Prometheus &Grafana

Mr.Tejas pokale

https://www.linkedin.com/in/tejas-pokale-devops/

What is Prometheus?

Prometheus is an open-source systems monitoring and alerting toolkit originally built at SoundCloud. Since its inception in 2012, many companies and organizations have adopted Prometheus, and the project has a very active developer and user community. It is now a standalone open source project and maintained independently of any company. To emphasize this, and to clarify the project's governance structure, Prometheus joined the Cloud Native Computing Foundation in 2016 as the second hosted project, after Kubernetes.

Prometheus collects and stores its metrics as time series data, i.e. metrics information is stored with the timestamp at which it was recorded, alongside optional key-value pairs called labels.

For more elaborate overviews of Prometheus, see the resources linked from the media section.

Features

Prometheus's main features are:

- a multi-dimensional data model with time series data identified by metric name and key/value pairs
- PromQL, a flexible query language to leverage this dimensionality
- no reliance on distributed storage; single server nodes are autonomous
- time series collection happens via a pull model over HTTP
- pushing time series is supported via an intermediary gateway
- targets are discovered via service discovery or static configuration
- multiple modes of graphing and dashboarding support

What are metrics?

Metrics are numerical measurements in layperson terms. The term time series refers to the recording of changes over time. What users want to measure differs from application to application. For a web server, it could be request times; for a database, it could be the number of active connections or active queries, and so on.

Metrics play an important role in understanding why your application is working in a certain way. Let's assume you are running a web application and discover that it is slow. To learn what is happening with your application, you will need some information. For example, when the number of requests is high, the application may become slow. If you have the request count metric, you can determine the cause and increase the number of servers to handle the load.

Components

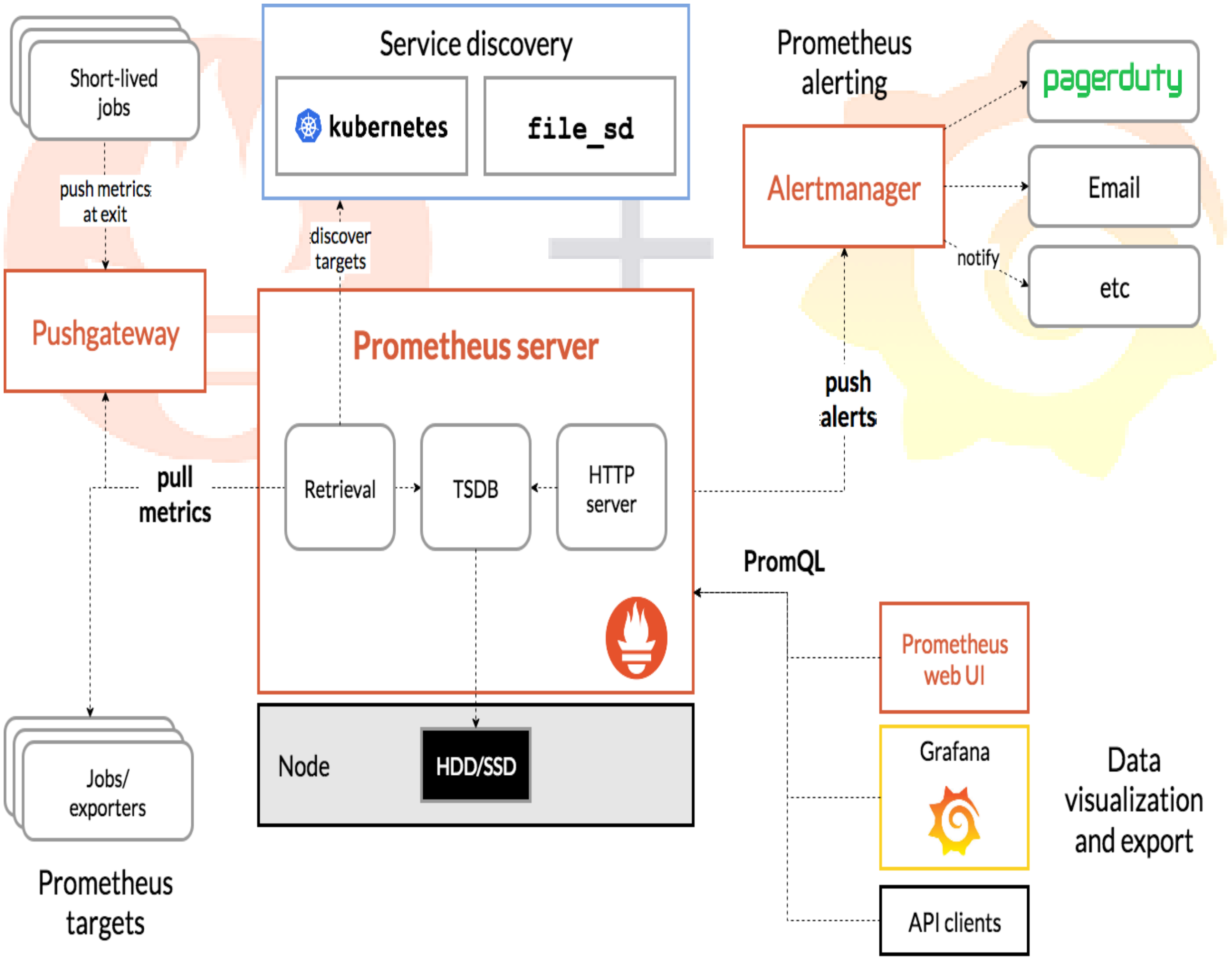
The Prometheus ecosystem consists of multiple components, many of which are optional:

- the main Prometheus server which scrapes and stores time series data

- client libraries for instrumenting application code
- a push gateway for supporting short-lived jobs
- special-purpose exporters for services like HAProxy, StatsD, Graphite, etc.
- an alertmanager to handle alerts
- various support tools

Architecture

This diagram illustrates the architecture of Prometheus and some of its ecosystem components:



When does it fit?

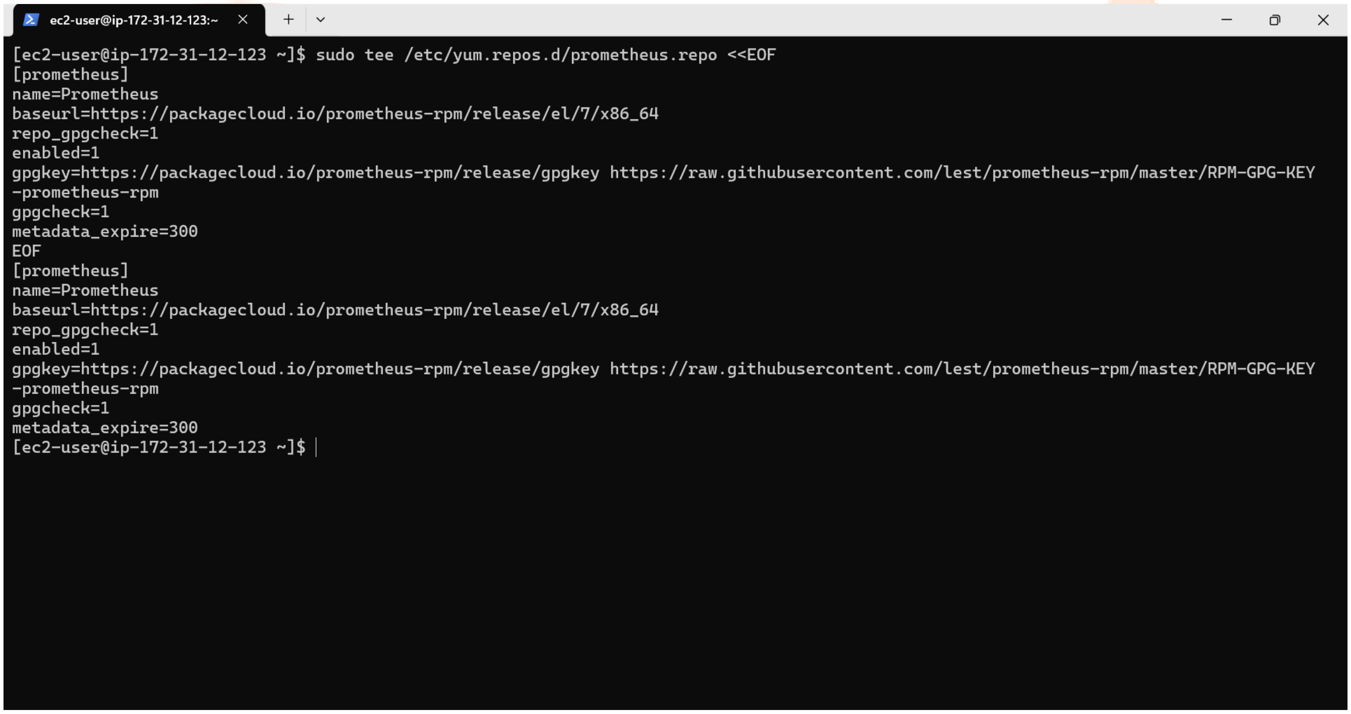
Prometheus works well for recording any purely numeric time series. It fits both machine-centric monitoring as well as monitoring of highly dynamic service-oriented architectures. In a world of microservices, its support for multi-dimensional data collection and querying is a particular strength.

Prometheus is designed for reliability, to be the system you go to during an outage to allow you to quickly diagnose problems. Each Prometheus server is standalone, not depending on network storage or other remote services. You can rely on it when other parts of your infrastructure are broken, and you do not need to setup extensive infrastructure to use it.

First steps with Prometheus

Installation

# Prometheus installation



```
[ec2-user@ip-172-31-12-123 ~]$ sudo tee /etc/yum.repos.d/prometheus.repo <<EOF
[prometheus]
name=Prometheus
baseurl=https://packagecloud.io/prometheus-rpm/release/el/7/x86_64
repo_gpgcheck=1
enabled=1
gpgkey=https://packagecloud.io/prometheus-rpm/release/gpgkey https://raw.githubusercontent.com/lest/prometheus-rpm/master/RPM-GPG-KEY
-prometheus-rpm
gpgcheck=1
metadata_expire=300
EOF
[prometheus]
name=Prometheus
baseurl=https://packagecloud.io/prometheus-rpm/release/el/7/x86_64
repo_gpgcheck=1
enabled=1
gpgkey=https://packagecloud.io/prometheus-rpm/release/gpgkey https://raw.githubusercontent.com/lest/prometheus-rpm/master/RPM-GPG-KEY
-prometheus-rpm
gpgcheck=1
metadata_expire=300
[ec2-user@ip-172-31-12-123 ~]$
```

sudo tee /etc/yum.repos.d/prometheus.repo <<EOF

[prometheus]

