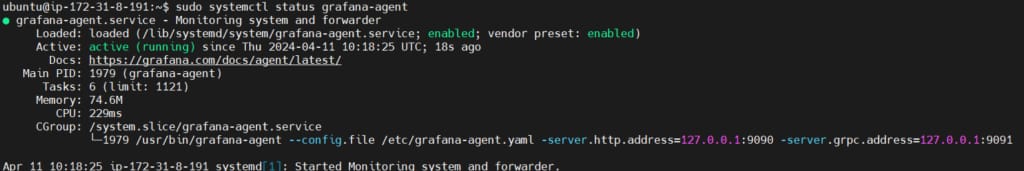
After this enable the grafana agent

sudo systemctl enable grafana-agent

C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\A8E864D04C95572D1AECE099AF852D0A\WhatsApp Image 2024-06-20 at 18.27.14_62456224.jpg

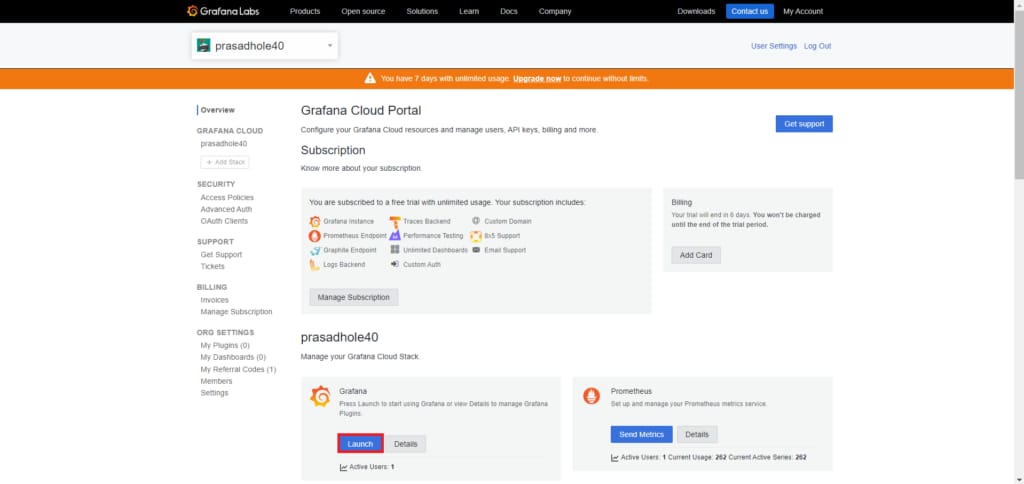
You can check if its running properly or not by running the following command.

