Computer Engineering &
Software Systems

Ain Shams University
Faculty of Engineering

Network Management System
Project Report

Under Supervision of

Dr. Wagdy Anis Aziz

&

Eng. John Soliman

Submitted By:

Ahmed Khaled Saad Ali Mekheimer
ID: 1809799
&
Ahmad Mohamed El-Sayed Barakat
ID: 18P9613

Table of Contents

Objective: implement a Network management system using Prometheus on multiple AWS EC2 instances

1- **Install Prometheus server on one EC2 instance.**
2- **Install Prometheus node-exporter on multiple EC2 instances.**
3- **Install Prometheus alerts system on the server EC2 instance.**
4- **Display Prometheus portal with KPIs (CPUs/MEM usage), and EC2 instances (Targets) monitoring.**

What is Prometheus?

The Prometheus Network Management System is an open-source monitoring and alerting system designed to collect and analyze metrics from various sources in real-time. It was created by the SoundCloud development team in 2012 and is now a part of the Cloud Native Computing Foundation (CNCF).

At its core, Prometheus works by scraping metrics from instrumented targets (e.g., servers, applications, containers, or devices) using a pull model. These targets expose their metrics via HTTP endpoints, which Prometheus can query periodically to gather data.

Once the data is collected, Prometheus stores it in a time-series database that allows for querying and analysis. Users can create complex queries using the PromQL (Prometheus Query Language) to filter, aggregate, and transform the collected data.

Prometheus also provides a built-in alerting system that can be configured to trigger notifications when certain thresholds are exceeded. This allows operators to proactively identify and address issues before they impact the end-user experience.

In addition to its core functionality, Prometheus has a large ecosystem of plugins and integrations that extend its capabilities. For example, Grafana can be used to visualize and explore Prometheus metrics, and exporters can be used to collect metrics from third-party systems.

Components of a Prometheus system

The most basic Prometheus system consists of 3 components

1- Prometheus server: the central node that collects data from all other nodes and is responsible for organizing that data and creating alerts if any node was to misbehave or go down
2- Prometheus targets: branch nodes that are running system critical programs for our cloud application that we would like to collect data on their performance and uptime
3- Prometheus node exporters: Exporters are software components that expose data to Prometheus. They run on the same host as the monitored application and provide metrics data.

Benefits of network management systems

A network management system (NMS) such as Prometheus provides a comprehensive view of an organization's network infrastructure, allowing network administrators to monitor network performance, identify issues, and take corrective action. The benefits of an NMS include increased network availability, faster troubleshooting, and improved security. With an NMS in place, organizations can proactively manage their network infrastructure, reduce downtime, and ensure optimal performance.

Why should we use Prometheus?

We should use Prometheus over other NMSes because:

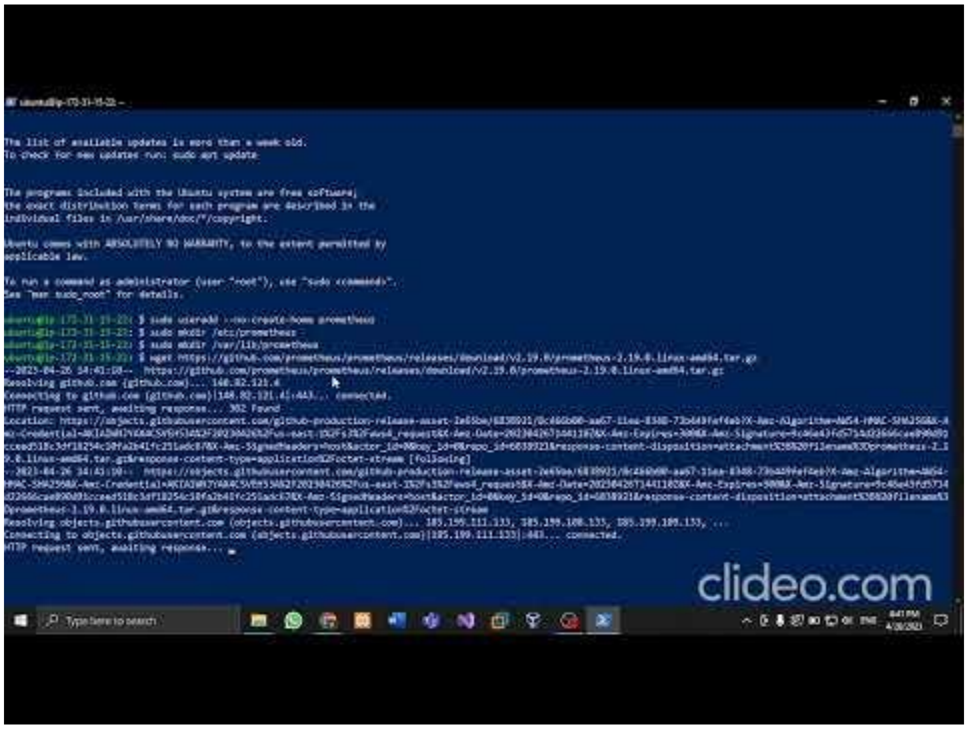
1- It is open source and free to use with highly customizable features.
2- It is highly scalable and able to handle large and complex networks.
3- It is designed to be highly available and fault tolerant.
4- It provides a powerful query language (PromQL) to analyze its collected data.
5- It has a large and active community of users that contribute to its codebase with new features.

After we have learned about Prometheus and its key benefits, we will now follow 4 links to install Prometheus, create Prometheus Server, create a node exporter, use them for Service Discovery and Alert management.

1st link: Install Prometheus on AWS EC2

https://codewizardly.com/prometheus-on-aws-ec2-part1/

Step-by-Step demo video and showing output, Click Image below.



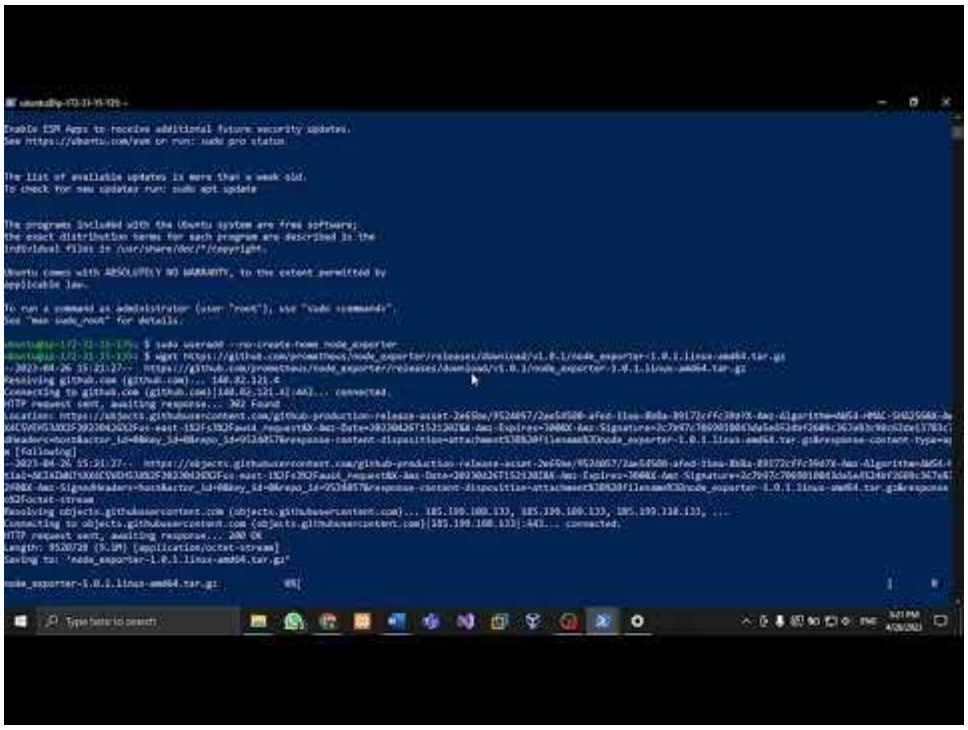
```
The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-15-22:~$ sudo useradd --no-create-home prometheus
ubuntu@ip-172-31-15-22:~$ sudo mkdir /etc/prometheus
ubuntu@ip-172-31-15-22:~$ sudo mkdir /var/lib/prometheus
ubuntu@ip-172-31-15-22:~$ wget https://github.com/prometheus/prometheus/releases/download/v2.19.0/prometheus-2.19.0.linux-amd64.tar.gz
--2023-04-26 14:41:10--  https://github.com/prometheus/prometheus/releases/download/v2.19.0/prometheus-2.19.0.linux-amd64.tar.gz
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/[illegible] [following]
--2023-04-26 14:41:10--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/[illegible]
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response...
```