name=Prometheus

baseurl=https://packagecloud.io/prometheus-rpm/release/el/7/x86_64

repo_gpgcheck=1

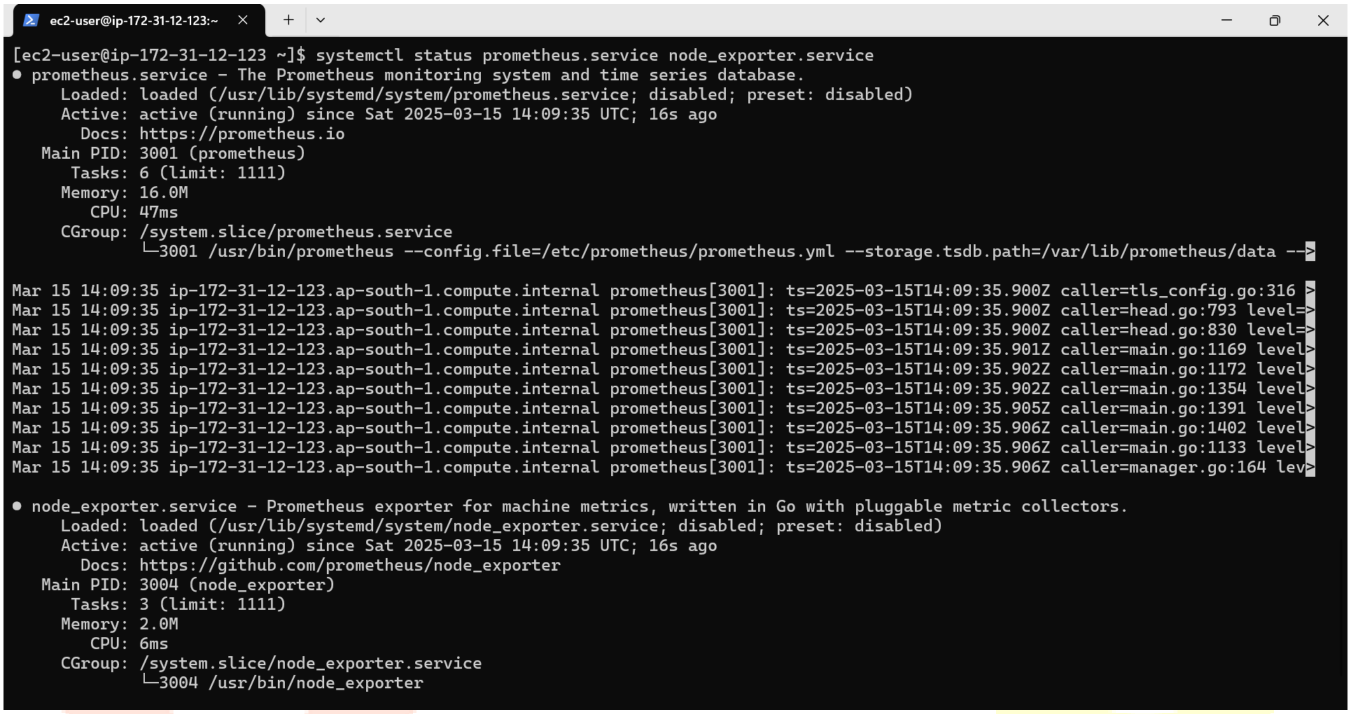
enabled=1

gpgkey=https://packagecloud.io/prometheus-rpm/release/gpgkey
https://raw.githubusercontent.com/lest/prometheus-rpm/master/RPM-GPG-KEY-prometheus-rpm

gpgcheck=1

metadata_expire=300

EOF



```
[ec2-user@ip-172-31-12-123 ~]$ systemctl status prometheus.service node_exporter.service
● prometheus.service - The Prometheus monitoring system and time series database.
     Loaded: loaded (/usr/lib/systemd/system/prometheus.service; disabled; preset: disabled)
     Active: active (running) since Sat 2025-03-15 14:09:35 UTC; 16s ago
       Docs: https://prometheus.io
   Main PID: 3001 (prometheus)
      Tasks: 6 (limit: 1111)
     Memory: 16.0M
        CPU: 47ms
     CGroup: /system.slice/prometheus.service
             └─3001 /usr/bin/prometheus --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/var/lib/prometheus/data -->

Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.900Z caller=tls_config.go:316 >
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.900Z caller=head.go:793 level=>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.900Z caller=head.go:830 level=>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.901Z caller=main.go:1169 level>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.902Z caller=main.go:1172 level>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.902Z caller=main.go:1354 level>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.905Z caller=main.go:1391 level>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.906Z caller=main.go:1402 level>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.906Z caller=main.go:1133 level>
Mar 15 14:09:35 ip-172-31-12-123.ap-south-1.compute.internal prometheus[3001]: ts=2025-03-15T14:09:35.906Z caller=manager.go:164 lev>

● node_exporter.service - Prometheus exporter for machine metrics, written in Go with pluggable metric collectors.
     Loaded: loaded (/usr/lib/systemd/system/node_exporter.service; disabled; preset: disabled)
     Active: active (running) since Sat 2025-03-15 14:09:35 UTC; 16s ago
       Docs: https://github.com/prometheus/node_exporter
   Main PID: 3004 (node_exporter)
      Tasks: 3 (limit: 1111)
     Memory: 2.0M
        CPU: 6ms
     CGroup: /system.slice/node_exporter.service
             └─3004 /usr/bin/node_exporter
```

sudo yum update -y

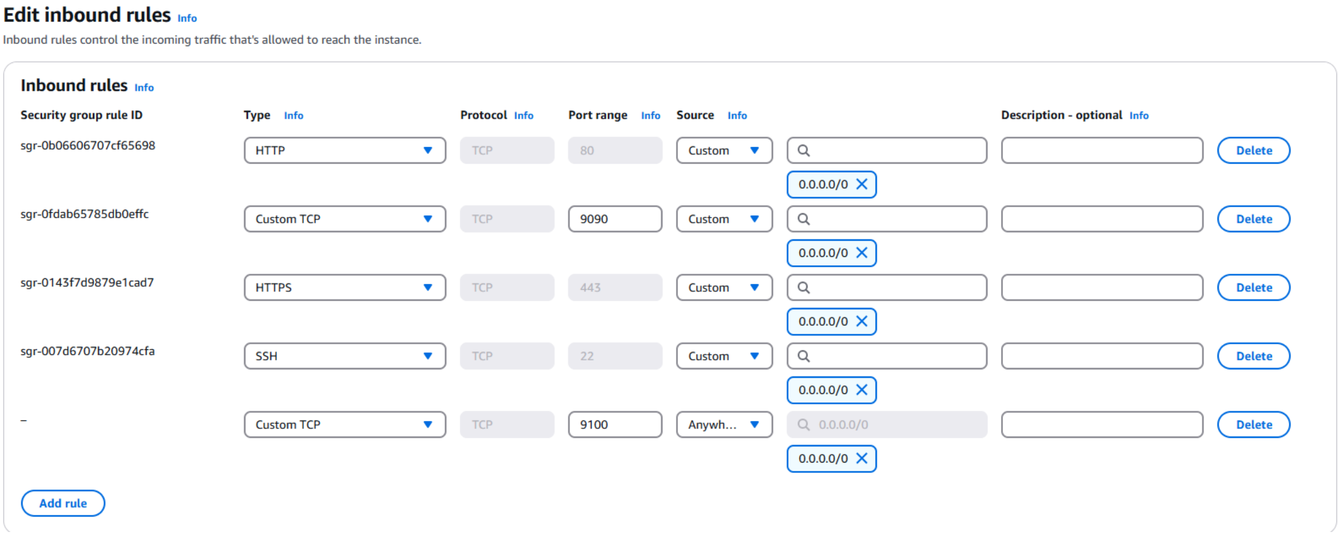
- sudo yum -y install prometheus2 node_exporter

- rpm -qi prometheus2

- sudo systemctl start prometheus node_exporter

- systemctl status prometheus.service node_exporter.service

Add port 9090&9100 in security group



Edit inbound rules Info

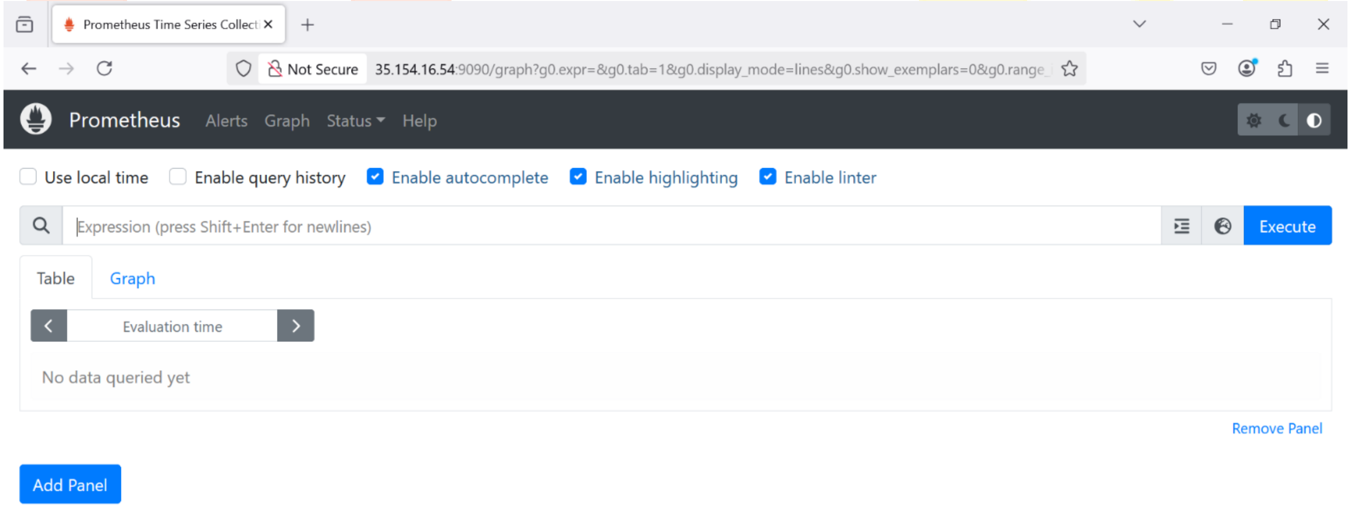
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0b06606707cf65698	HTTP	TCP	80	Custom	0.0.0.0/0		Delete
sgr-0fdab65785db0effc	Custom TCP	TCP	9090	Custom	0.0.0.0/0		Delete
sgr-0143f7d9879e1cad7	HTTPS	TCP	443	Custom	0.0.0.0/0		Delete
sgr-007d6707b20974cfa	SSH	TCP	22	Custom	0.0.0.0/0		Delete
–	Custom TCP	TCP	9100	Anywh...	0.0.0.0/0 0.0.0.0/0		Delete

copy ec2 public IP and paste in browser with port no 9090

- now you should see prometheus dashboard



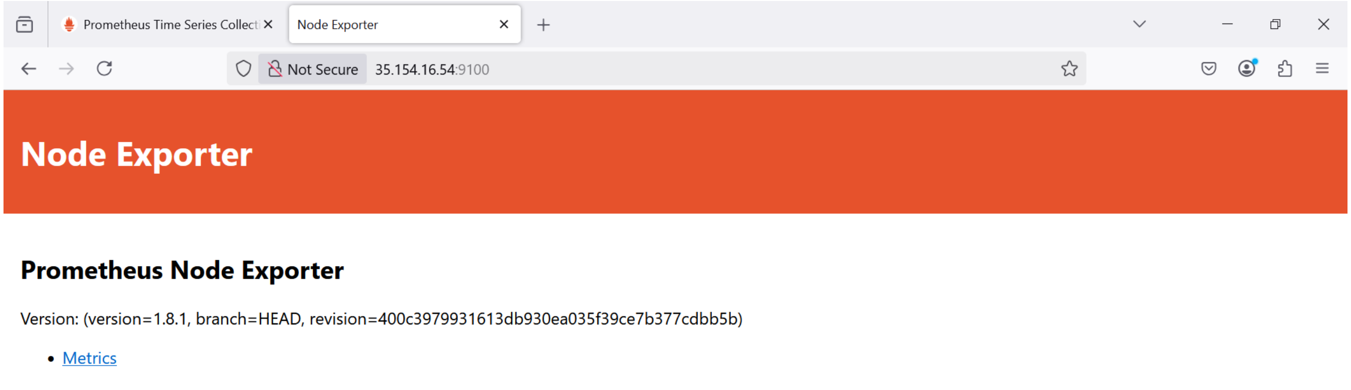
35.154.16.54:9090/graph?g0.expr=&g0.tab=1&g0.display_mode=lines&g0.show_exemplars=0&g0.range_

□ Use local time □ Enable query history ☑ Enable autocomplete ☑ Enable highlighting ☑ Enable linter

Expression (press Shift+Enter for newlines)