sudo systemctl status grafana-agent

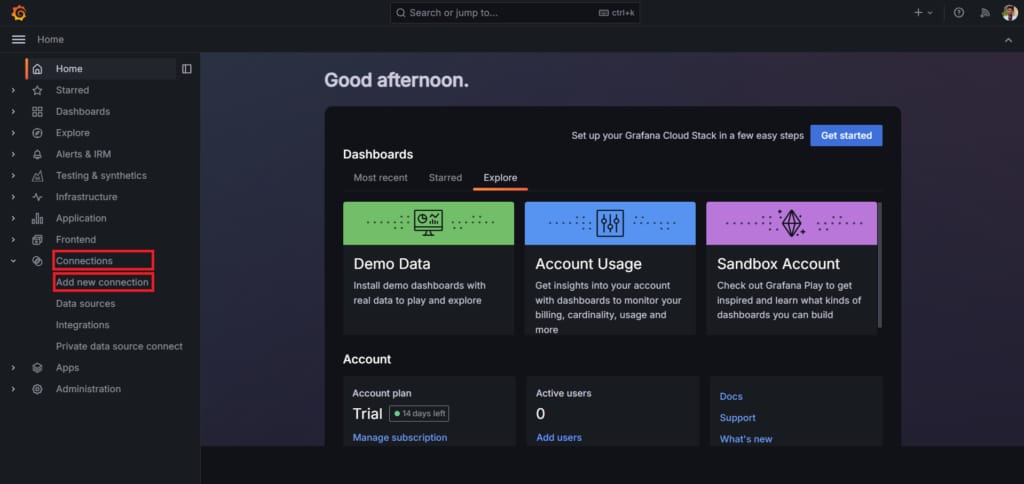


1. **Install Linux Server Integration for Grafana Cloud**

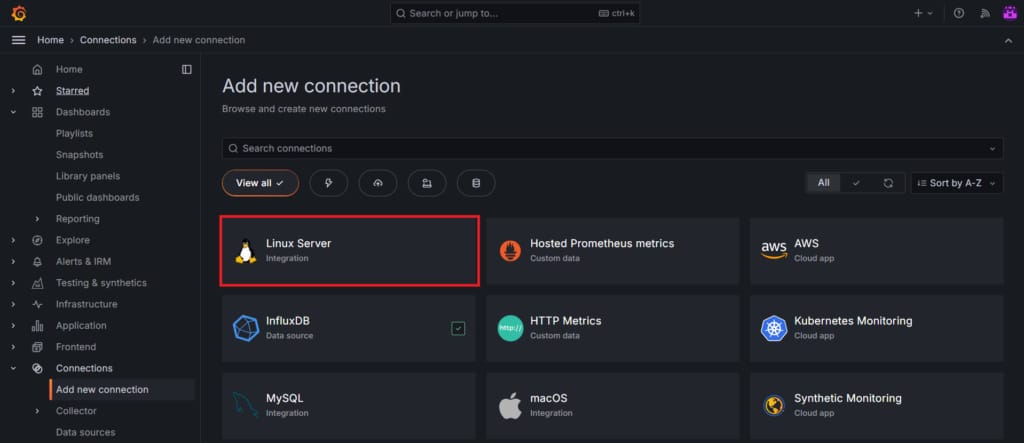
First Login to Grafana Cloud and launch the Grafana instance**.**

****

then select the Connections > Add new connections from the home bar.

