



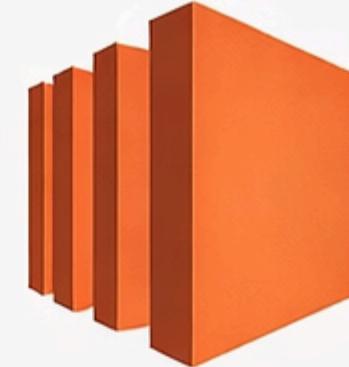
REAL-TIME MONITORING AND LOGGING ON AWS EC2 USING DOCKER, PROMETHEUS AND GRAFANA PLATFORM

PRIYANKA RAJAGOPAL
GitHub



PROJECT OVERVIEW

- The Project demonstrates a complete monitoring and logging solution for cloud-based applications using AWS EC2 and Docker containers.
- It integrates Prometheus for metrics collection, Grafana for visualization, Node Exporter for server-level metrics, Loki and Promtail for log management, and Alertmanager for proactive notifications.
- The goal is to provide real-time insights into system performance, track application logs, and set up automated alerts for potential issues, ensuring reliability and operational efficiency.



AWS EC2
Amazon Linux



Docker Containers



Prometheus
Metrics Collection & Monitoring



Node Exporter
System Metrics



Grafana



Loki
Log Aggregation



Promtail
Log Collection



Alertmanager
Alerting & Notifications

EC2 > Instances > Launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name
Ec2-Monitoring [Add additional tags](#)

Application and OS Images (Amazon Machine Image) Info

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Search our full catalog including 1000s of application and OS images

Recents [Quick Start](#)

Amazon Linux 
macOS 
Ubuntu 
Windows 
Red Hat 
SUSE Linux 

Del 

Amazon Machine Image (AMI)

Summary

Number of instances [Info](#)
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.9.2...[read more](#)
ami-068c0051b15cdb816

Virtual server type (instance type)
t3.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

[Cancel](#) [Launch instance](#) [Preview code](#)

Launched an instance with "Amazon Linux".

Search [ALT+S] ASK Amazon Q

EC2 Instances Launch an instance

Instance type Info | Get advice

Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

All genera Compare insta

Additional costs apply for AMIs with pre-installed software

Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before launching the instance.

Key pair name - required

Monitor-key

Create new key pair

Cancel Create key pair

Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

Monitor-key

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA RSA encrypted private and public key pair

ED25519 ED25519 encrypted private and public key pair

Private key file format

.pem For use with OpenSSH

.ppk For use with PuTTY

⚠️ When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more ↗](#)

Used t2 micro instance type with the respective key pair.



Search

[Alt+S] Ask Amazon Q



United States (N. Virginia) ▾

priyanka-monitor

☰ EC2 > Security Groups > Create security group

ⓘ 🔍 ⌂

vpc-0da18053b3b36a508 (VPC-Monitoring)

Inbound rules Info

Type	Info	Protocol	Info	Port range	Info	Source	Info	Description - optional	Info	
SSH	▼	TCP		22		Cust...	▼	122.178.163.107/?	X	<button>Delete</button>
								122.178.163.107/32	X	
HTTP	▼	TCP		80		Any...	▼	0.0.0.0/0	X	<button>Delete</button>
Custom TCP	▼	TCP		9090		Any...	▼	0.0.0.0/0	X	Prometheus <button>Delete</button>
Custom TCP	▼	TCP		3000		Any...	▼	0.0.0.0/0	X	Grafana <button>Delete</button>
Custom TCP	▼	TCP		9100		Any...	▼	0.0.0.0/0	X	Node Exporter <button>Delete</button>
Custom TCP	▼	TCP		3100		Any...	▼	0.0.0.0/0	X	Loki <button>Delete</button>

Security group -Inbound rules are created for Prometheus,Grafana,Node exporter and Loki .Outbound rules are given as All traffic.

Search [ALT+S] ASK Amazon Q United States (N. Virginia) priyanka-monitor

EC2 > Instances > Launch an instance

▼ Network settings [Info](#)

VPC - required [Info](#)

vpc-0da18053b3b36a508 (VPC-Monitoring)
10.0.0.0/16

Subnet [Info](#)

subnet-0ae06ce47e5361d55 Pub-1a
VPC: vpc-0da18053b3b36a508 Owner: 946769779757
Availability Zone: us-east-1a (use1-az1) Zone type: Availability Zone
IP addresses available: 251 CIDR: 10.0.1.0/24

Create new subnet ↗

Auto-assign public IP [Info](#)

Enable

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Common security groups [Info](#)

Select security groups

SG-Monitoring sg-0995b5b573240e849 X
VPC: vpc-0da18053b3b36a508

Compare security group rules ↗

Security groups that you add or remove here will be added to or removed from all your network interfaces.

► Advanced network configuration

▼ Summary

Number of instances [Info](#)

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.9.2...[read more](#)
ami-068c0051b15cdb816

Virtual server type (instance type)

t2.micro

Firewall (security group)

SG-Monitoring

Storage (volumes)

1 volume(s) - 8 GiB

Cancel [Launch instance](#) [Preview code](#)

Attached the security group with instance creating and “Auto assign public IP” is enabled.