Evaluation time

No data queried yet



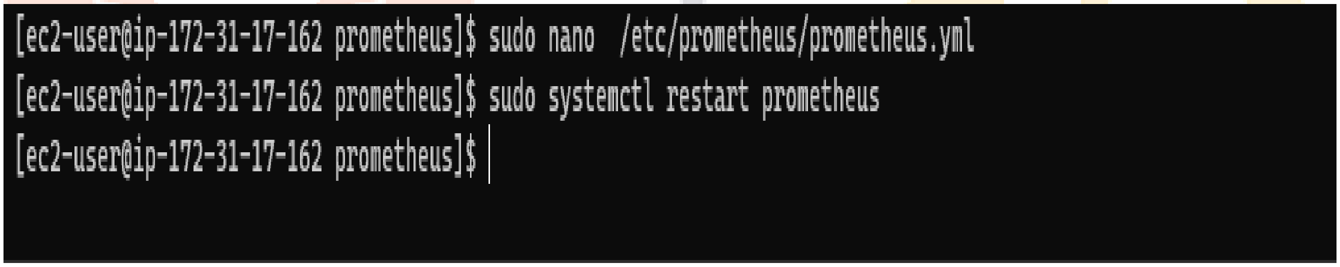
Node Exporter

Prometheus Node Exporter

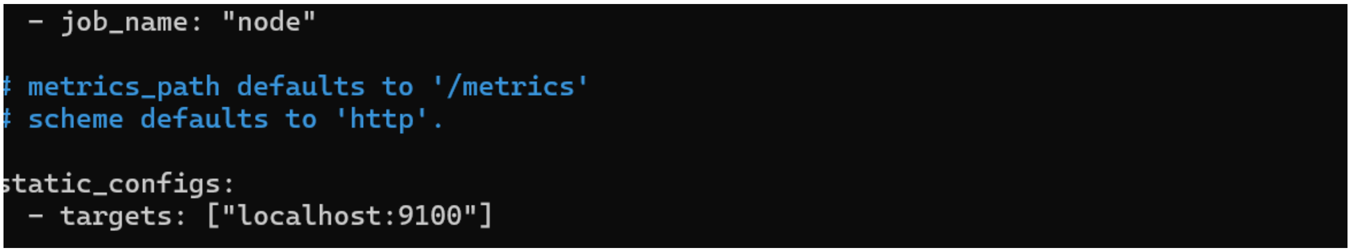
Version: (version=1.8.1, branch=HEAD, revision=400c3979931613db930ea035f39ce7b377cdbb5b)

- Metrics

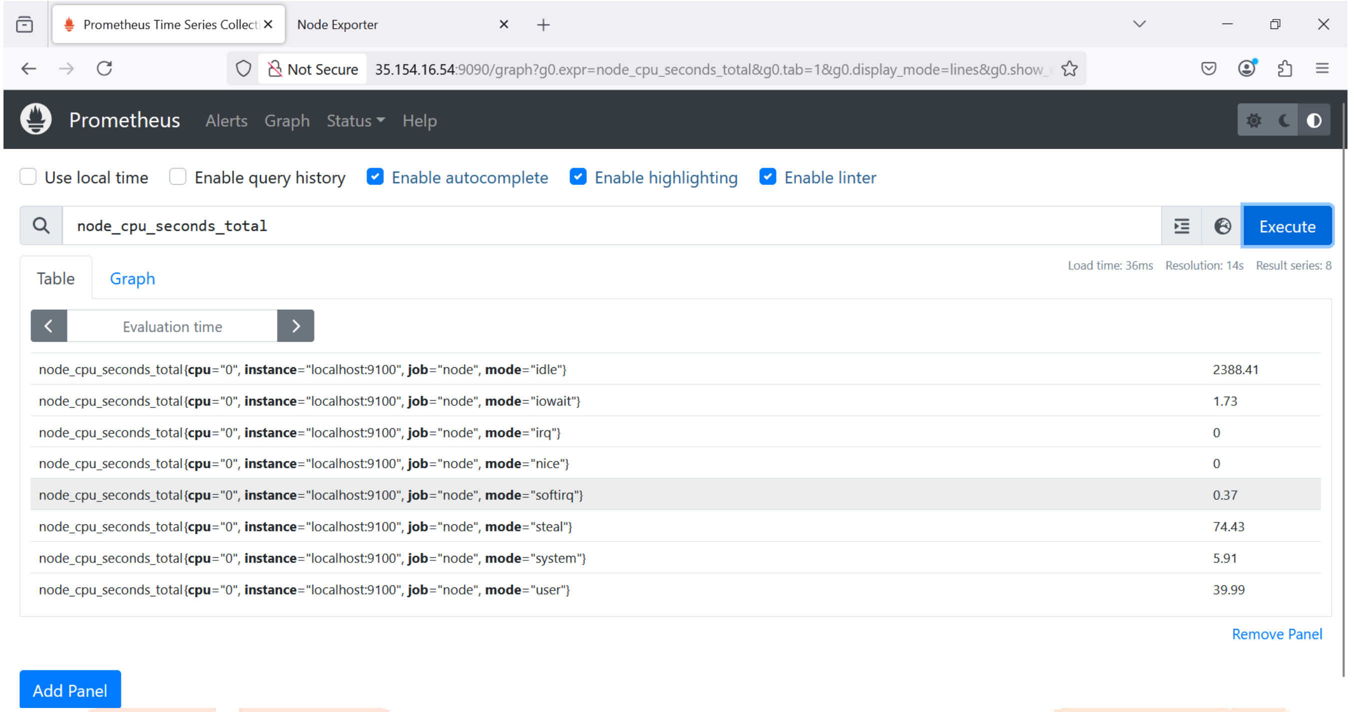
Checking the cpu usages



```
[ec2-user@ip-172-31-17-162 prometheus]$ sudo nano  /etc/prometheus/prometheus.yml
[ec2-user@ip-172-31-17-162 prometheus]$ sudo systemctl restart prometheus
[ec2-user@ip-172-31-17-162 prometheus]$
```



```
 - job_name: "node"

# metrics_path defaults to '/metrics'
# scheme defaults to 'http'.

static_configs:
 - targets: ["localhost:9100"]
```


Prometheus Time Series Collect
Node Exporter
Not Secure
35.154.16.54:9090/graph?g0.expr=node_cpu_seconds_total&g0.tab=1&g0.display_mode=lines&g0.show_
Prometheus
Alerts
Graph
Status
Help

□ Use local time □ Enable query history ☑ Enable autocomplete ☑ Enable highlighting ☑ Enable linter

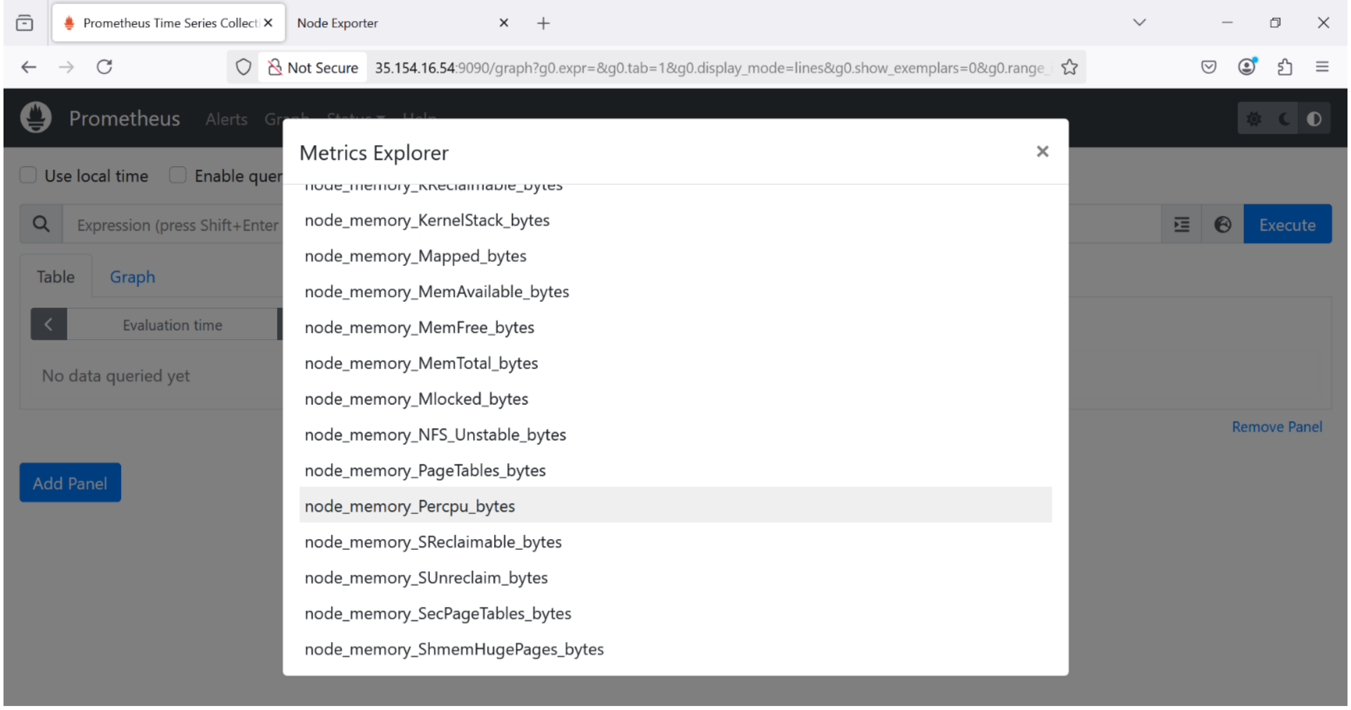
```
node_cpu_seconds_total
```

Load time: 36ms Resolution: 14s Result series: 8

Table
Graph
Evaluation time

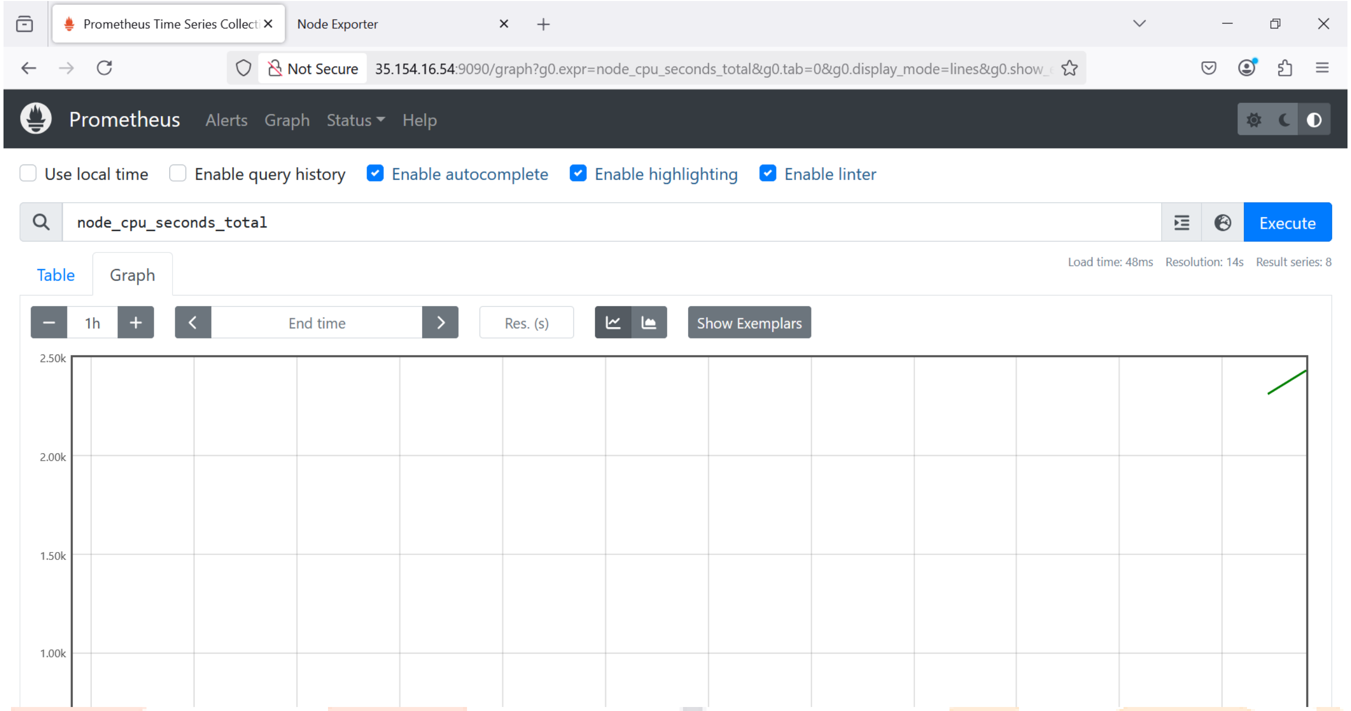
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="idle"}	2388.41
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="iowait"}	1.73
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="irq"}	0
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="nice"}	0
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="softirq"}	0.37
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="steal"}	74.43
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="system"}	5.91
node_cpu_seconds_total{**cpu**="0", **instance**="localhost:9100", **job**="node", **mode**="user"}	39.99