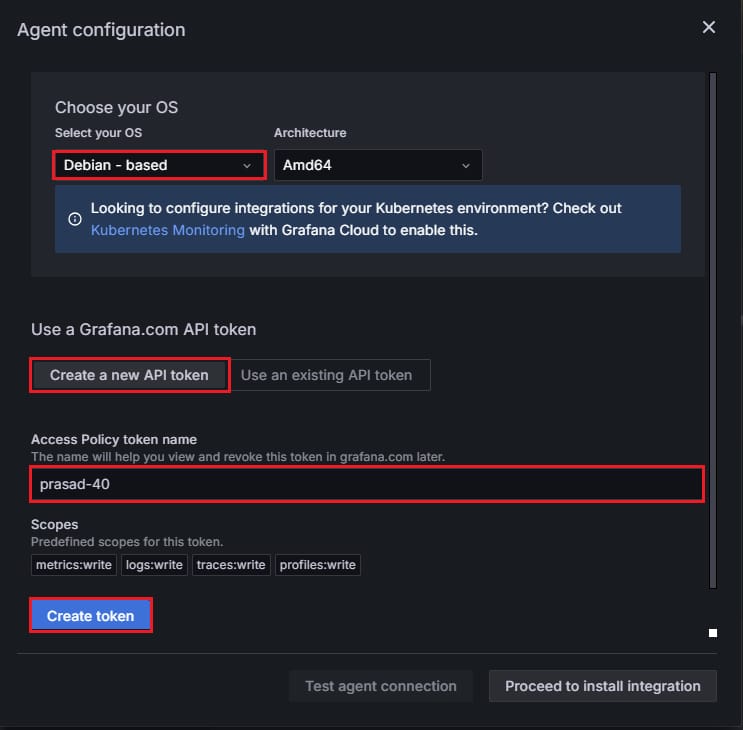


Select the Linux Server



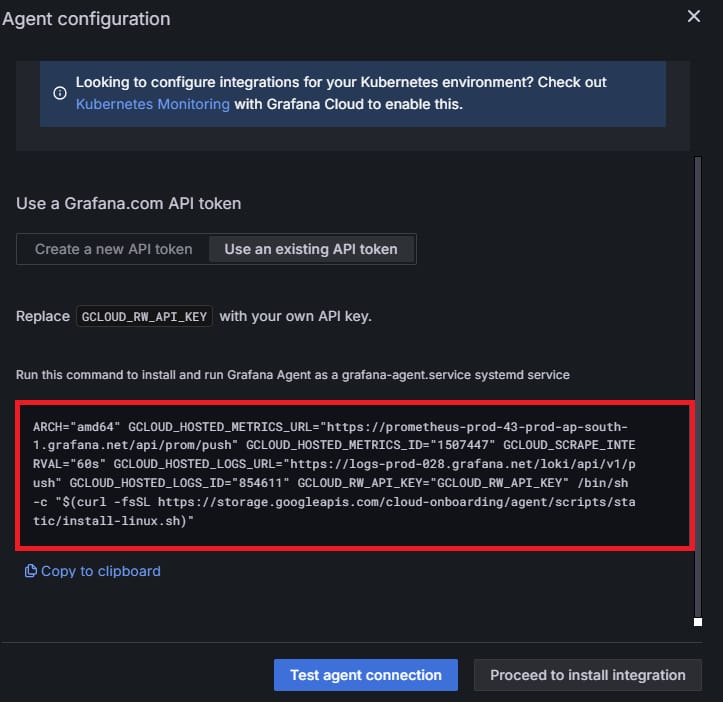
Click on Grafana agent and choose the operating system on which your linux server is running. Then click on Create a new API token or you can use an existing on if you have, give the Access policy token name and click on create token.

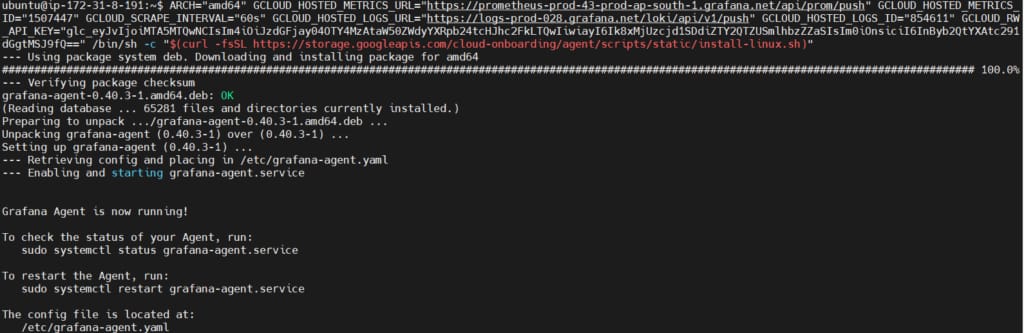
This will generate API token which we will use to set up the grafana agent.



Copy the code run it to install and run grafana agent as a grafana-agent.service systemd service.

