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

Submitted:

From – Shruti Agnihotri

To – Abhinav Dwivedi Sir

**Table of contents**

1. Introduction
2. Problem Statement
3. Implementation
4. Short steps
5. Graph

1**.Introduction**

Grafana is an open- source platform used for monitoring , visualization , and analysis of data from various sources. It helps users turn complex data into clear, interactive dashboards.

**2.Problem Statement**

Monitoring system performance is critical in any production environment. This project aims to address the need to track CPU usage trends and identify performance bottlenecks on a Linux server using Grafana dashboards integrated with Prometheus and Node Exporter.

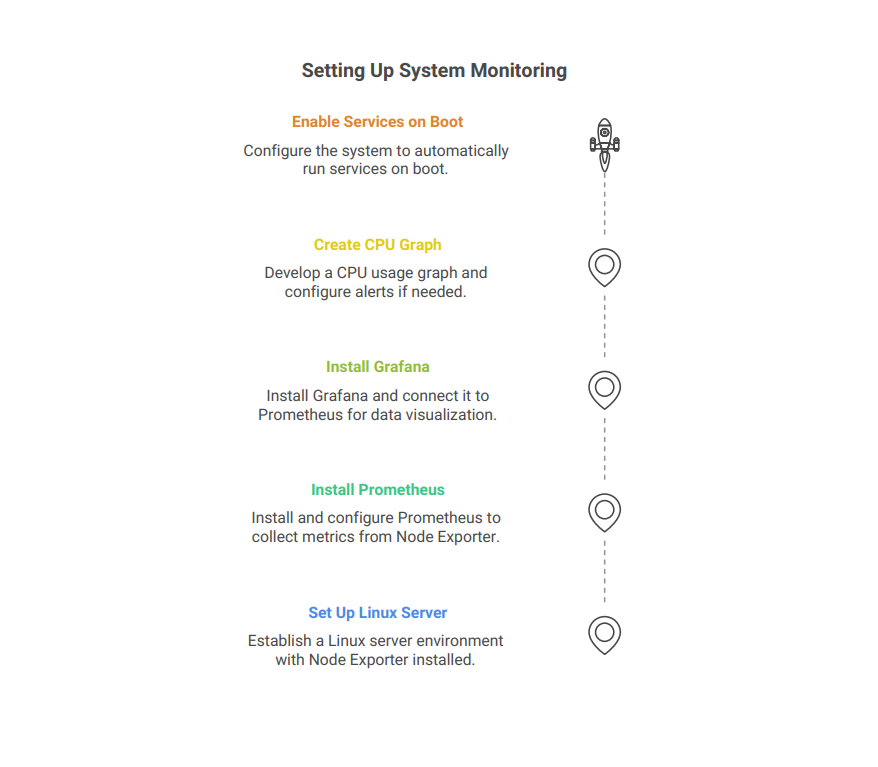
**3.Implementation**

Set up a Linux server with Node Exporter.Install Prometheus and configure it to read metrics from NodeExporter.

Install Grafana, connect it to Prometheus, and build a dashboard.

Create a CPU usage graph and optionally configure alerts.

Enable system to run services on boot**.**



**4.Short Steps**

**step 1**: Install Node Exporter on Linux Server

Node Exporter collects system metrics like CPU, memory, disk.

**Step 2** : Install Prometheus

Prometheus scrapes data from node exporter and sends it to Grafana.