Metric explorer


Prometheus Time Series Collect
Node Exporter
Not Secure
35.154.16.54:9090/graph?g0.expr=&g0.tab=1&g0.display_mode=lines&g0.show_exemplars=0&g0.range_
Prometheus
Alerts
Use local time
Enable quer
Expression (press Shift+Enter
Execute
Table
Graph
Evaluation time
No data queried yet
Remove Panel
Add Panel

Metrics Explorer

node_memory_KReclaimable_bytes
node_memory_KernelStack_bytes
node_memory_Mapped_bytes
node_memory_MemAvailable_bytes
node_memory_MemFree_bytes
node_memory_MemTotal_bytes
node_memory_Mlocked_bytes
node_memory_NFS_Unstable_bytes
node_memory_PageTables_bytes
node_memory_Percpu_bytes
node_memory_SReclaimable_bytes
node_memory_SUnreclaim_bytes
node_memory_SecPageTables_bytes
node_memory_ShmemHugePages_bytes



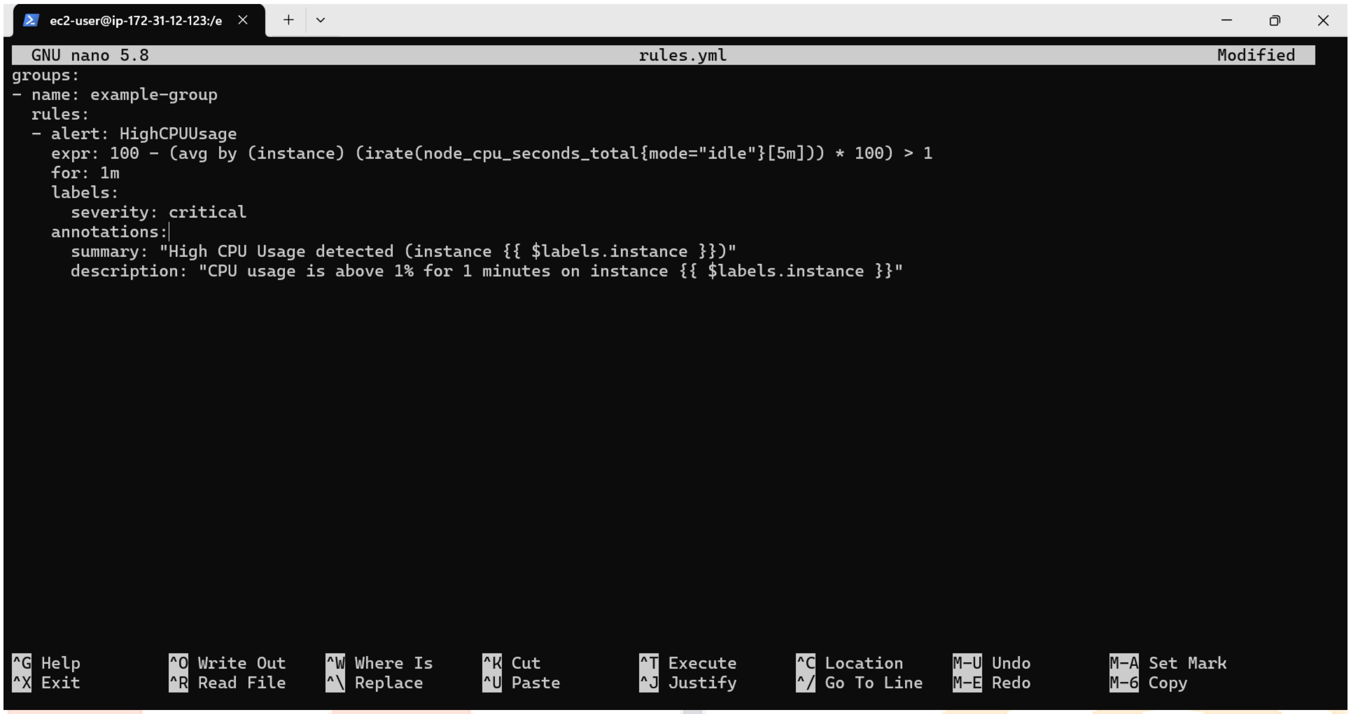
Not Secure 35.154.16.54:9090/graph?g0.expr=node_cpu_seconds_total&g0.tab=0&g0.display_mode=lines&g0.show_

□ Use local time □ Enable query history ☑ Enable autocomplete ☑ Enable highlighting ☑ Enable linter

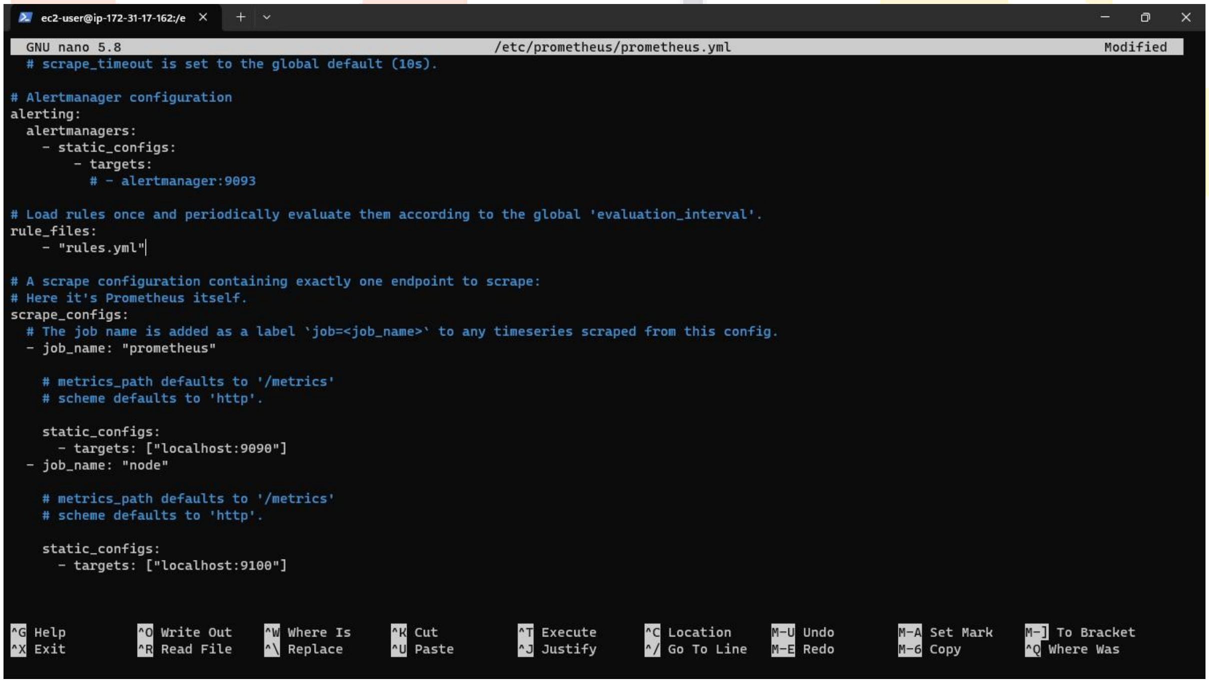
```
node_cpu_seconds_total
```

Load time: 48ms Resolution: 14s Result series: 8

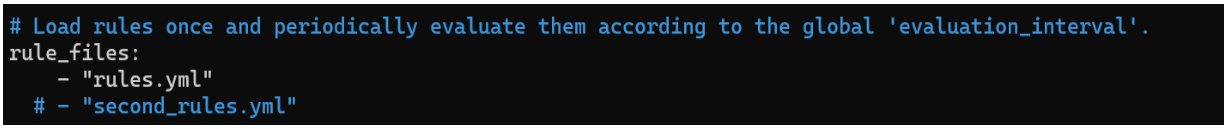
Rules.yml



```
  GNU nano 5.8                            rules.yml                              Modified
groups:
- name: example-group
  rules:
  - alert: HighCPUUsage
    expr: 100 - (avg by (instance) (irate(node_cpu_seconds_total{mode="idle"}[5m])) * 100) > 1
    for: 1m
    labels:
      severity: critical
    annotations:
      summary: "High CPU Usage detected (instance {{ $labels.instance }})"
      description: "CPU usage is above 1% for 1 minutes on instance {{ $labels.instance }}"
```

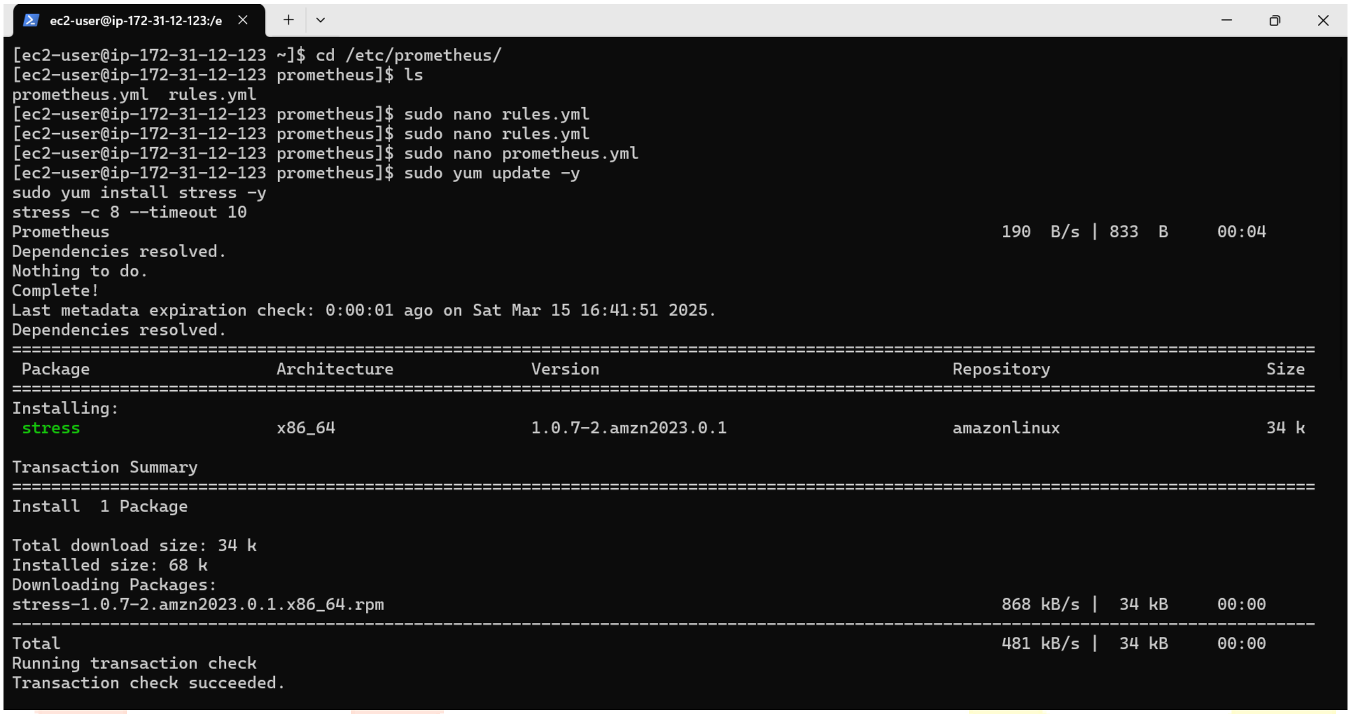


```
  GNU nano 5.8                  /etc/prometheus/prometheus.yml                  Modified
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
    - "rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]
  - job_name: "node"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9100"]
```