Put the API token key generated before instead of **“GCLOUD\_RW\_API\_KEY“**

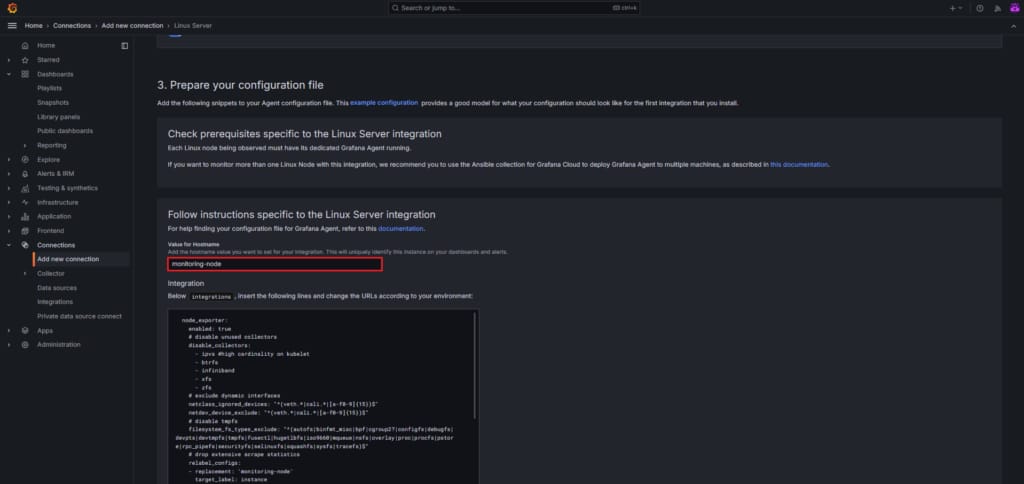
****

****

1. **Modify the Grafana agent yaml file**

Now go back to the grafana cloud and scroll down.

In prepare your configuration file give the value for hostname. Here I’ve given **monitoring-node**

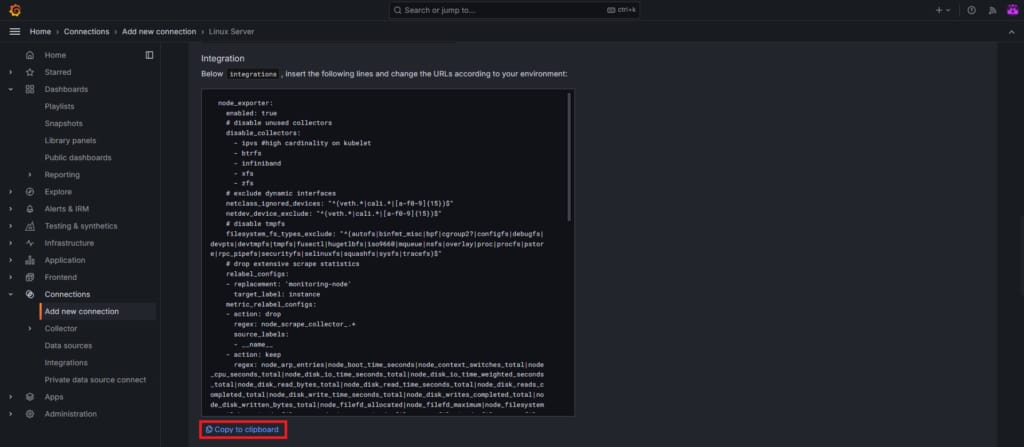


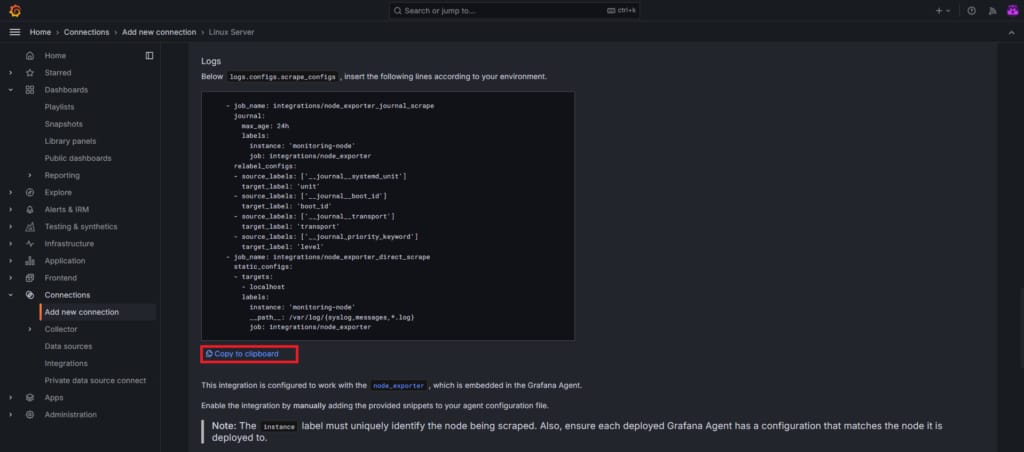
go to the /etc/grafana-agent.yaml by using following command.

sudo nano /etc/grafana-agent.yaml

C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\F7E9050C92A851B0016442AB604B0488\WhatsApp Image 2024-06-20 at 18.57.51_fc22e5a1.jpg

Then copy the Integration and logs code lines from the grafana cloud and insert it in /etc/grafana-agent.yaml file according to your environment.