**Step 3 :** Install Grafana

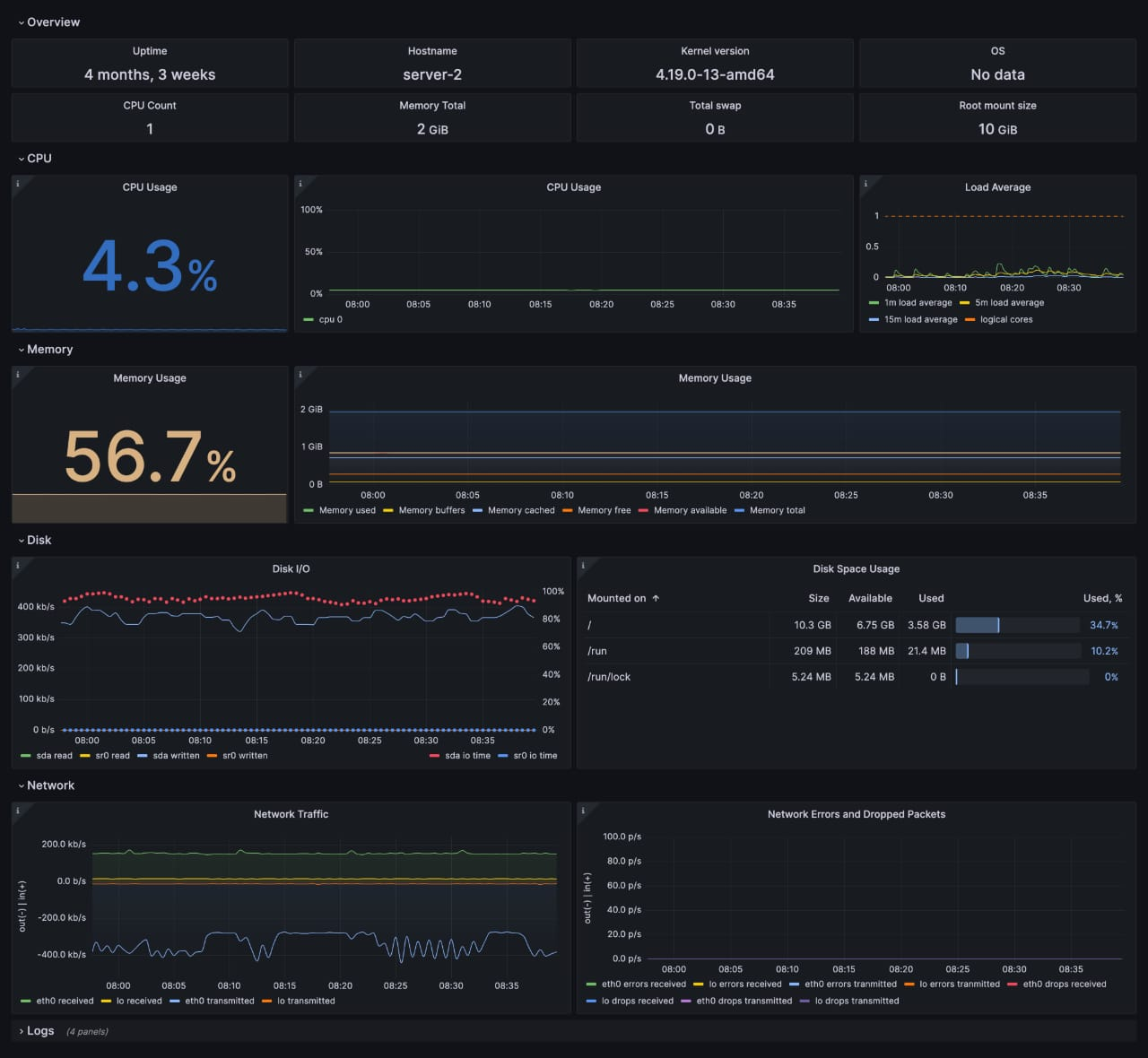
**Step 4 :**Acess Grafana dashboard

**Step 5 :** import CPU monitoring dashboard

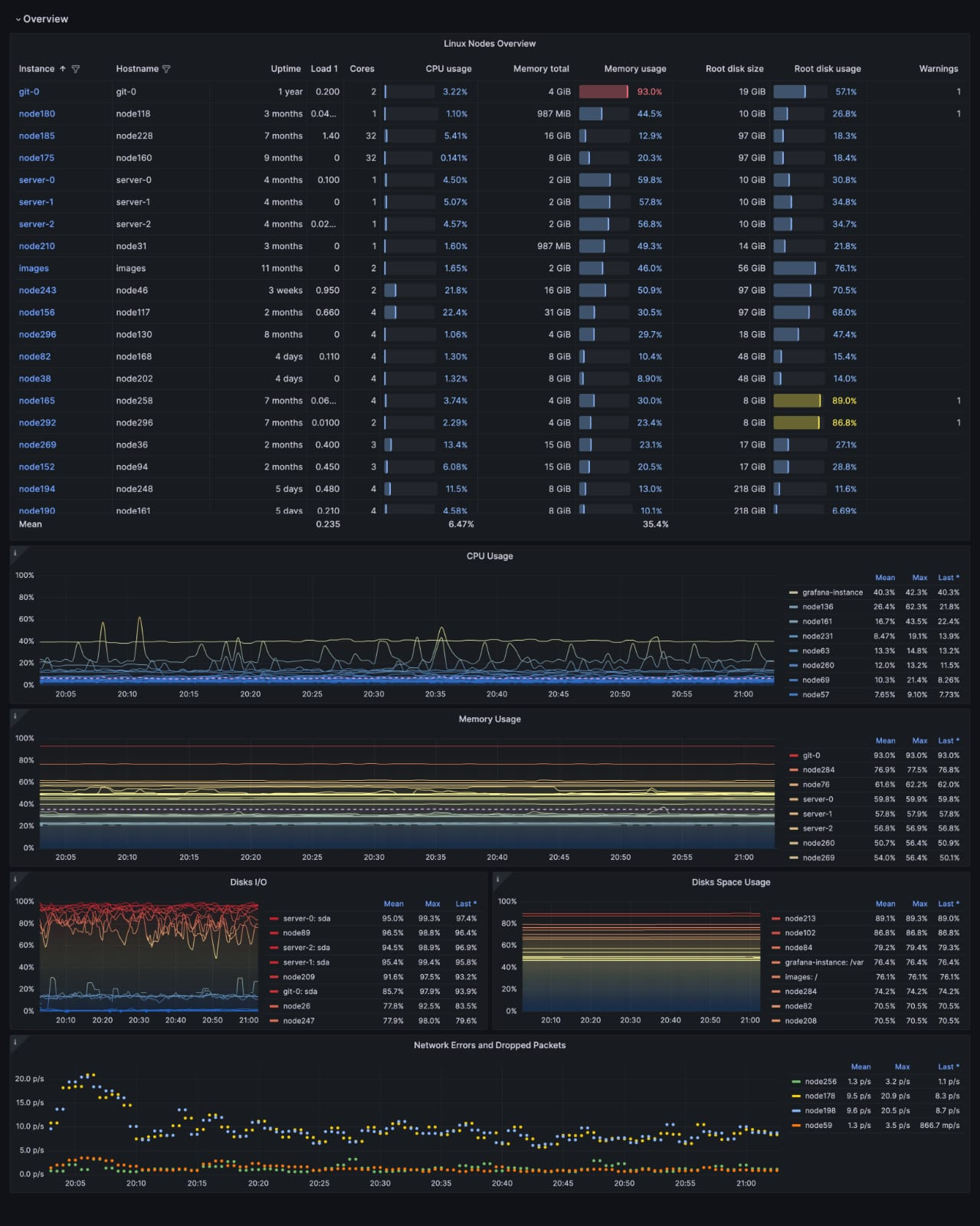
Dashboarads

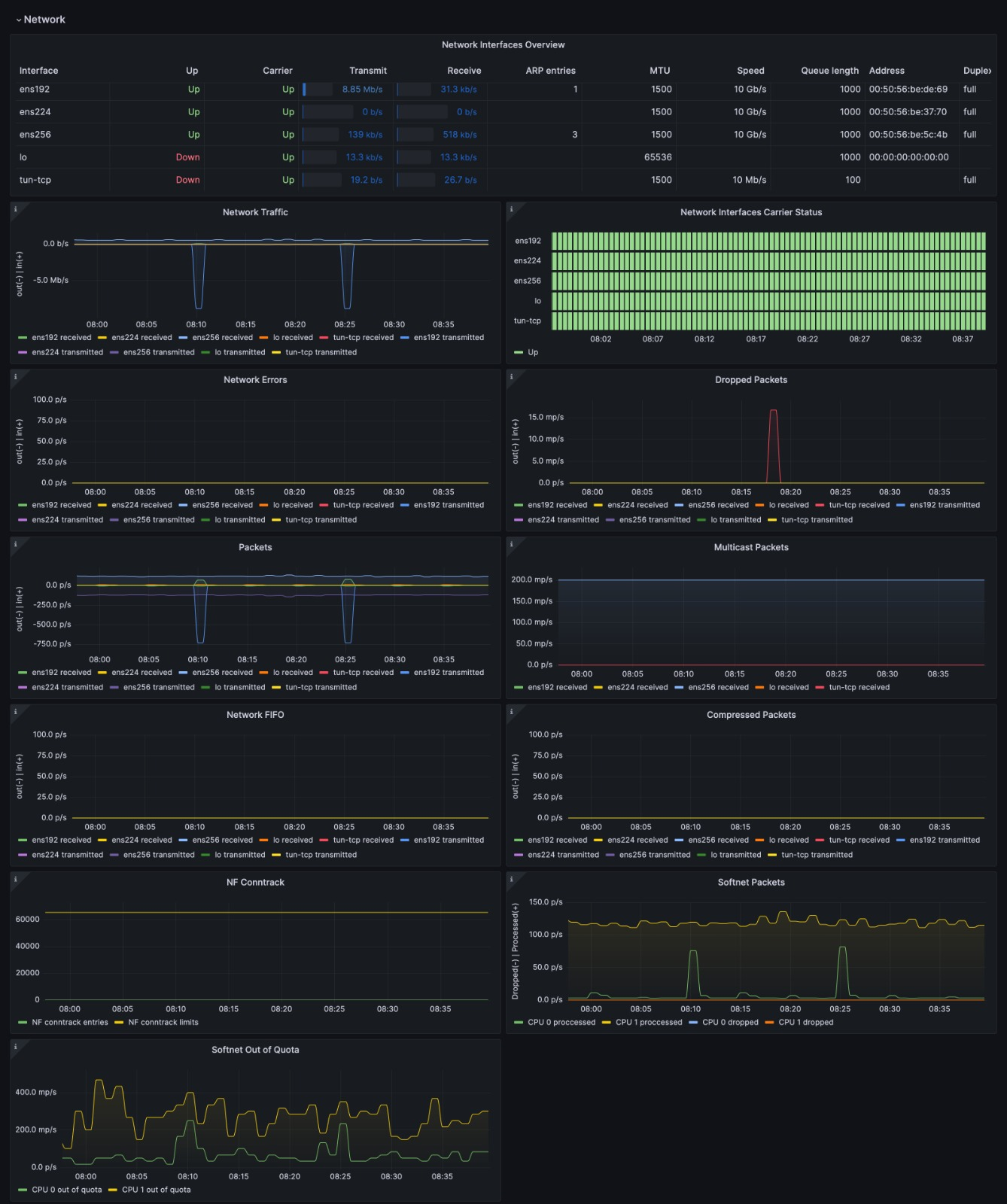
The Linux Server integration installs the following dashboards in your GrafanaCloud instance to help monitor your system.

* Linux node / CPU and system
* Linux node / filesystem and disks
* Linux node / fleet overview
* Linux node / logs
* Linux node / memory
* Linux node / network
* Linux node / overview

Node overview dashboard

Fleet overview dashboard



Drill down dashboards :Network interfaces

Metrics

The most important metrics provided by the Linux Server integration, which are used on the pre-built dashboards and Prometheus alerts, are as follows:

* 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\_filesystem\_readonly
* node\_filesystem\_size\_bytes
* node\_intr\_total
* node\_load1
* node\_load15
* node\_load5
* node\_md\_disks
* node\_md\_disks\_required
* node\_memory\_Active\_anon\_bytes
* node\_memory\_Active\_bytes
* node\_memory\_Active\_file\_bytes
* node\_memory\_AnonHugePages\_bytes
* node\_memory\_AnonPages\_bytes
* node\_memory\_Bounce\_bytes
* node\_memory\_Buffers\_bytes
* node\_memory\_Cached\_bytes
* node\_memory\_CommitLimit\_bytes
* node\_memory\_Committed\_AS\_bytes
* node\_memory\_DirectMap1G\_bytes
* node\_memory\_DirectMap2M\_bytes
* node\_memory\_DirectMap4k\_bytes
* node\_memory\_Dirty\_bytes
* node\_memory\_HugePages\_Free
* node\_memory\_HugePages\_Rsvd
* node\_memory\_HugePages\_Surp
* node\_memory\_HugePages\_Total
* node\_memory\_Hugepagesize\_bytes
* node\_memory\_Inactive\_anon\_bytes
* node\_memory\_Inactive\_bytes
* node\_memory\_Inactive\_file\_bytes
* node\_memory\_Mapped\_bytes
* node\_memory\_MemAvailable\_bytes
* node\_memory\_MemFree\_bytes
* node\_memory\_MemTotal\_bytes
* node\_memory\_SReclaimable\_bytes
* node\_memory\_SUnreclaim\_bytes
* node\_memory\_ShmemHugePages\_bytes
* node\_memory\_ShmemPmdMapped\_bytes
* node\_memory\_Shmem\_bytes
* node\_memory\_Slab\_bytes
* node\_memory\_SwapTotal\_bytes
* node\_memory\_VmallocChunk\_bytes
* node\_memory\_VmallocTotal\_bytes
* node\_memory\_VmallocUsed\_bytes
* node\_memory\_WritebackTmp\_bytes
* node\_memory\_Writeback\_bytes
* node\_netstat\_Icmp6\_InErrors
* node\_netstat\_Icmp6\_InMsgs
* node\_netstat\_Icmp6\_OutMsgs
* node\_netstat\_Icmp\_InErrors
* node\_netstat\_Icmp\_InMsgs
* node\_netstat\_Icmp\_OutMsgs
* node\_netstat\_IpExt\_InOctets
* node\_netstat\_IpExt\_OutOctets
* node\_netstat\_TcpExt\_ListenDrops
* node\_netstat\_TcpExt\_ListenOverflows
* node\_netstat\_TcpExt\_TCPSynRetrans
* node\_netstat\_Tcp\_InErrs
* node\_netstat\_Tcp\_InSegs
* node\_netstat\_Tcp\_OutRsts
* node\_netstat\_Tcp\_OutSegs
* node\_netstat\_Tcp\_RetransSegs
* node\_netstat\_Udp6\_InDatagrams
* node\_netstat\_Udp6\_InErrors
* node\_netstat\_Udp6\_NoPorts
* node\_netstat\_Udp6\_OutDatagrams
* node\_nrorsetstat\_Udp6\_RcvbufErrors
* node\_netstat\_Udp6\_SndbufEr
* node\_netstat\_UdpLite\_InErrors
* node\_netstat\_Udp\_InDatagrams
* node\_netstat\_Udp\_InErrors
* node\_netstat\_Udp\_NoPorts
* node\_netstat\_Udp6\_OutDatagrams
* node\_netstat\_Udp6\_RcvbufErrors
* node\_netstat\_Udp6\_SndbufErrors
* node\_netstat\_UdpLite\_InErrors
* node\_netstat\_Udp\_InDatagrams
* node\_netstat\_Udp\_InErrors
* node\_netstat\_Udp\_NoPorts
* node\_netstat\_Udp\_OutDatagrams
* node\_netstat\_Udp\_RcvbufErrors
* node\_netstat\_Udp\_SndbufErrors
* node\_network\_carrier
* node\_network\_info
* node\_network\_mtu\_bytes
* node\_network\_receive\_bytes\_total
* node\_network\_receive\_compressed\_total
* node\_network\_receive\_drop\_total
* node\_network\_receive\_errs\_total
* node\_network\_receive\_fifo\_total
* node\_network\_receive\_multicast\_total
* node\_network\_receive\_packets\_total
* node\_network\_speed\_bytes
* node\_network\_transmit\_bytes\_total
* node\_network\_transmit\_compressed\_total
* node\_network\_transmit\_drop\_total
* node\_network\_transmit\_errs\_total
* node\_network\_transmit\_fifo\_total
* node\_network\_transmit\_multicast\_total
* node\_network\_transmit\_packets\_total
* node\_network\_transmit\_queue\_length
* node\_network\_up
* node\_nf\_conntrack\_entries
* node\_nf\_conntrack\_entries\_limit
* node\_os\_info
* node\_procs\_running
* node\_sockstat\_FRAG6\_inuse
* node\_sockstat\_FRAG\_inuse
* node\_sockstat\_RAW6\_inuse
* node\_sockstat\_RAW\_inuse
* node\_sockstat\_TCP6\_inuse
* node\_sockstat\_TCP\_alloc
* node\_sockstat\_TCP\_inuse
* node\_sockstat\_TCP\_mem
* node\_sockstat\_TCP\_mem\_bytes
* node\_sockstat\_TCP\_orphan
* node\_sockstat\_TCP\_tw
* node\_sockstat\_UDP6\_inuse
* node\_sockstat\_UDPLITE6\_inuse
* node\_sockstat\_UDPLITE\_inuse
* node\_sockstat\_UDP\_inuse
* node\_sockstat\_UDP\_mem
* node\_sockstat\_UDP\_mem\_bytes
* node\_sockstat\_sockets\_used
* node\_softnet\_dropped\_total
* node\_softnet\_processed\_total
* node\_softnet\_times\_squeezed\_total
* node\_systemd\_service\_restart\_total
* node\_systemd\_unit\_state
* node\_textfile\_scrape\_error
* node\_time\_zone\_offset\_seconds
* node\_timex\_estimated\_error\_seconds
* node\_timex\_maxerror\_seconds
* node\_timex\_offset\_seconds
* node\_timex\_sync\_status
* node\_uname\_info
* node\_vmstat\_oom\_kill
* node\_vmstat\_pgfault
* node\_vmstat\_pgmajfault
* node\_vmstat\_pgpgin
* node\_vmstat\_pgpgout
* node\_vmstat\_pswpin
* node\_vmstat\_pswpout
* process\_max\_fds
* process\_open\_fds
* up