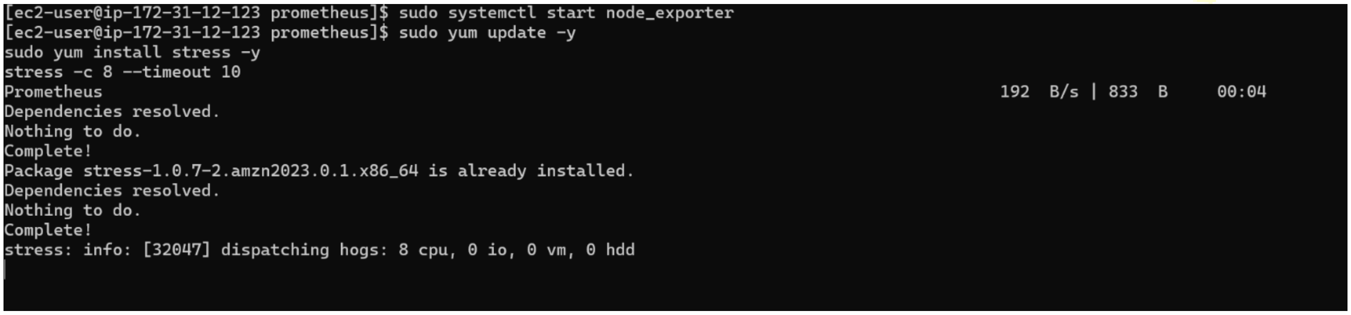
```
# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
    - "rules.yml"
  # - "second_rules.yml"
```

Increased the stress of cpu

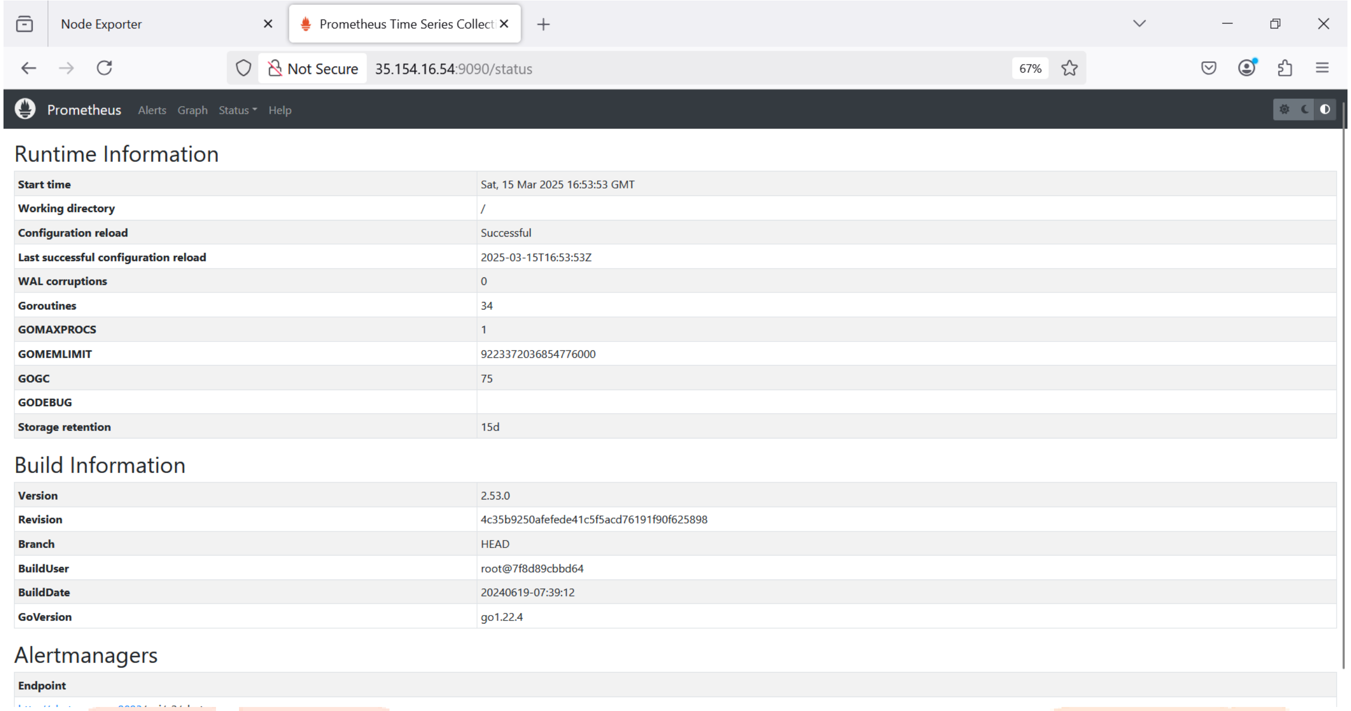


```
[ec2-user@ip-172-31-12-123 ~]$ cd /etc/prometheus/
[ec2-user@ip-172-31-12-123 prometheus]$ ls
prometheus.yml  rules.yml
[ec2-user@ip-172-31-12-123 prometheus]$ sudo nano rules.yml
[ec2-user@ip-172-31-12-123 prometheus]$ sudo nano rules.yml
[ec2-user@ip-172-31-12-123 prometheus]$ sudo nano prometheus.yml
[ec2-user@ip-172-31-12-123 prometheus]$ sudo yum update -y
sudo yum install stress -y
stress -c 8 --timeout 10
Prometheus                                                                                     190  B/s | 833  B     00:04
Dependencies resolved.
Nothing to do.
Complete!
Last metadata expiration check: 0:00:01 ago on Sat Mar 15 16:41:51 2025.
Dependencies resolved.
==================================================================================================================================
 Package                   Architecture              Version                                   Repository                    Size
==================================================================================================================================
Installing:
 stress                    x86_64                    1.0.7-2.amzn2023.0.1                      amazonlinux                   34 k

Transaction Summary
==================================================================================================================================
Install  1 Package

Total download size: 34 k
Installed size: 68 k
Downloading Packages:
stress-1.0.7-2.amzn2023.0.1.x86_64.rpm                                                         868 kB/s |  34 kB     00:00
----------------------------------------------------------------------------------------------------------------------------------
Total                                                                                          481 kB/s |  34 kB     00:00
Running transaction check
Transaction check succeeded.
```

Restart the node _exporter



```
[ec2-user@ip-172-31-12-123 prometheus]$ sudo systemctl start node_exporter
[ec2-user@ip-172-31-12-123 prometheus]$ sudo yum update -y
sudo yum install stress -y
stress -c 8 --timeout 10
Prometheus                                                                                     192  B/s | 833  B     00:04
Dependencies resolved.
Nothing to do.
Complete!
Package stress-1.0.7-2.amzn2023.0.1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
stress: info: [32047] dispatching hogs: 8 cpu, 0 io, 0 vm, 0 hdd
```



Runtime Information

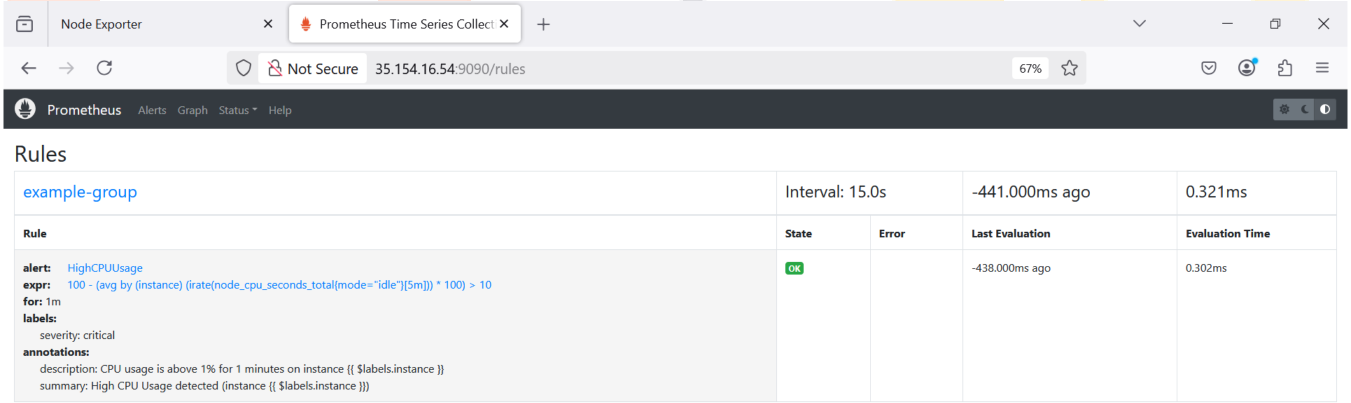
Start time	Sat, 15 Mar 2025 16:53:53 GMT
Working directory	/
Configuration reload	Successful
Last successful configuration reload	2025-03-15T16:53:53Z
WAL corruptions	0
Goroutines	34
GOMAXPROCS	1
GOMEMLIMIT	9223372036854776000
GOGC	75
GODEBUG	
Storage retention	15d

Build Information

Version	2.53.0
Revision	4c35b9250afefede41c5f5acd76191f90f625898
Branch	HEAD
BuildUser	root@7f8d89cbbd64
BuildDate	20240619-07:39:12
GoVersion	go1.22.4

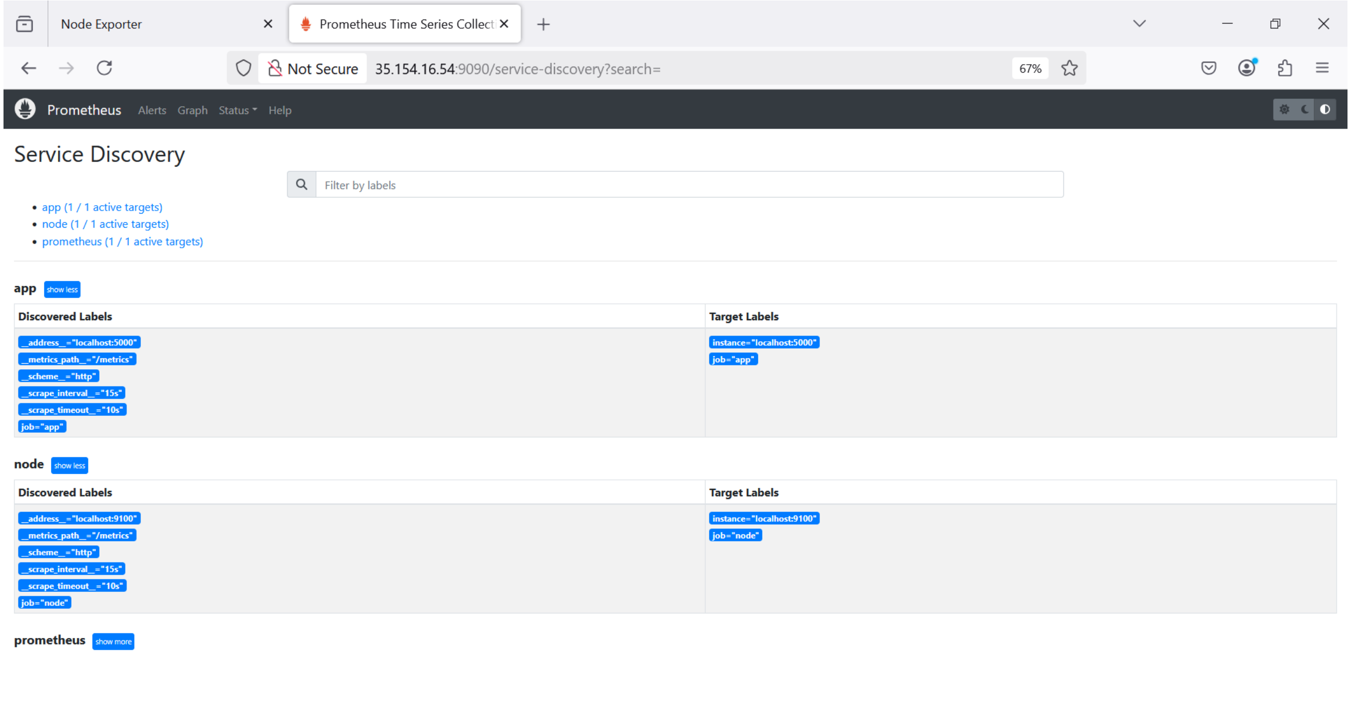
Alertmanagers

Endpoint



Rules

example-group		Interval: 15.0s	-441.000ms ago	0.321ms
Rule	**State**	**Error**	**Last Evaluation**	**Evaluation Time**
alert: HighCPUUsage **expr:** 100 - (avg by (instance) (irate(node_cpu_seconds_total{mode="idle"}[5m])) * 100) > 10 **for:** 1m **labels:** severity: critical **annotations:** description: CPU usage is above 1% for 1 minutes on instance {{ $labels.instance }} summary: High CPU Usage detected (instance {{ $labels.instance }})	OK		-438.000ms ago	0.302ms