**integrations:**

**prometheus\_remote\_write:**

**- basic\_auth:**

**password: glc\_eyJvIjoiMTA5MTQwNCIsIm4iOiJzdGFjay04OTY4MzAtaW50ZWdyYXRpb24tcHJhc2FkLTQwIiwiayI6Ik8xMjUzcjd1SDdiZTY2QTZUSmlhbzZZaSIsIm0iOnsiciI6InByb2QtYXAtc291dGgtMSJ9fQ==**

**username: 1507447**

**url: https://prometheus-prod-43-prod-ap-south-1.grafana.net/api/prom/push**

**agent:**

**enabled: true**

**relabel\_configs:**

**- action: replace**

**source\_labels:**

**- agent\_hostname**

**target\_label: instance**

**- action: replace**

**target\_label: job**

**replacement:**

**"integrations/agent-check"**

**metric\_relabel\_configs:**

**- action: keep**

**regex: (prometheus\_target\_sync\_length\_seconds\_sum|prometheus\_target\_scrapes\_.|prometheus\_target\_interval.|prometheus\_sd\_discovered\_targets|agent\_build.\*|agent\_wal\_samples\_appended\_total|process\_start\_time\_seconds)**

**source\_labels:**

**- \_name\_**

**# Add here any snippet that belongs to the integrations section.**

**# For a correct indentation, paste snippets copied from Grafana Cloud at the beginning of the line.**

**node\_exporter:**

**enabled: true**

**# disable unused collectors**

**disable\_collectors:**

**- ipvs #high cardinality on kubelet**

**- btrfs**

**- infiniband**

**- xfs**

**- zfs**

**# exclude dynamic interfaces**

**netclass\_ignored\_devices: "^(veth.|cali.|[a-f0-9]{15})$"**

**netdev\_device\_exclude: "^(veth.|cali.|[a-f0-9]{15})$"**

**# disable tmpfs**

**filesystem\_fs\_types\_exclude: "^(autofs|binfmt\_misc|bpf|cgroup2?|configfs|debugfs|devpts|devtmpfs|tmpfs|fusectl|hugetlbfs|iso9660|mqueue|nsfs|overlay|proc|procfs|pstore|rpc\_pipefs|securityfs|selinuxfs|squashfs|sysfs|tracefs)$"**

**# drop extensive scrape statistics**

**relabel\_configs:**

**- replacement: 'monitoring-node'**

**target\_label: instance**

**metric\_relabel\_configs:**

**- action: drop**

**regex: node\_scrape\_collector\_.+**

**source\_labels:**

**- \_name\_**

**- action: keep**

**regex: node\_arp\_entries|node\_boot\_time\_seconds|node\_context\_switches\_total|node\_cpu\_seconds\_total|node\_disk\_io\_time\_seconds\_total|node\_disk\_io\_time\_weighted\_seconds\_total|node\_disk\_read\_bytes\_total|node\_disk\_read\_time\_seconds\_total|node\_disk\_reads\_completed\_total|node\_disk\_write\_time\_seconds\_total|node\_disk\_writes\_completed\_total|node\_disk\_written\_bytes\_total|node\_filefd\_allocated|node\_filefd\_maximum|node\_filesystem\_avail\_bytes|node\_filesystem\_device\_error|node\_filesystem\_files|node\_filesystem\_files\_free|node\_files>**