Search ALT+S ASK Amazon ?

United States (N. Virginia) priyanka-monitor

Instances

Instances (1/1) Info

Connect Instance state Actions Launch instances

Find Instance by attribute or tag (case-sensitive)

All states

View ▾

Name Instance ID Instance state Instance type Status check Alarm status Availability

Ec2-Monitoring i-0bc7d0ddb457cb033 Running t2.micro 2/2 checks passed View alarms us-east-1a

i-0bc7d0ddb457cb033 (Ec2-Monitoring)

Details Status and alarms Monitoring Security Networking Storage Tags

Instance summary

Instance ID: i-0bc7d0ddb457cb033

IPv6 address:

Hostname type: IP name: ip-10-0-1-232.ec2.internal

Answer private resource DNS name:

Public IPv4 address copied: 44.204.93.21 | open address ↗

Instance state: Running

Private IP DNS name (IPv4 only): ip-10-0-1-232.ec2.internal

Instance type: t2.micro

Private IPv4 addresses: 10.0.1.232

Public DNS:

Elastic IP addresses:

Launched instance with the Putty Software using the Public IP address.

```
[root@ip-10-0-1-232 ~]# sudo yum update -y
Amazon Linux 2023 Kernel Livepatch repository
Dependencies resolved.
Nothing to do.
Complete!
```

```
[root@ip-10-0-1-232 ~]# sudo yum install docker -y
Last metadata expiration check: 0:05:56 ago on Fri Dec 26 03:19:13 2025.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
<hr/>				
Installing:				
docker	x86_64	25.0.13-1.amzn2023.0.2	amazonlinux	46 M
<hr/>				
Installing dependencies:				
container-selinux	noarch	4:2.242.0-1.amzn2023	amazonlinux	58 k
containerd	x86_64	2.1.5-1.amzn2023.0.1	amazonlinux	23 M
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	401 k
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	183 k
libcgroup	x86_64	3.0-1.amzn2023.0.1	amazonlinux	75 k
libnetfilter_conntrack	x86_64	1.0.8-2.amzn2023.0.2	amazonlinux	58 k
libnfnetlink	x86_64	1.0.1-19.amzn2023.0.2	amazonlinux	30 k
libnftnl	x86_64	1.2.2-2.amzn2023.0.2	amazonlinux	84 k
pigz	x86_64	2.5-1.amzn2023.0.3	amazonlinux	83 k
runc	x86_64	1.3.3-2.amzn2023.0.1	amazonlinux	3.9 M

Transaction Summary

```
Install 11 Packages
```

```
Total download size: 74 M
Installed size: 280 M
```

Installed dependencies and docker.

```
[root@ip-10-0-1-232 monitoring]# sudo systemctl enable docker
[root@ip-10-0-1-232 monitoring]# sudo usermod -aG docker $USER
[root@ip-10-0-1-232 monitoring]# docker version
Client:
  Version:          25.0.13
  API version:     1.44
  Go version:      go1.24.9
  Git commit:       0bab007
  Built:            Mon Nov  3 00:00:00 2025
  OS/Arch:          linux/amd64
  Context:          default

Server:
  Engine:
    Version:          25.0.13
    API version:     1.44 (minimum version 1.24)
    Go version:      go1.24.9
    Git commit:       165516e
    Built:            Mon Nov  3 00:00:00 2025
    OS/Arch:          linux/amd64
    Experimental:    false
  containerd:
    Version:          2.1.5
    GitCommit:        fcd43222d6b07379a4be9786bda52438f0dd16a1
  runc:
    Version:          1.3.3
    GitCommit:        d842d7719497cc3b774fd71620278ac9e17710e0
  docker-init:
    Version:          0.19.0
    GitCommit:        de40ad0
[root@ip-10-0-1-232 monitoring]#
```

Started docker and Version of docker is confirmed.

```
[root@ip-10-0-1-232 monitoring]# docker run -d --name node-exporter --network monitoring-net -p 9100:9100 prom/node-exporter
e3ddee30179a374488ff55df27b5a97aa1453e4972f2bleffa60ed5771ed0d75
[root@ip-10-0-1-232 monitoring]#
```

```
[root@ip-10-0-1-232 monitoring]# curl localhost:9100/metrics
# HELP go_gc_duration_seconds A summary of the wall-time pause (stop-the-world) duration in garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 0
go_gc_duration_seconds{quantile="0.25"} 0
go_gc_duration_seconds{quantile="0.5"} 0
go_gc_duration_seconds{quantile="0.75"} 0
go_gc_duration_seconds{quantile="1"} 0
go_gc_duration_seconds_sum 0
go_gc_duration_seconds_count 0
# HELP go_gc_gogc_percent Heap size target percentage configured by the user, otherwise 100. This value is set by the GOGC environment variable, and the runtime/debug.SetGCPercen function. Sourced from /gc/gogc:percent.
# TYPE go_gc_gogc_percent gauge
go_gc_gogc_percent 100
# HELP go_gc_gomemlimit_bytes Go runtime memory limit configured by the user, otherwise math.MaxInt64. This value is set by the GOMEMLIMIT environment variable, and the runtime/debug.SetMemoryLimit function. Sourced from /gc/gomemlimit:bytes.
# TYPE go_gc_gomemlimit_bytes gauge
go_gc_gomemlimit_bytes 9.223372036854776e+18
# HELP go_goroutines Number of goroutines that currently exist.
```

Started Node Exporter using Docker and validated metric exposure via the /metrics endpoint.