Node Exporter
Prometheus Time Series Collect
Not Secure
35.154.16.54:9090/service-discovery?search=
67%
Prometheus
Alerts
Graph
Status
Help

Service Discovery

Filter by labels

- app (1 / 1 active targets)
- node (1 / 1 active targets)
- prometheus (1 / 1 active targets)

app show less

Discovered Labels	Target Labels

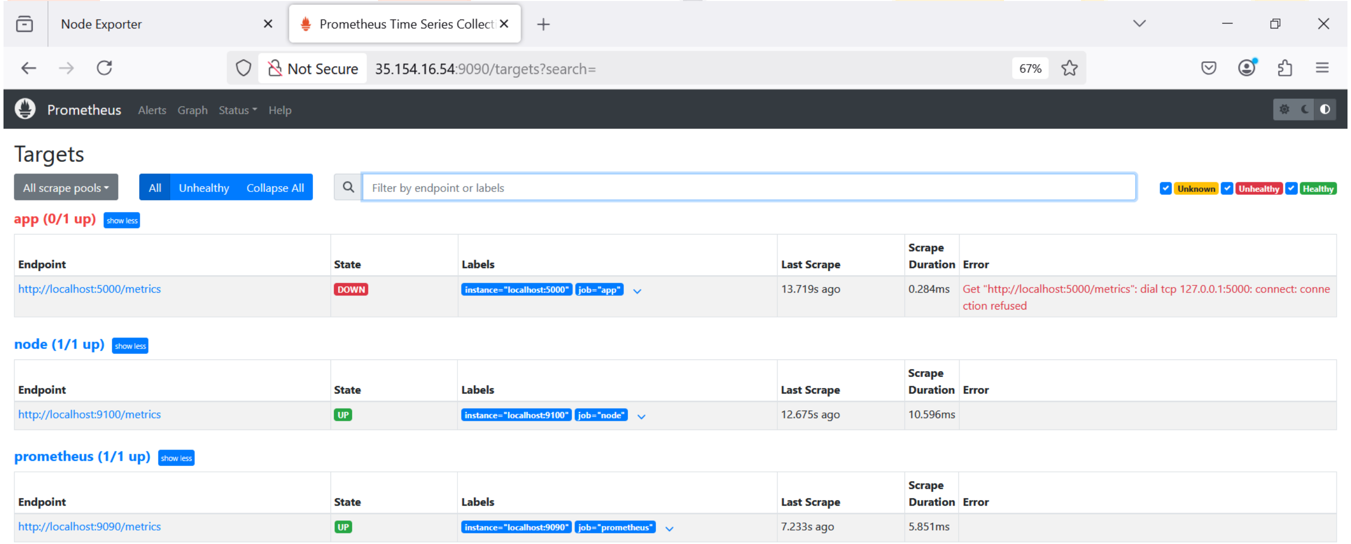
__address__="localhost:5000"
__metrics_path__="/metrics"
__scheme__="http"
__scrape_interval__="15s"
__scrape_timeout__="10s"
job="app"
	instance="localhost:5000"
	job="app"

node show less

Discovered Labels	Target Labels

__address__="localhost:9100"
__metrics_path__="/metrics"
__scheme__="http"
__scrape_interval__="15s"
__scrape_timeout__="10s"
job="node"
	instance="localhost:9100"
	job="node"

prometheus show more



Node Exporter
Prometheus Time Series Collect
Not Secure
35.154.16.54:9090/targets?search=
67%
Prometheus
Alerts
Graph
Status
Help

Targets

All scrape pools ▾ | All | Unhealthy | Collapse All | Filter by endpoint or labels | ☑ Unknown ☑ Unhealthy ☑ Healthy

app (0/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:5000/metrics	DOWN	instance="localhost:5000" job="app"	13.719s ago	0.284ms	Get "http://localhost:5000/metrics": dial tcp 127.0.0.1:5000: connect: connection refused

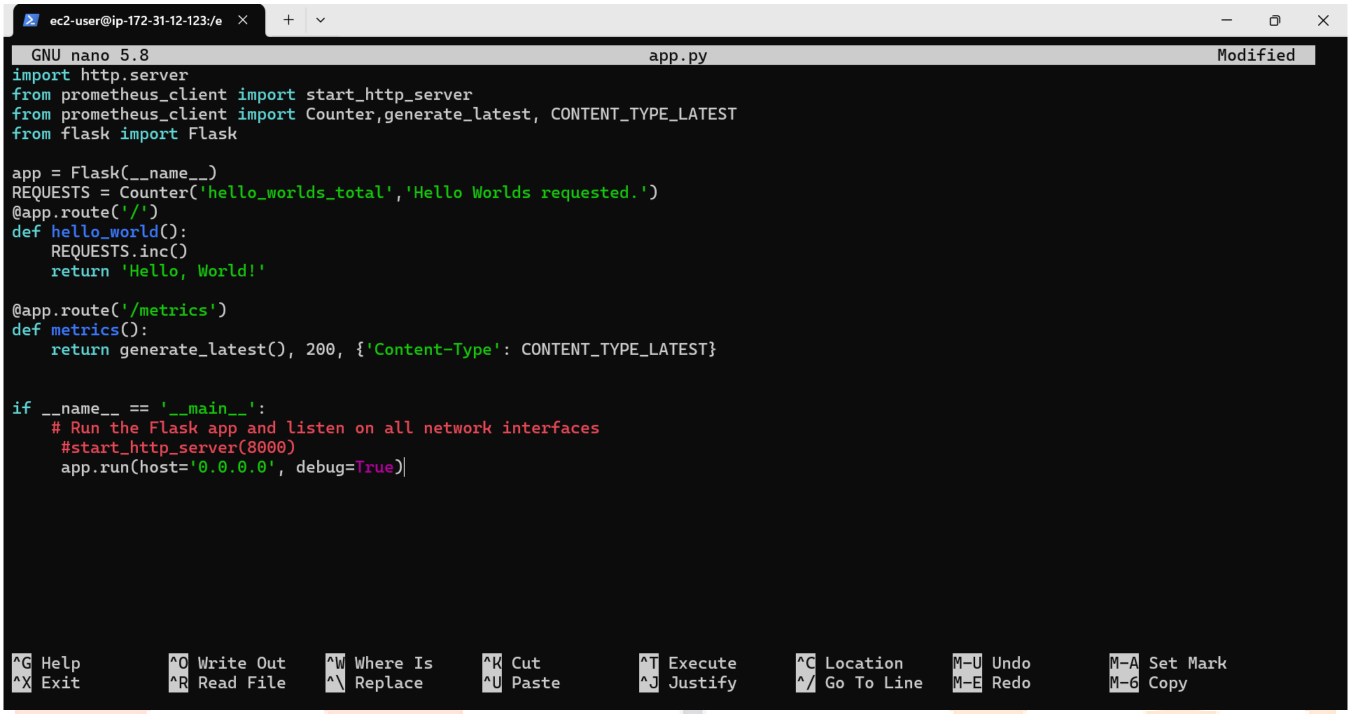
node (1/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9100/metrics	UP	instance="localhost:9100" job="node"	12.675s ago	10.596ms	

prometheus (1/1 up) show less

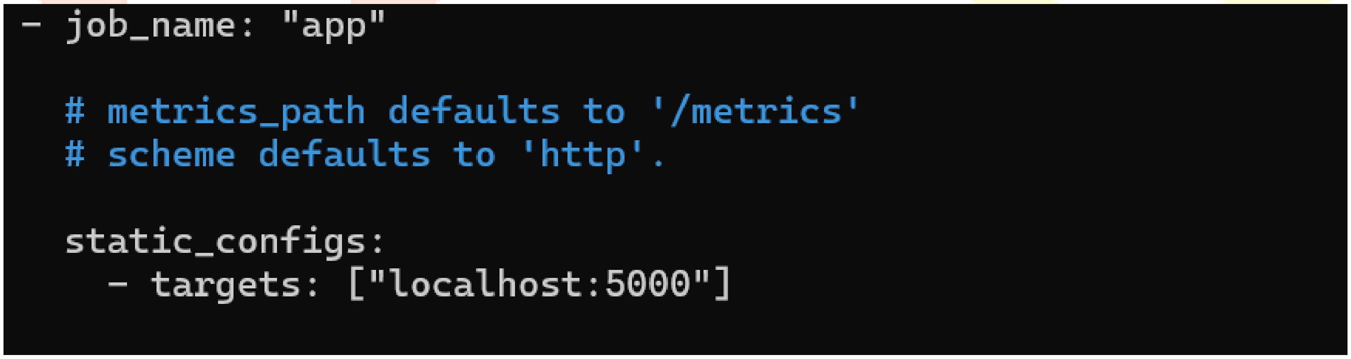
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	7.233s ago	5.851ms	

Application usage checking –



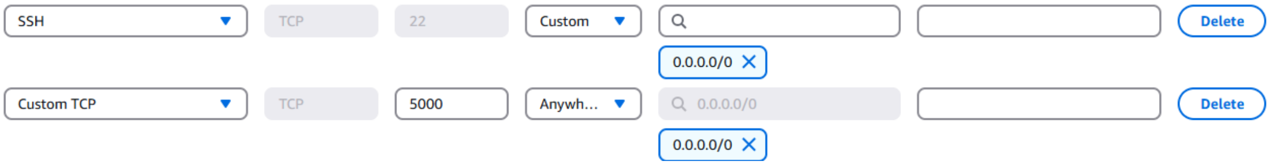
```
 GNU nano 5.8                              app.py                              Modified
import http.server
from prometheus_client import start_http_server
from prometheus_client import Counter,generate_latest, CONTENT_TYPE_LATEST
from flask import Flask

app = Flask(__name__)
REQUESTS = Counter('hello_worlds_total','Hello Worlds requested.')
@app.route('/')
def hello_world():
    REQUESTS.inc()
    return 'Hello, World!'

@app.route('/metrics')
def metrics():
    return generate_latest(), 200, {'Content-Type': CONTENT_TYPE_LATEST}


if __name__ == '__main__':
    # Run the Flask app and listen on all network interfaces
     #start_http_server(8000)
     app.run(host='0.0.0.0', debug=True)








^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

Add the job name promethues.yml file

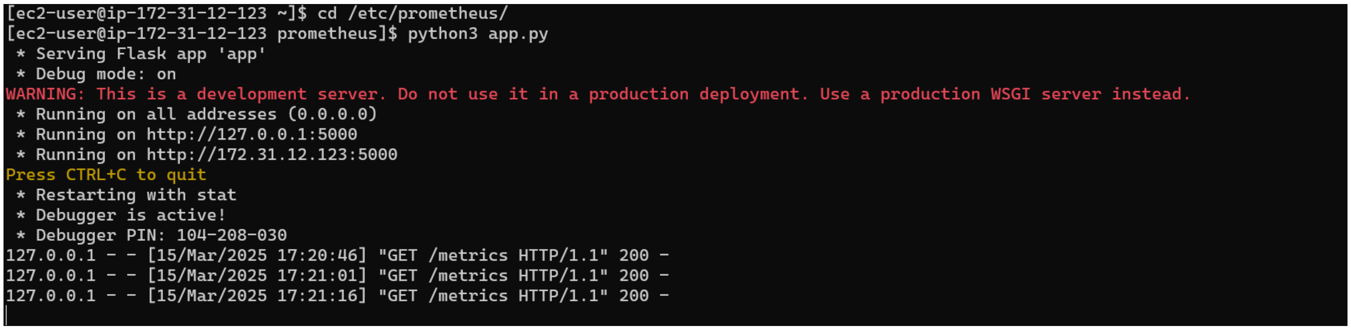