**source\_labels:**

**- \_name\_**

**logs:**

**configs:**

**- clients:**

**- basic\_auth:**

**password: glc\_eyJvIjoiMTA5MTQwNCIsIm4iOiJzdGFjay04OTY4MzAtaW50ZWdyYXRpb24tcHJhc2FkLTQwIiwiayI6Ik8xMjUzcjd1SDdiZTY2QTZUSmlhbzZZaSIsIm0iOnsiciI6InByb2QtYXAtc291dGgtMSJ9fQ==**

**username: 854611**

**url: https://logs-prod-028.grafana.net/loki/api/v1/push**

**name: integrations**

**positions:**

**filename: /tmp/positions.yaml**

**scrape\_configs:**

**# Add here any snippet that belongs to the logs.configs.scrape\_configs section.**

**# For a correct indentation, paste snippets copied from Grafana Cloud at the beginning of the line.**

**- job\_name: integrations/node\_exporter\_journal\_scrape**

**journal:**

**max\_age: 24h**

**labels:**

**instance: 'monitoring-node'**

**job: integrations/node\_exporter**

**relabel\_configs:**

**- source\_labels: ['\_journal\_systemd\_unit']**

**target\_label: 'unit'**

**- source\_labels: ['\_journal\_boot\_id']**

**target\_label: 'boot\_id'**

**- source\_labels: ['\_journal\_transport']**

**target\_label: 'transport'**

**- source\_labels: ['\_\_journal\_priority\_keyword']**

**target\_label: 'level'**

**- job\_name: integrations/node\_exporter\_direct\_scrape**

**static\_configs:**

**- targets:**

**- localhost**

**labels:**

**instance: 'monitoring-node'**

**\_path\_: /var/log/{syslog,messages,\*.log}**

**job: integrations/node\_exporter**

**metrics:**

**configs:**

**- name: integrations**

**remote\_write:**

**- basic\_auth:**

**password: glc\_eyJvIjoiMTA5MTQwNCIsIm4iOiJzdGFjay04OTY4MzAtaW50ZWdyYXRpb24tcHJhc2FkLTQwIiwiayI6Ik8xMjUzcjd1SDdiZTY2QTZUSmlhbzZZaSIsIm0iOnsiciI6InByb2QtYXAtc291dGgtMSJ9fQ==**

**username: 1507447**

**url: https://prometheus-prod-43-prod-ap-south-1.grafana.net/api/prom/push**

**scrape\_configs:**

**# Add here any snippet that belongs to the metrics.configs.scrape\_configs section.**

**# For a correct indentation, paste snippets copied from Grafana Cloud at the beginning of the line.**

**global:**

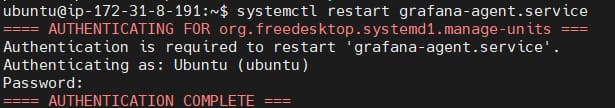
**scrape\_interval: 60s**

**wal\_directory: /tmp/grafana-agent-wal**

save the file and restart the grafana agent service

systemctl restart grafana-agent.service

Give the password to authenticate.

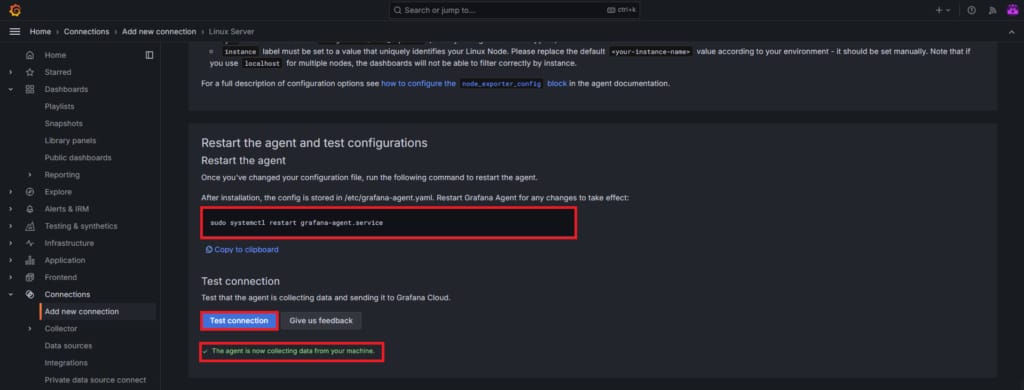


If you don’t know the password you can change it using following commanad.