```
root@ip-10-0-1-232:~/monitoring
GNU nano 8.3
prometheus/prometheus.yml
```

global:
 scrape_interval: 15s

scrape_configs:
 - job_name: prometheus
 static_configs:
 - targets: ["localhost:9090"]

 - job_name: node
 static_configs:
 - targets: ["localhost:9100"]

```
root@ip-10-0-1-232:~/monitoring
[root@ip-10-0-1-232 monitoring]# docker run -d \
--name prometheus \
-p 9090:9090 \
-v $(pwd)/prometheus:/etc/prometheus \
prom/prometheus
Unable to find image 'prom/prometheus:latest' locally
latest: Pulling from prom/prometheus
9d85dc8d0609: Already exists
d0f7326b7716: Already exists
c5e95088860d: Pull complete
7408017f0d80: Pull complete
48a0f696008d: Pull complete
2430d5ddfd0e: Pull complete
b8d9ef68618b: Pull complete
b351d2c02e9d: Pull complete
08a0c52f4aa7: Pull complete
fd581fe2de26: Pull complete
Digest: sha256:2b6f734e372c1b4717008f7d0a0152316aedd4d13ae17ef1e3268dbfaf68041b
Status: Downloaded newer image for prom/prometheus:latest
daf4b3e7ac6ce1fa82e7ac4bb4f42da384e7075bd7b2a6adcf70041f9444bf65
```

Configured Prometheus using a custom `prometheus.yml` file and started the service using Docker.

```
root@ip-10-0-1-232:~/alertmanager
GNU nano 8.3                               /root/monitoring/alertmanager/alertmanager.yml
global:
  resolve_timeout: 5m

route:
  group_by: ['alertname']
  receiver: 'dummy'

receivers:
- name: 'dummy'

```

```
root@ip-10-0-1-232:~/monitoring
[root@ip-10-0-1-232 monitoring]# nano alertmanager/alertmanager.yml
[root@ip-10-0-1-232 monitoring]# docker run -d \
--name alertmanager \
-p 9093:9093 \
-v $(pwd)/alertmanager:/etc/alertmanager \
prom/alertmanager
Unable to find image 'prom/alertmanager:latest' locally
latest: Pulling from prom/alertmanager
9d85dc8d0609: Pull complete
d0f7326b7716: Pull complete
1e6a3a57898c: Pull complete
315eb988c8ec: Pull complete
e102c75aa13f: Pull complete
389784bee712: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:abb750ac7b63116761c16dd481ae92496fbe04721686c0920f0fa4d0728cd4a6
Status: Downloaded newer image for prom/alertmanager:latest
2bf6b5e20b845ee19541e4f076f6d9a12d27a4fa8561265db9b193c90b1e5125
[root@ip-10-0-1-232 monitoring]#
```

**Configured Alertmanager with alert routing
rules and started it using Docker.**

```
[root@ip-10-0-1-232:~/monitoring]
```

```
[root@ip-10-0-1-232 monitoring]# docker run -d \
--name grafana \
-p 3000:3000 \
grafana/grafana
```

```
Unable to find image 'grafana/grafana:latest' locally
latest: Pulling from grafana/grafana
014e56e61396: Pull complete
dafcd853cdea: Pull complete
fe0fb9ad3b64: Pull complete
f50da64eedfd: Pull complete
4a965f51fa32: Pull complete
835a6d0e4902: Pull complete
a2a0d06388d6: Pull complete
a7d3fdc2e5af: Pull complete
c8585378023b: Pull complete
d2772eb63bbd: Pull complete
Digest: sha256:2175aaa91c96733d86d31cf270d5310b278654b03f5718c59de12a865380a31f
Status: Downloaded newer image for grafana/grafana:latest
468bcde705232234276266d5baeb8a354301aa471649e93e3f3d2ad0a93b7906
[root@ip-10-0-1-232 monitoring]# █
```

Started Grafana using Docker

```
GNU nano 8.3                               /root/monitoring/loki/loki-config.yml      Modified
auth_enabled: false

server:
  http_listen_port: 3100

ingester:
  lifecycler:
    address: 127.0.0.1
    ring:
      kvstore:
        store: inmemory
        replication_factor: 1
    chunk_idle_period: 5m
    max_chunk_age: 1h
    chunk_target_size: 1048576

schema_config:
  configs:
    - from: 2020-10-24
      store: boltdb
      object_store: filesystem
      schema: v11
      index:
        prefix: index_
        period: 168h

storage_config:
```

```
[root@ip-10-0-1-232 monitoring]# docker run -d \
--name loki \
-p 3100:3100 \
-v $(pwd)/loki/loki-config.yml:/etc/loki/loki-config.yml \
-v $(pwd)/loki/wal:/wal \
-v $(pwd)/loki/chunks:/loki/chunks \
-v $(pwd)/loki/index:/loki/index \
grafana/loki:2.9.0 \
-config.file=/etc/loki/loki-config.yml
046a03b5f418a02494268dbfdb4ed446417ee576260e2cb339a865c91631cb65
[root@ip-10-0-1-232 monitoring]#
```

Loki was configured with storage and ingestion settings and then started it using docker