**Changelog**

# 1.6.2 - May 2025

\* Update Kubernetes Helm chart snippets.

# 1.6.1 - April 2025

\* Update log snippets:

\* Use new snippets to see journal logs on Node logs dashboard filters.

# 1.6.0 - March 2025

\* Update node mixin:

\* Add 'Load average' and 'Network usage' panels to fleet dashboard

\* Fix legends in disk panels on fleet dashboard

\* Fix CPU usage (per core) query.

# 1.5.2 - January 2025

\* Fix status panel metrics data source.

# 1.5.1 - January 2025

\* Fix status panel logs query.

# 1.5.0 - January 2025

\* Update node mixin, add new alerts:

\* NodeHasRebooted

\* NodeProcessesCountIsHigh

\* Fix logs dashboard not showing any logs if cluster label is missing.

# 1.4.2 - November 2024

\* Update Log dashboard job selector to always have a selected option.

# 1.4.1 - November 2024

\* Update status panel check queries.

# 1.4.0 - September 2024

\* Add asserts support.

# 1.3.0 - June 2024

\* Add new alert: NodeSystemdServiceCrashlooping

\* Fix links in the fleet overview table.

# 1.2.3 - December 2023

\* Accept `integrations/unix` for compatibility with default flow mode node\_exporter job name.

# 1.2.2 - December 2023

\* Fix issues with showing data on dashboards when `cluster` label has no value.