sudo passwd ubuntu

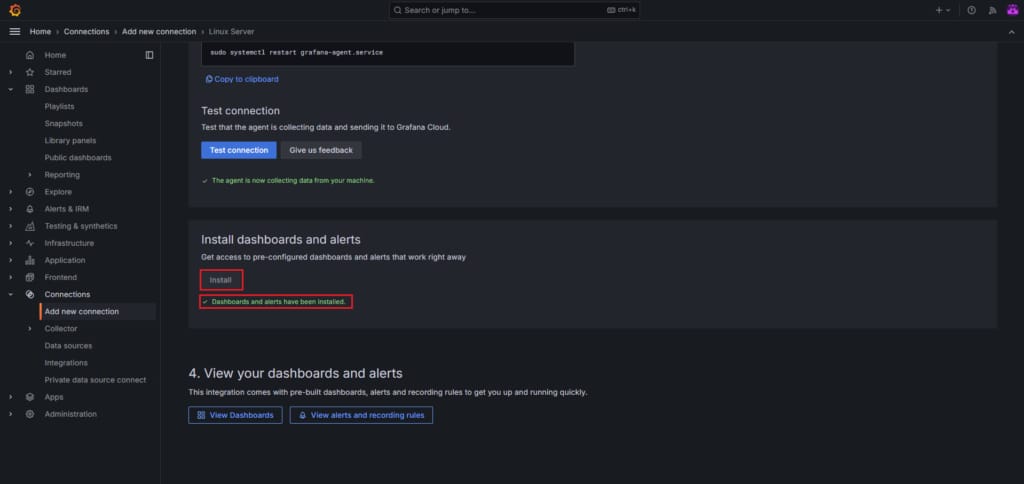
scroll down and click on Test connections to test that the agent is collecting data and sending it to Grafana Cloud.

You will get the message **“The agent is now collecting data from your machine.“**

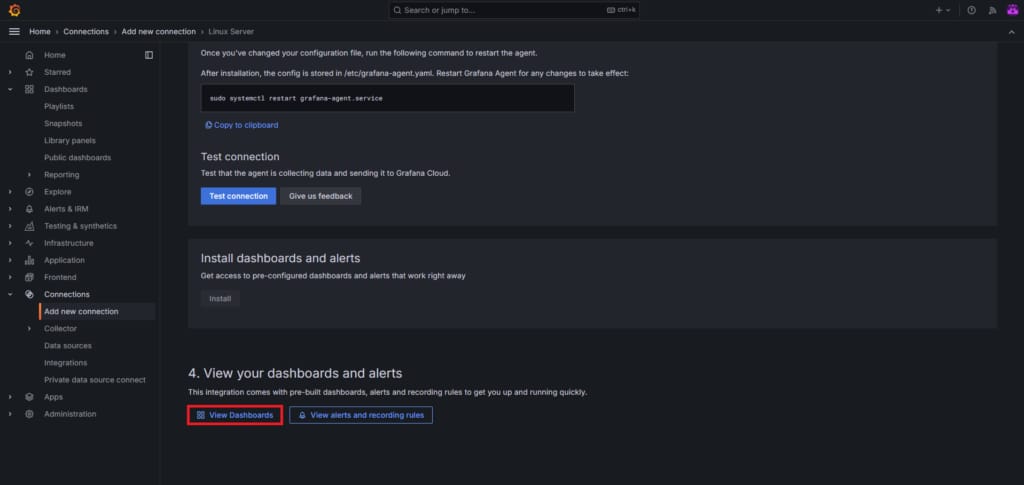
****

1. **Install Dashboards and alerts on Grafana Cloud**

Next click on Install to install the pre-configured dashboards and alerts.