```
root@ip-10-0-1-232:~/monitoring/loki-data          /root/promtail/promtail-config.yml      Modified
GNU nano 8.3
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
  - url: http://loki:3100/loki/api/v1/push

scrape_configs:
  - job_name: system
    static_configs:
      - targets:
          - localhost
        labels:
          job: varlogs
          __path__: /var/log/*.log
```

```
[root@ip-10-0-1-232 monitoring]# docker run -d \
--name promtail \
-v /var/log:/var/log \
-v $(pwd)/promtail:/etc/promtail \
grafana/promtail:2.9.0 \
--config.file=/etc/promtail/promtail-config.yml
Unable to find image 'grafana/promtail:2.9.0' locally
2.9.0: Pulling from grafana/promtail
7d97e254a046: Pull complete
39932bcaalcc: Pull complete
4059d897dde4: Pull complete
325d59b59788: Pull complete
9e9c5b842176: Pull complete
01befd6b403e: Pull complete
Digest: sha256:c2c423196c75a2c9c26f6fe0ba7200c3167334b14975747f5dcff678bd1a32e9
Status: Downloaded newer image for grafana/promtail:2.9.0
75d39e73bd4de8fb5b20e047cad1f3cb2ad7a04e4f847f431dac7f2130704618
[root@ip-10-0-1-232 monitoring]#
```

Configured Promtail to ship logs to Loki and started it using Docker.

```
[root@ip-10-0-1-232 ~]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
NAMES
a35c8fc949f1 grafana/promtail:2.9.0 "/usr/bin/promtail -..." 36 minutes ago Up 36 minutes
promtail
26571702f4d5 grafana/loki:2.9.0 "/usr/bin/loki -conf..." 37 minutes ago Up 37 minutes 0.0.0.0:3100->3100/tcp,
:::3100->3100/tcp loki
e3ddee30179a prom/node-exporter "/bin/node_exporter" 49 minutes ago Up 49 minutes 0.0.0.0:9100->9100/tcp,
:::9100->9100/tcp node-exporter
3126560b4bdd prom/alertmanager "/bin/alertmanager -..." 51 minutes ago Up 51 minutes 0.0.0.0:9093->9093/tcp,
:::9093->9093/tcp alertmanager
f476c40cd968 grafana/grafana "/run.sh" 52 minutes ago Up 52 minutes 0.0.0.0:3000->3000/tcp,
:::3000->3000/tcp grafana
dd783c737a73 prom/prometheus "/bin/prometheus --c..." 52 minutes ago Up 52 minutes 0.0.0.0:9090->9090/tcp,
:::9090->9090/tcp prometheus
[root@ip-10-0-1-232 ~]#
```