```
- job_name: "app"

  # metrics_path defaults to '/metrics'
  # scheme defaults to 'http'.

  static_configs:
    - targets: ["localhost:5000"]
```

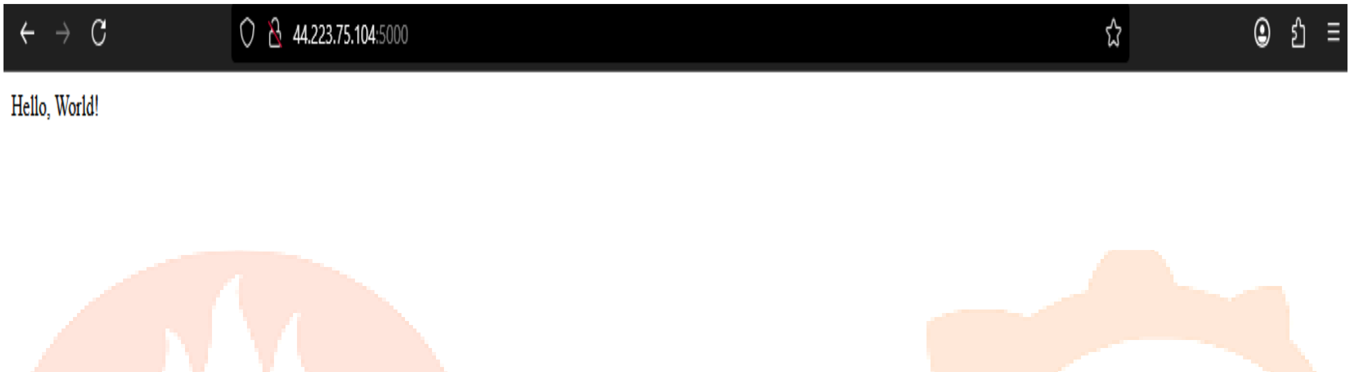
Add 5000 port to security



SSH ▼	TCP	22	Custom ▼	🔍 0.0.0.0/0 ✕		Delete
Custom TCP ▼	TCP	5000	Anywh... ▼	🔍 0.0.0.0/0 0.0.0.0/0 ✕		Delete

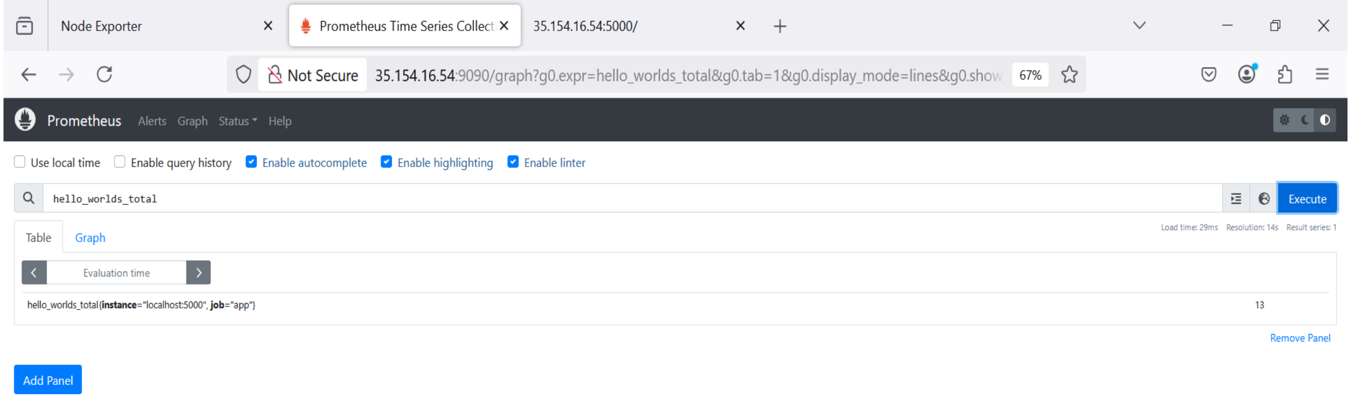


```
[ec2-user@ip-172-31-12-123 ~]$ cd /etc/prometheus/
[ec2-user@ip-172-31-12-123 prometheus]$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.31.12.123:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 104-208-030
127.0.0.1 - - [15/Mar/2025 17:20:46] "GET /metrics HTTP/1.1" 200 -
127.0.0.1 - - [15/Mar/2025 17:21:01] "GET /metrics HTTP/1.1" 200 -
127.0.0.1 - - [15/Mar/2025 17:21:16] "GET /metrics HTTP/1.1" 200 -
```



44.223.75.104:5000

Hello, World!



35.154.16.54:9090/graph?g0.expr=hello_worlds_total&g0.tab=1&g0.display_mode=lines&g0.show

□ Use local time □ Enable query history ☑ Enable autocomplete ☑ Enable highlighting ☑ Enable linter

hello_worlds_total

Load time: 29ms Resolution: 14s Result series: 1

hello_worlds_total{instance="localhost:5000", job="app"} 13

Grafana

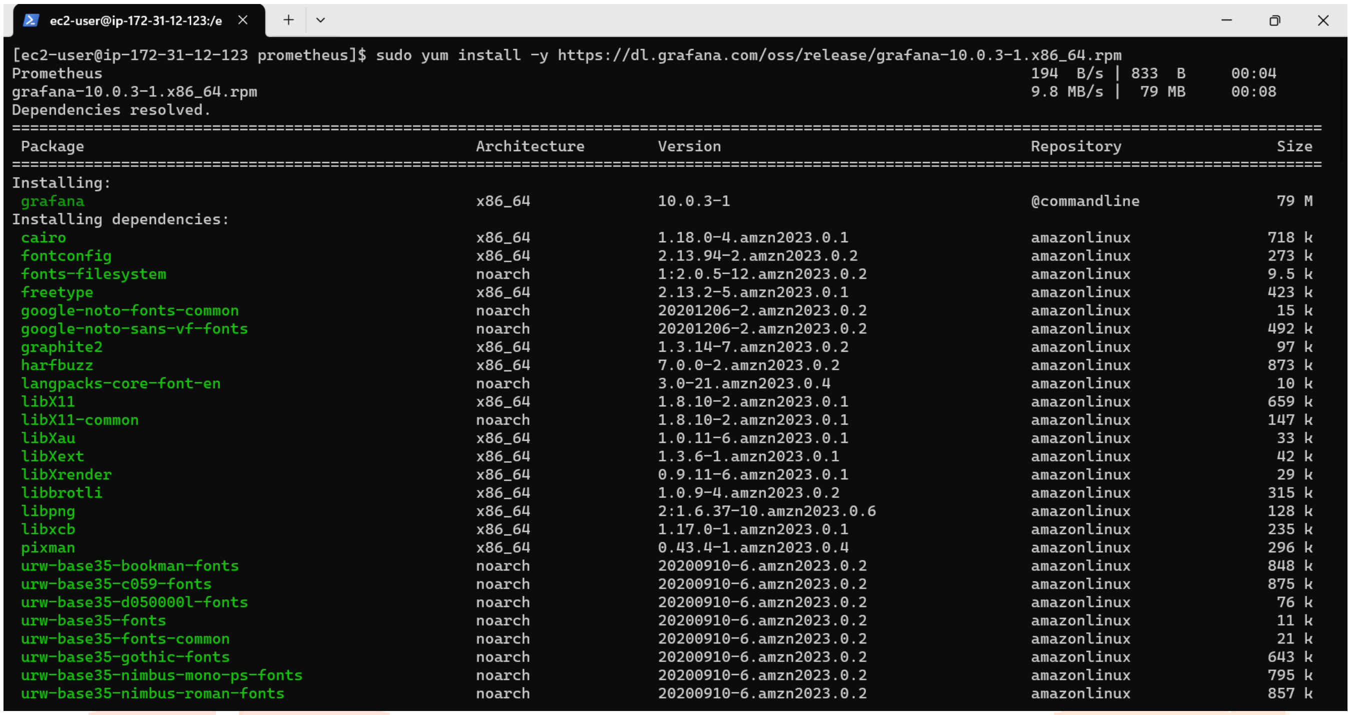
Introduction

Grafana Open Source Software (OSS) enables you to query, visualize, alert on, and explore your metrics, logs, and traces wherever they’re stored. Grafana data source plugins enable you to query data sources including time series databases like Prometheus and CloudWatch, logging tools like Loki and Elasticsearch, NoSQL/SQL databases like Postgres, CI/CD tooling like GitHub, and many more. Grafana OSS provides you with tools to display that data on live dashboards with insightful graphs and visualizations.

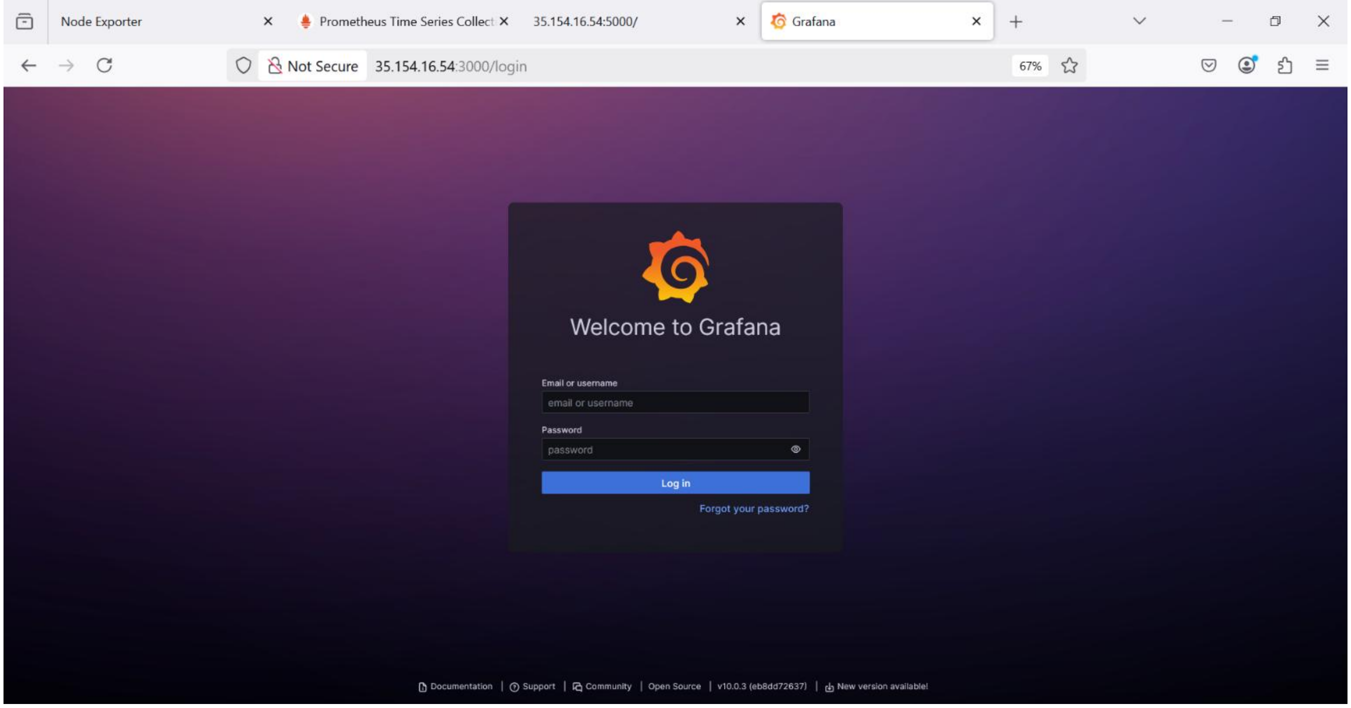
Grafana Enterprise is a commercial edition of Grafana that includes exclusive data source plugins and additional features not found in the open source version. You also get 24x7x365 support and training from the core Grafana team. To learn more about these features, refer to Enterprise features.

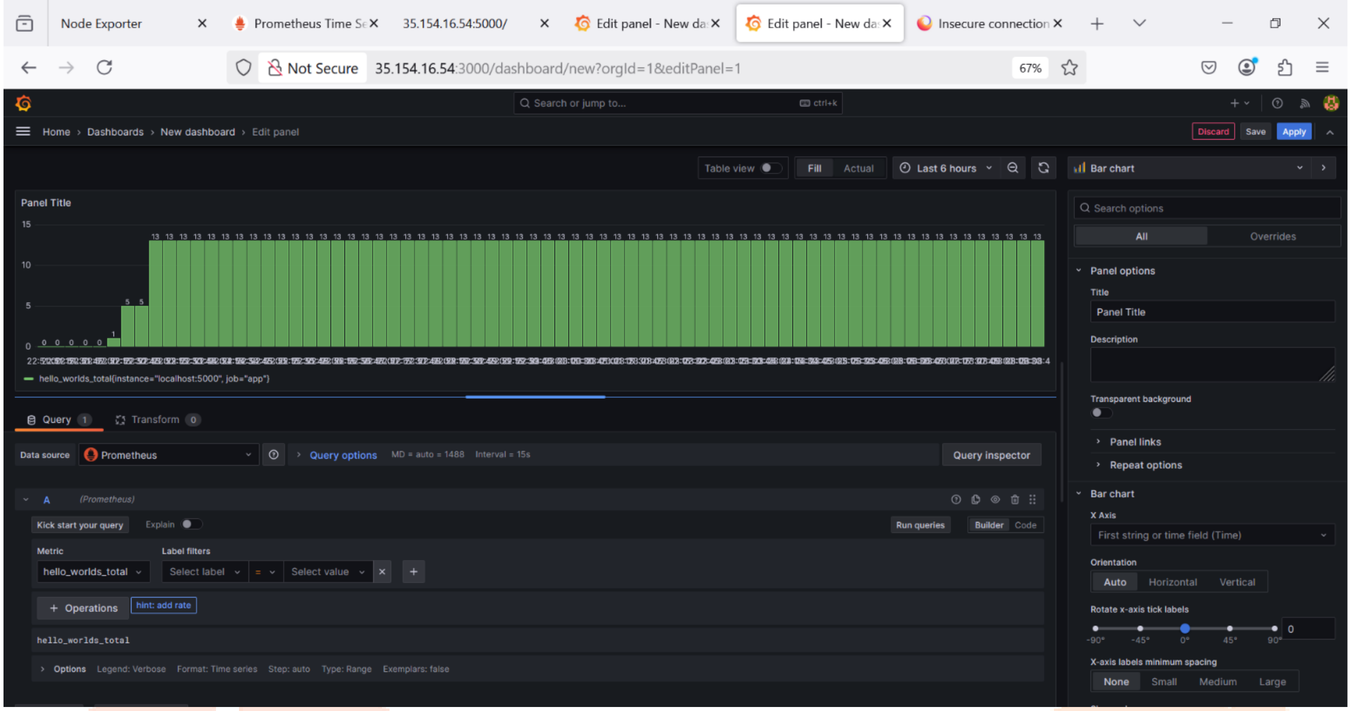
Grafana installation