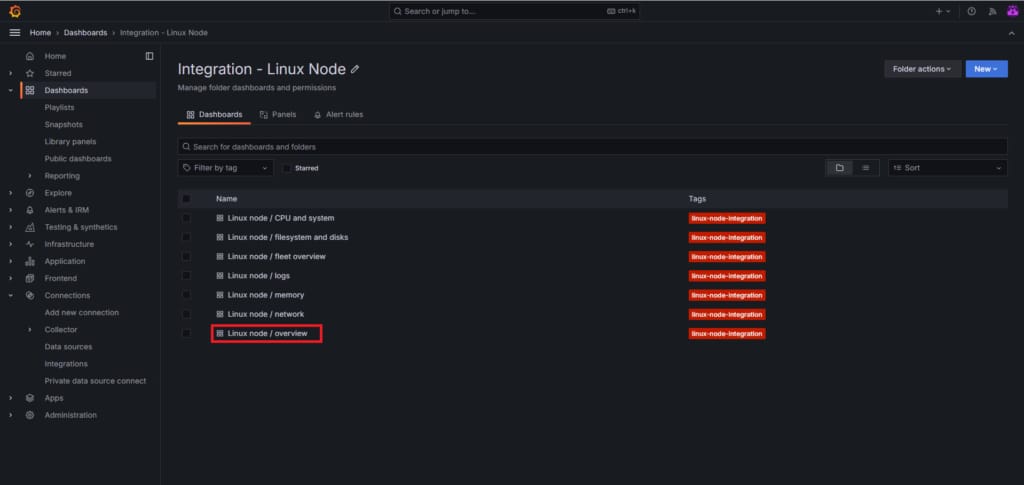


Now click on View Dashboards.



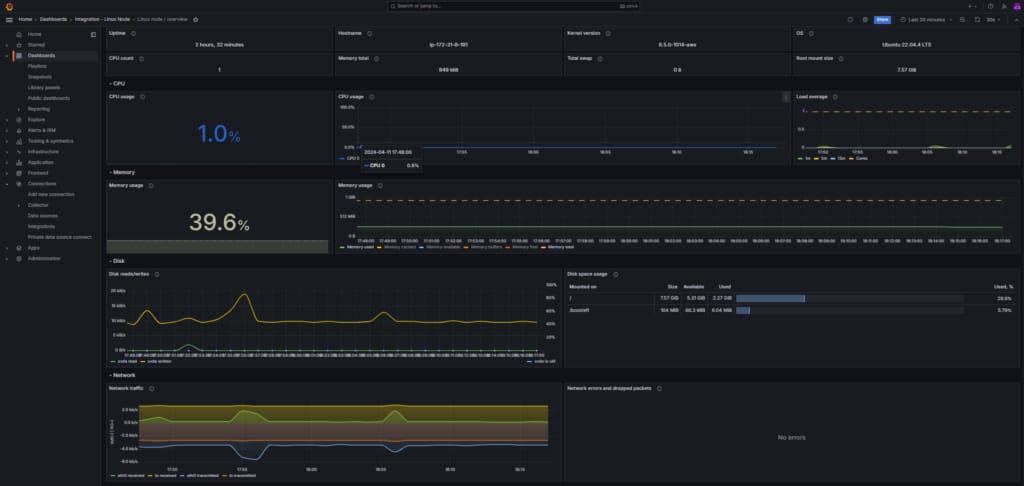
You will see the many pre-build dashboards installed. For now we will select

**Linux node/overviewDashboard**

****

The dashboard displays various graphs and metrics which are related to server performance.

Like CPU usage, memory usage, network traffice and many more.



**Conclusion:**

Integrating Linux servers with Grafana Cloud provides valuable insights into the performance and health of your infrastructure. You can seamlessly integrate Linux server with Grafana Cloud. By following the steps outlined in this guide, you can easily set up monitoring and visualization for your Linux servers and gain visibility into their operation. Whether you’re managing a single server or a large-scale deployment, Grafana Cloud offers the tools you need to monitor and optimize your infrastructure effectively.