```
[root@ip-10-0-1-232 ~]# docker ps | grep promtail
a35c8fc949f1 grafana/promtail:2.9.0 "/usr/bin/promtail -..." 38 minutes ago Up 38 minutes
promtail
[root@ip-10-0-1-232 ~]# docker logs promtail
level=info ts=2025-12-26T05:13:09.431022538Z caller=promtail.go:133 msg="Reloading configuration file" md5sum=23e8de9c5290
afffb26ff0c488d3d970
level=info ts=2025-12-26T05:13:09.442036369Z caller=server.go:322 http=[::]:9080 grpc=[::]:33439 msg="server listening on
addresses"
level=info ts=2025-12-26T05:13:09.455504615Z caller=main.go:174 msg="Starting Promtail" version="(version=2.9.0, branch=HEAD,
revision=2feb64f69)"
level=warn ts=2025-12-26T05:13:09.461299457Z caller=promtail.go:263 msg="enable watchConfig"
level=info ts=2025-12-26T05:13:14.457514508Z caller=filetargetmanager.go:361 msg="Adding target" key="/var/log/*.log:{job=
\"varlogs\"}"
level=info ts=2025-12-26T05:13:14.460491337Z caller=filetarget.go:313 msg="watching new directory" directory=/var/log
ts=2025-12-26T05:13:14.460782408Z caller=log.go:168 level=info msg="Seeked /var/log/cloud-init-output.log - &{Offset:0 Whe
nce:0}"
level=info ts=2025-12-26T05:13:14.461418867Z caller=tailer.go:145 component=tailer msg="tail routine: started" path=/var/l
og/cloud-init-output.log
```

**Verified container status using `docker ps` and
confirmed logs were successfully shipped to Loki.**

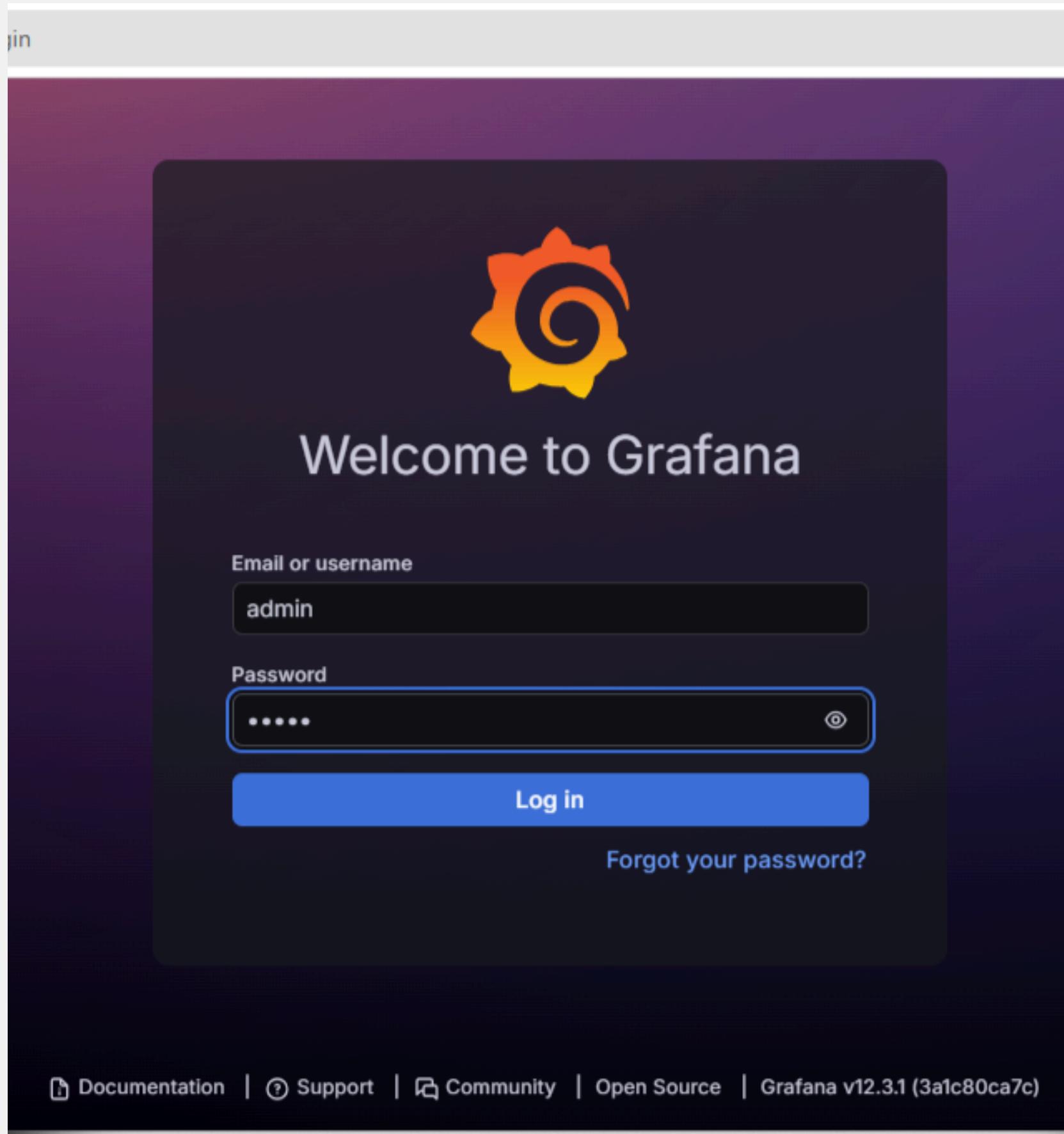
The screenshot shows the Prometheus web interface at the URL 44.204.93.21:9090/targets. The top navigation bar includes links for 'Query', 'Alerts', and 'Status > Target health'. The main content area displays two targets: 'node-exporter' and 'prometheus'. Each target has a table with columns for 'Endpoint', 'Labels', 'Last scrape', and 'State'. The 'node-exporter' target has one endpoint at 'http://node-exporter:9100/metrics' with labels 'instance="node-exporter:9100"' and 'job="node-exporter"'. It was scraped 5.224s ago and is in an 'UP' state. The 'prometheus' target has one endpoint at 'http://localhost:9090/metrics' with labels 'instance="localhost:9090"' and 'job="prometheus"'. It was scraped 8.661s ago and is in an 'UP' state. Both targets are marked as '1 / 1 up'.