# 1.2.1 - December 2023

\* Fix queries for memoryBuffers memoryCached metrics

\* Update network traffic panels to show only interfaces that had traffic

\* Update network errors/drops panels to show only values greater than 0.

# 1.2.0 - November 2023

\* Dashboards prefixes are changed to 'Linux node/ '

\* Add new Loki based annotations:

\* Service failed

\* Critical system event

\* Session (ssh,console) opened/closed

\* Apply panel changes, some examples:

\* Use Sentence case in titles

\* Memory TS panel: Show only 'Memory total' and 'Memory used' by default

\* CPU usage TS panel: Use Blue-Yellow-Red color Schema

\* Add OS and group labels(job, cluster) as columns in Fleet overview table

\* NodeSystemSaturation alert severity is set to warning

\* Attach integration status panel to fleet and logs dashboards.

# 1.1.2 - August 2023

\* Add regex filter for logs datasource.

# 1.1.1 - July 2023

\* New Filter Metrics option for configuring the Grafana Agent, which saves on metrics cost by dropping any metric not used by this integration. Beware that anything custom built using metrics that are not on the snippet will stop working.

# 1.1.0 - June 2023

\* This update introduces generic logs dashboard 'Node Exporter / Node Logs'

\* Drop log panels 'Node Overview' dashboard.

# 1.0.1 - June 2023

\* This update includes the following, by updating to the latest mixin:

\* Panel description typos have been fixed

\* Incorrect data links in the "Node Fleet Overview" Dashboard now correctly include the dashboard selector.

# 1.0.0 - April 2023

\* This update introduces 3-tier view of linux nodes:

\* TOP: Fleet view: see group of your linux instances at once

\* Overview of the specific node: see specific node at a glance

\* Drill down: Set of dashboards for deep analysis using advanced metrics (Memory, CPU and System, Filesystem and Disk, Networking)

\* Links and data links are provided for better navigation between views

\* Update agent's filter config in docs, to reduce number of timeseries generated per node

\* Metrics filter instructions to exclude dynamic network devices, temp filesystems and extended scrape statistics

\* Remove USE dashboards

\* Convert all graphs to timeseries panels

\* Add information row

\* New alerts

\* Split alerts into two alert groups

\* Annotations for events: Reboot, OOMkill, and 'Kernel update'.

# 0.0.8 - October 2022

\* Update upstream node\_exporter mixin

\* Enable multicluster dashboards for use in kubernetes.

\* Add direct log file scrape to the agent snippets.

# 0.0.7 - September 2022

\* Remove source\_address from relabel\_configs.

# 0.0.6 - May 2022

\* Reverse fsSpaceAvailableCriticalThreshold and fsSpaceAvailableWarningThreshold

\* Update units for disk and networking panels.

# 0.0.5 - May 2022

\* Update 'Disk Space Usage' panel to table format.

# 0.0.4 - April 2022

\* Fixed alerts and recording rules by providing proper nodeSelector.

# 0.0.3 - February 2022

\* Added logs support from Loki datasource.

# 0.0.2 - October 2021

\* Update all rate queries to use `$\_\_rate\_interval`.

# 0.0.1 - June 2020

\* Initial release.

**Cost**

By connecting your Linux Server instance to Grafana Cloud, you might incur charges. To view information on the number of active series that your Grafana Cloud account uses for metrics included in each Cloud tier, see [Active series and dpm usage](https://grafana.com/docs/grafana-cloud/fundamentals/active-series-and-dpm/) and [Cloud tier pricing](https://grafana.com/products/cloud/pricing/).

**CONCLUSION**

Grafana is a powerful open-source data visualization and monitoring tool that enables users to analyze and track system performance in real time. With its wide range of data source integrations, customizable dashboards, and alerting features, Grafana empowers organizations to make informed decisions based on visual insights. It is especially effective in IT operations, DevOps, and business intelligence environments. Its flexibility, user-friendly interface, and strong community support make it a valuable asset for both beginners and advanced users seeking effective observability and monitoring solutions.

---