```
sudo yum install -y https://dl.grafana.com/oss/release/grafana-10.0.3-1.x86_64.rpm sudo

service grafana-server start

sudo service grafana-server status
```



```
[ec2-user@ip-172-31-12-123 prometheus]$ sudo yum install -y https://dl.grafana.com/oss/release/grafana-10.0.3-1.x86_64.rpm
Prometheus                                                                                              194  B/s | 833  B     00:04
grafana-10.0.3-1.x86_64.rpm                                                                             9.8 MB/s |  79 MB     00:08
Dependencies resolved.
================================================================================================================================================
 Package                                           Architecture        Version                                  Repository                 Size
================================================================================================================================================
Installing:
 grafana                                           x86_64              10.0.3-1                                 @commandline                79 M
Installing dependencies:
 cairo                                             x86_64              1.18.0-4.amzn2023.0.1                    amazonlinux                718 k
 fontconfig                                        x86_64              2.13.94-2.amzn2023.0.2                   amazonlinux                273 k
 fonts-filesystem                                  noarch              1:2.0.5-12.amzn2023.0.2                  amazonlinux                9.5 k
 freetype                                          x86_64              2.13.2-5.amzn2023.0.1                    amazonlinux                423 k
 google-noto-fonts-common                          noarch              20201206-2.amzn2023.0.2                  amazonlinux                 15 k
 google-noto-sans-vf-fonts                         noarch              20201206-2.amzn2023.0.2                  amazonlinux                492 k
 graphite2                                         x86_64              1.3.14-7.amzn2023.0.2                    amazonlinux                 97 k
 harfbuzz                                          x86_64              7.0.0-2.amzn2023.0.2                     amazonlinux                873 k
 langpacks-core-font-en                            noarch              3.0-21.amzn2023.0.4                      amazonlinux                 10 k
 libX11                                            x86_64              1.8.10-2.amzn2023.0.1                    amazonlinux                659 k
 libX11-common                                     noarch              1.8.10-2.amzn2023.0.1                    amazonlinux                147 k
 libXau                                            x86_64              1.0.11-6.amzn2023.0.1                    amazonlinux                 33 k
 libXext                                           x86_64              1.3.6-1.amzn2023.0.1                     amazonlinux                 42 k
 libXrender                                        x86_64              0.9.11-6.amzn2023.0.1                    amazonlinux                 29 k
 libbrotli                                         x86_64              1.0.9-4.amzn2023.0.2                     amazonlinux                315 k
 libpng                                            x86_64              2:1.6.37-10.amzn2023.0.6                 amazonlinux                128 k
 libxcb                                            x86_64              1.17.0-1.amzn2023.0.1                    amazonlinux                235 k
 pixman                                            x86_64              0.43.4-1.amzn2023.0.4                    amazonlinux                296 k
 urw-base35-bookman-fonts                          noarch              20200910-6.amzn2023.0.2                  amazonlinux                848 k
 urw-base35-c059-fonts                             noarch              20200910-6.amzn2023.0.2                  amazonlinux                875 k
 urw-base35-d050000l-fonts                         noarch              20200910-6.amzn2023.0.2                  amazonlinux                 76 k
 urw-base35-fonts                                  noarch              20200910-6.amzn2023.0.2                  amazonlinux                 11 k
 urw-base35-fonts-common                           noarch              20200910-6.amzn2023.0.2                  amazonlinux                 21 k
 urw-base35-gothic-fonts                           noarch              20200910-6.amzn2023.0.2                  amazonlinux                643 k
 urw-base35-nimbus-mono-ps-fonts                   noarch              20200910-6.amzn2023.0.2                  amazonlinux                795 k
 urw-base35-nimbus-roman-fonts                     noarch              20200910-6.amzn2023.0.2                  amazonlinux                857 k
```





35.154.16.54:3000/dashboard/new?orgId=1&editPanel=1

hello_worlds_total{instance="localhost:5000", job="app"}

Data source: Prometheus

Query options MD = auto = 1488 Interval = 15s

A (Prometheus)

Metric: hello_worlds_total

Label filters: Select label = Select value

```
hello_worlds_total
```

Options Legend: Verbose Format: Time series Step: auto Type: Range Exemplars: false

Title: Panel Title

Description:

Transparent background

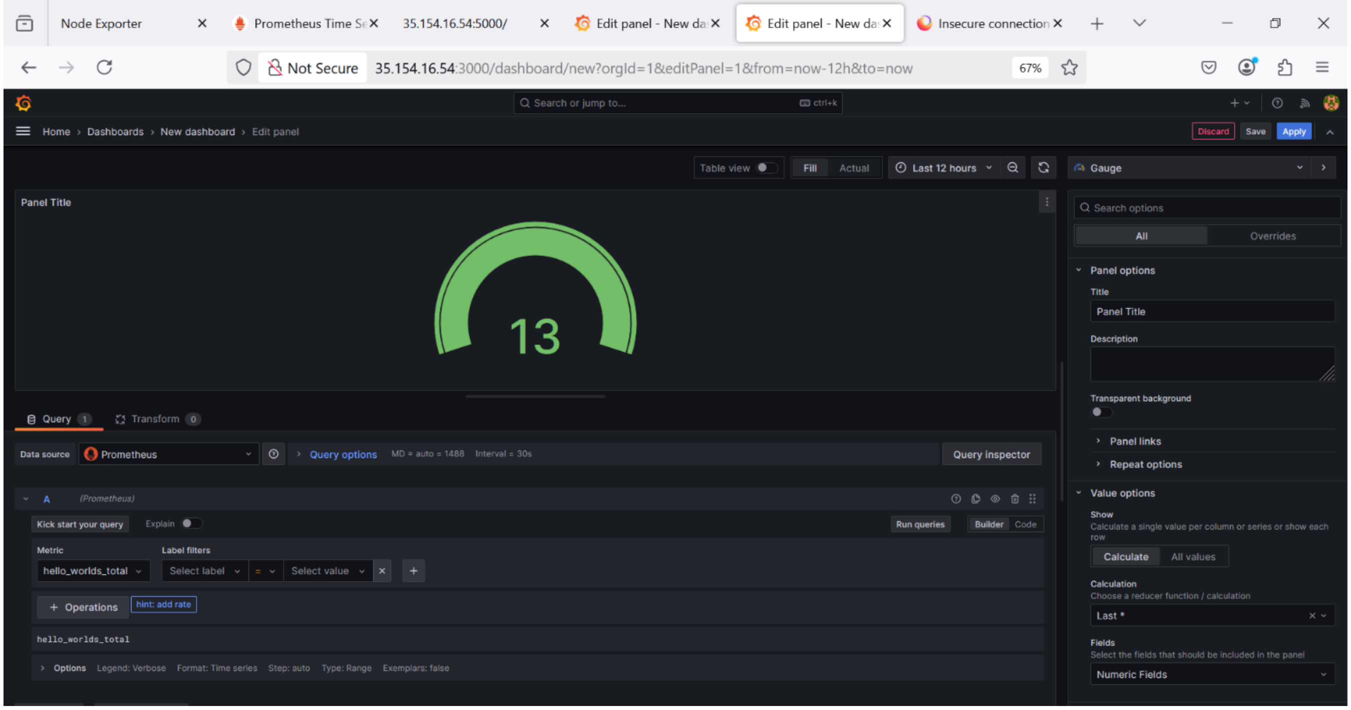
Bar chart

X Axis: First string or time field (Time)

Orientation: Auto Horizontal Vertical

Rotate x-axis tick labels: -90° -45° 0° 45° 90° 0

X-axis labels minimum spacing: None Small Medium Large



35.154.16.54:3000/dashboard/new?orgId=1&editPanel=1&from=now-12h&to=now

Data source: Prometheus

Query options MD = auto = 1488 Interval = 30s

A (Prometheus)

Metric: hello_worlds_total

Label filters: Select label = Select value

```
hello_worlds_total
```

Options Legend: Verbose Format: Time series Step: auto Type: Range Exemplars: false

Title: Panel Title

Description:

Transparent background

Value options

Show: Calculate a single value per column or series or show each row — Calculate All values

Calculation: Choose a reducer function / calculation — Last *

Fields: Select the fields that should be included in the panel — Numeric Fields