Endpoint	Labels	Last scrape	State
http://node-exporter:9100/metrics	instance="node-exporter:9100" job="node-exporter"	5.224s ago	15ms UP

Endpoint	Labels	Last scrape	State
http://localhost:9090/metrics	instance="localhost:9090" job="prometheus"	8.661s ago	5ms UP

**Verified target status of Prometheus and confirmed
Node Exporter was successfully scraped**

jin



The image shows the Grafana login interface. It features a dark purple header with the word "jin" in white. Below the header is a large orange and yellow gear logo. The main title "Welcome to Grafana" is displayed in white. There are two input fields: "Email or username" containing "admin" and "Password" containing four asterisks. A blue "Log in" button is centered below the inputs. To the right of the inputs is a link "Forgot your password?". At the bottom of the page, there is a footer with links: "Documentation", "Support", "Community", "Open Source", and "Grafana v12.3.1 (3a1c80ca7c)".

jin

Welcome to Grafana

Email or username

admin

Password

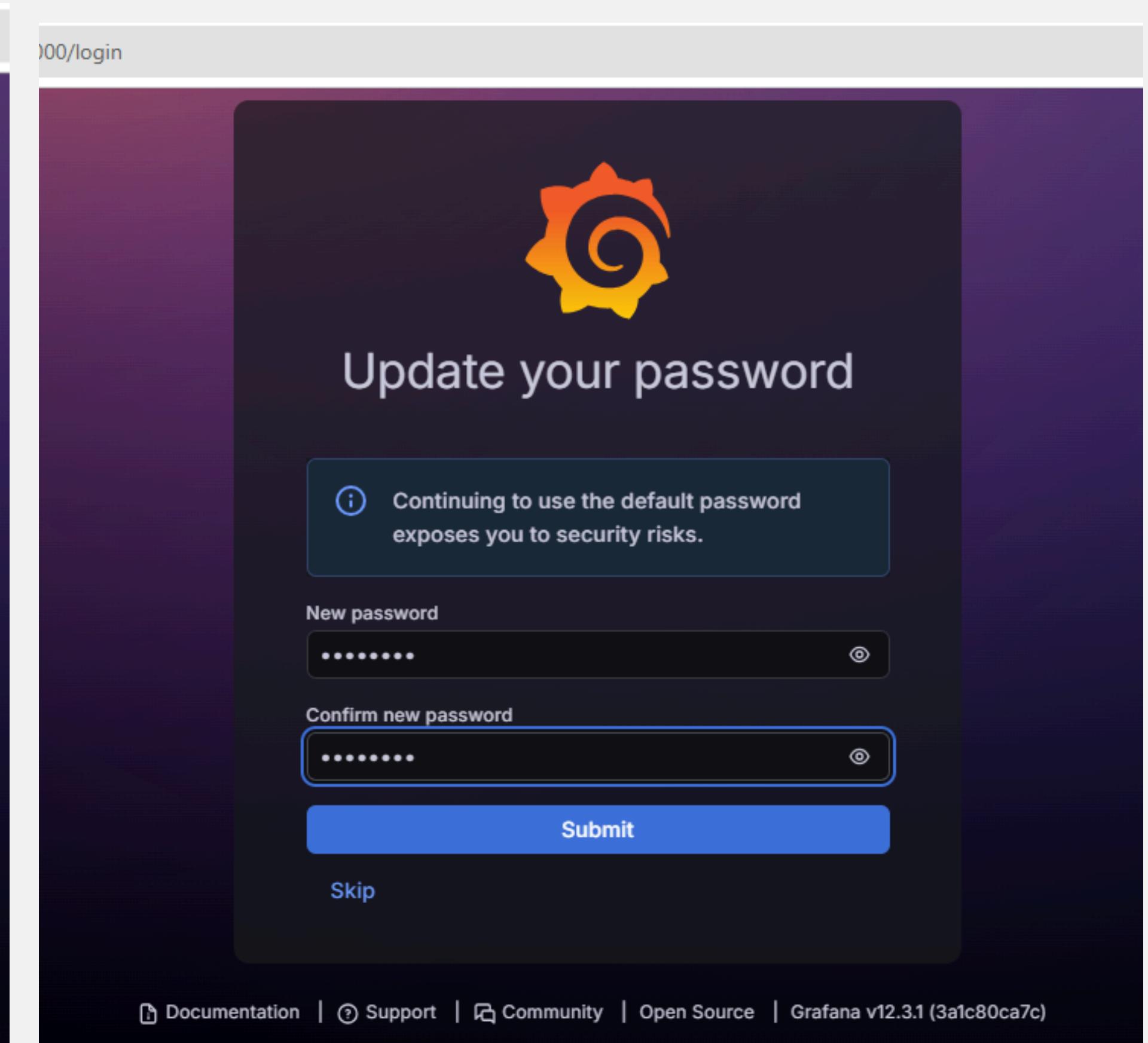
.....

Log in

Forgot your password?

Documentation | Support | Community | Open Source | Grafana v12.3.1 (3a1c80ca7c)

jin



The image shows the Grafana password update interface. The header "jin" is visible at the top left. The main title "Update your password" is centered. Below the title is a warning message in a teal box: "Continuing to use the default password exposes you to security risks." There are two input fields: "New password" and "Confirm new password", both containing five asterisks. A blue "Submit" button is located below the password fields. To the left of the "Submit" button is a "Skip" link. At the bottom of the page, there is a footer with links: "Documentation", "Support", "Community", "Open Source", and "Grafana v12.3.1 (3a1c80ca7c)".

jin

00/login

Update your password

Continuing to use the default password exposes you to security risks.

New password

.....

Confirm new password

.....

Submit

Skip

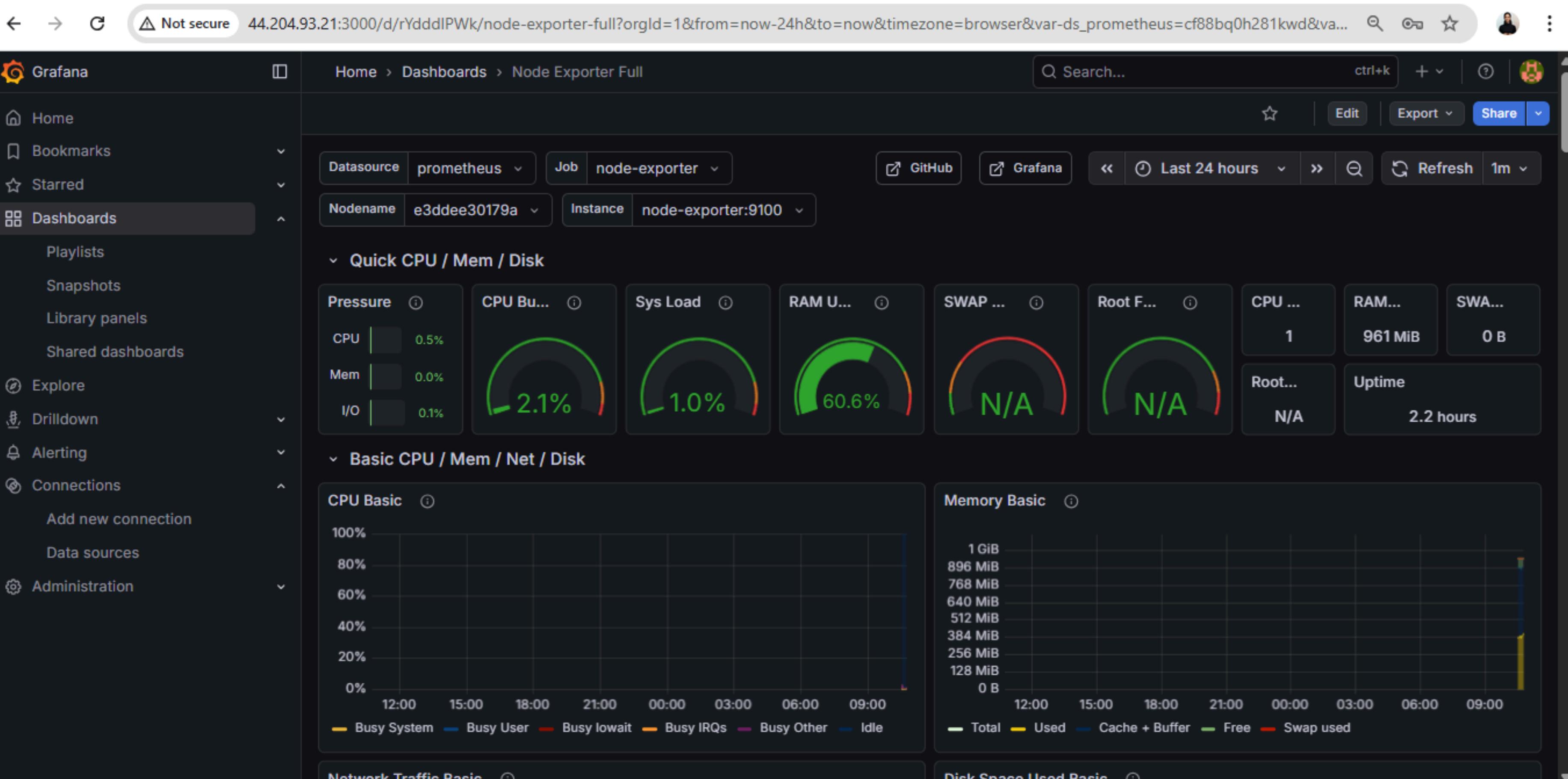
Documentation | Support | Community | Open Source | Grafana v12.3.1 (3a1c80ca7c)