2nd Link: Prometheus Node Exporter on AWS EC2

https://codewizardly.com/prometheus-on-aws-ec2-part2/

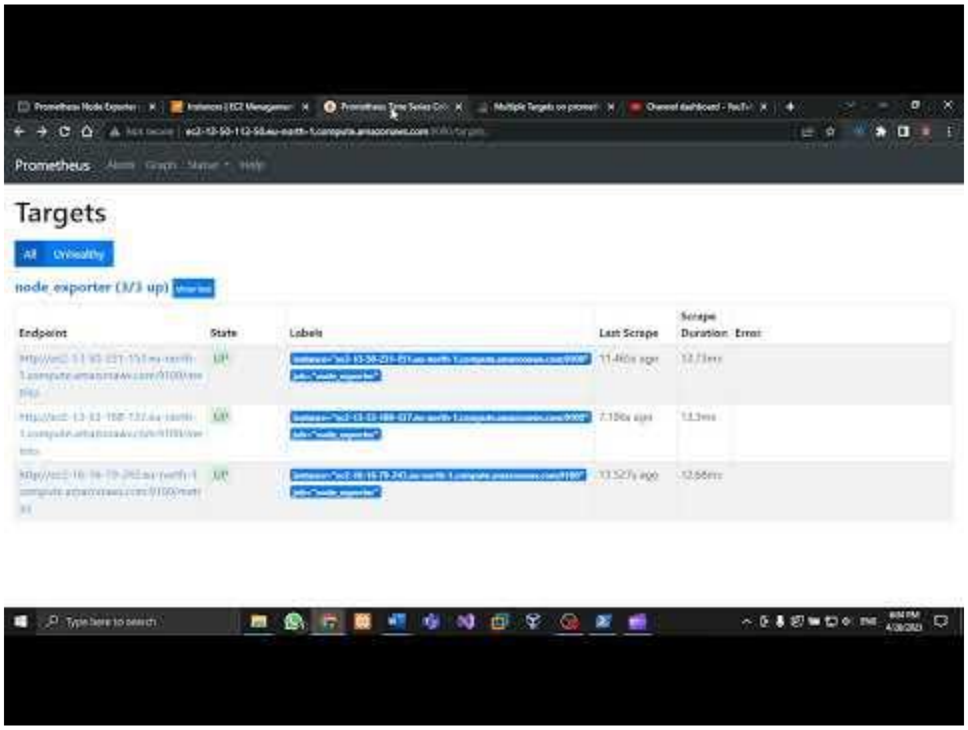
Following this link, we will launch a second AWS EC2 instance, install Prometheus Node Exporter in it and finally, configure Prometheus to collect its metrics.