Grafana service is running and accessible via web interface."

The screenshot shows the 'Add data source' page in Grafana. The left sidebar is dark-themed and includes links for Home, Bookmarks, Starred, Dashboards, Explore, Drilldown, Alerting, Connections, and Administration. The 'Data sources' link is highlighted with an orange border. The main content area has a light background and features a search bar at the top right. Below it, a section titled 'Time series databases' lists four options: Prometheus, Graphite, InfluxDB, and OpenTSDB. Each option has a small icon, a name, a description, and a 'Core' button. The 'Prometheus' option is selected, indicated by a blue border around its card.

The screenshot shows the 'Edit data source' page for the Prometheus connection. The left sidebar is identical to the previous screenshot. The main content area shows configuration settings for the Prometheus data source. It includes sections for 'Custom query parameters' (with a placeholder 'Example: max_source_resolution=5m&tir'), 'HTTP method' (set to 'POST'), 'Series limit' (set to '40000'), and 'Use series endpoint' (disabled). A 'Disable recording rules (beta)' toggle switch is also present. On the right, there is a success message: 'Successfully queried the Prometheus API.' followed by the instruction 'Next, you can start to visualize data by [building a dashboard](#), or by querying data in the [Explore view](#)'. At the bottom right are 'Delete' and 'Save & test' buttons.

Configured Prometheus (port 9090) and Loki (port 3100) as datasources in Grafana for metrics and log visualization.



Verified metrics visualization on Prometheus dashboard

The screenshot shows the Grafana Explore interface for the 'loki' data source. The left sidebar is collapsed, and the main area displays log entries. The top navigation bar includes 'Home', 'Explore', 'loki', a search bar, and various control buttons like 'ctrl+k', 'Split', 'Add', and 'Live'. The central area has tabs for 'Outline' and 'loki', with 'loki' selected. Below is a 'Queries' section and a 'Logs' section. The logs are displayed as a list of text entries:

```
CCESS: running modules for final
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,940 - util.py[DEBUG]: cloud-init mode 'modules' took 0.162 seconds (0.17)
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,940 - util.py[DEBUG]: Read 10 bytes from /proc/uptime
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,940 - util.py[DEBUG]: Reading from /proc/uptime (quiet=False)
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,940 - util.py[DEBUG]: Creating symbolic link from '/run/cloud-init/result.json' => '../../../../../var/lib/cloud/data/result.json'
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,940 - atomic_helper.py[DEBUG]: Atomically writing to file /var/lib/cloud/data/result.json (via temporary file /var/lib/cloud/data/tmp3yu5rvk) - w: [644] 64 bytes/chars
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,939 - atomic_helper.py[DEBUG]: Atomically writing to file /var/lib/cloud/data/status.json (via temporary file /var/lib/cloud/data/tmp7f1n8q7j) - w: [644] 540 bytes/chars
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,939 - main.py[DEBUG]: Ran 20 modules with 0 failures
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,938 - handlers.py[DEBUG]: finish: modules-final/config-power-state-change: SUCCESS: config-power-state-change previously ran
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,938 - helpers.py[DEBUG]: config-power-state-change already ran (freq=once-per-instance)
: 2025-12-26 10:43:14.508 DEBUG 2025-12-26 03:04:03,938 - handlers.py[DEBUG]: start: modules-final/conf
```

Logs collected by Promtail are displayed in Grafana through Loki for monitoring and analysis.

Logs are organized and displayed in table format in Grafana via Loki for easy analysis

[Home](#)[Bookmarks](#)[Starred](#)[Dashboards](#)[Playlists](#)[Snapshots](#)[Library panels](#)[Shared dashboards](#)[Explore](#)[Drilldown](#)[Alerting](#)[Alert rules](#)[Contact points](#)[Notification policies](#)[Silences](#)[Active notifications](#)[Recently deleted](#)[Settings](#)

Alert-monitor-rule

Normal

Notifications are delivered to

email-alert

Evaluation interval

Every 1m

Edit

More

High CPU Usage

[Query and conditions](#) [Instances](#) [History](#) [Details](#) [Versions](#)**Rule****Rule type**

Grafana-managed alert rule

Rule identifier

ef88e1nn182kgc ⓘ

Last updated by

admin

Last updated at

2025-12-26 11:09:55 (a few seconds ago)

Evaluation**Last evaluated**

-

Last evaluation duration

0 ms

Pending period

1m

Keep firing for

2m

Alert state**Alert state if no data or all values are null**

NoData

Alert state if execution error or timeout

Error

Annotations**description**

Severity = Warning

summary

High CPU Usage

Notification configuration**Contact point**

email-alert

Alertmanager rule created, showing all configured settings and routing details.

Grafana

Home > Alerting > Alert rules

Contact point: Choose a contact point

Search: Search... (ctrl+k)

View as: Grouped (selected), List, State

1 rule 1 normal

Grafana-managed

Alert-folder > Monitor-evaluation

1 normal | 1m | ⏲ ⚒

State	Name	Health	Summary	Next evaluation	Actions
Normal	Alert-monitor-rule	ok	High CPU Usage	in a few seconds	<a>View <a>Edit <a>More

Data source-managed

No rules found.

+ New data source-managed recording rule

Alert rules (highlighted)

Contact points

Notification policies

Silences

Active notifications

Recently deleted

Settings

Alert rule status displayed as **normal, indicating no active alerts**

PROJECT SUMMARY

- This project successfully showcases an end-to-end monitoring and alerting framework using modern cloud and container technologies.
- By combining metrics collection, visualization, and alerting, it enables real-time system monitoring, faster troubleshooting, and improved infrastructure reliability.
- The setup demonstrates cloud-native tools like Prometheus, Grafana, Loki, and Alertmanager can work together seamlessly, providing a scalable and efficient solution for monitoring applications.
- Detailed documentation is available in [GitHub](#)

Thank you

 [PRIYANKA RAJAGOPAL](#)

 priyankaraj0919@gmail.com