Step-by-Step demo video and showing output, Click Image below.



```
ubuntu@ip-172-31-15-125: ~

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-15-125:~$ sudo useradd --no-create-home node_exporter
ubuntu@ip-172-31-15-125:~$ wget https://github.com/prometheus/node_exporter/releases/download/v1.0.1/node_exporter-1.0.1.linux-amd64.tar.gz
--2023-04-26 15:21:27--  https://github.com/prometheus/node_exporter/releases/download/v1.0.1/node_exporter-1.0.1.linux-amd64.tar.gz
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/[illegible]
[illegible]
[illegible]node_exporter-1.0.1.linux-amd64.tar.gz&response-content-type=[illegible]
m [following]
--2023-04-26 15:21:27--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/[illegible]
[illegible]
[illegible]node_exporter-1.0.1.linux-amd64.tar.gz&response[illegible]
%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9520728 (9.1M) [application/octet-stream]
Saving to: 'node_exporter-1.0.1.linux-amd64.tar.gz'

node_exporter-1.0.1.linux-amd64.tar.gz          0%[
```

Since we will need multiple nodes, 2 more Nodes were created following same steps of 2nd Link. Click Image below.



Prometheus

Targets

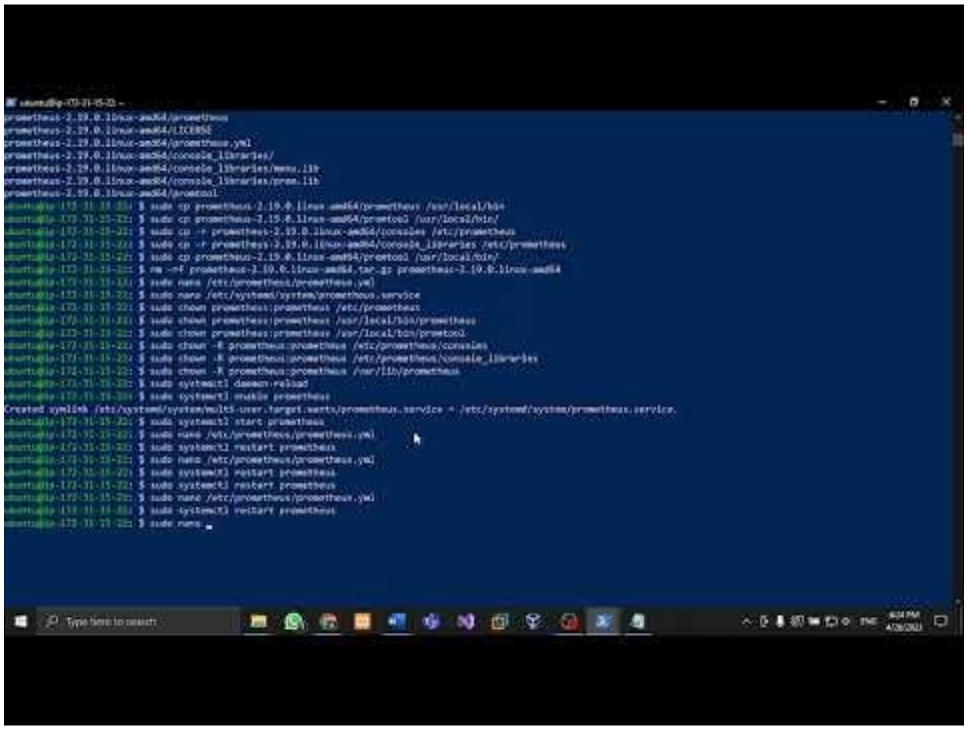
node_exporter (3/3 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
[illegible]	UP	[illegible] job="node_exporter"	11.460s ago	12.73ms	
[illegible]	UP	[illegible] job="node_exporter"	7.106s ago	13.3ms	
[illegible]	UP	[illegible] job="node_exporter"	13.527s ago	12.68ms	

3rd Link: Prometheus Service Discovery on AWS EC2

https://codewizardly.com/prometheus-on-aws-ec2-part3/

Step-by-Step demo video and showing output, Click Image below.

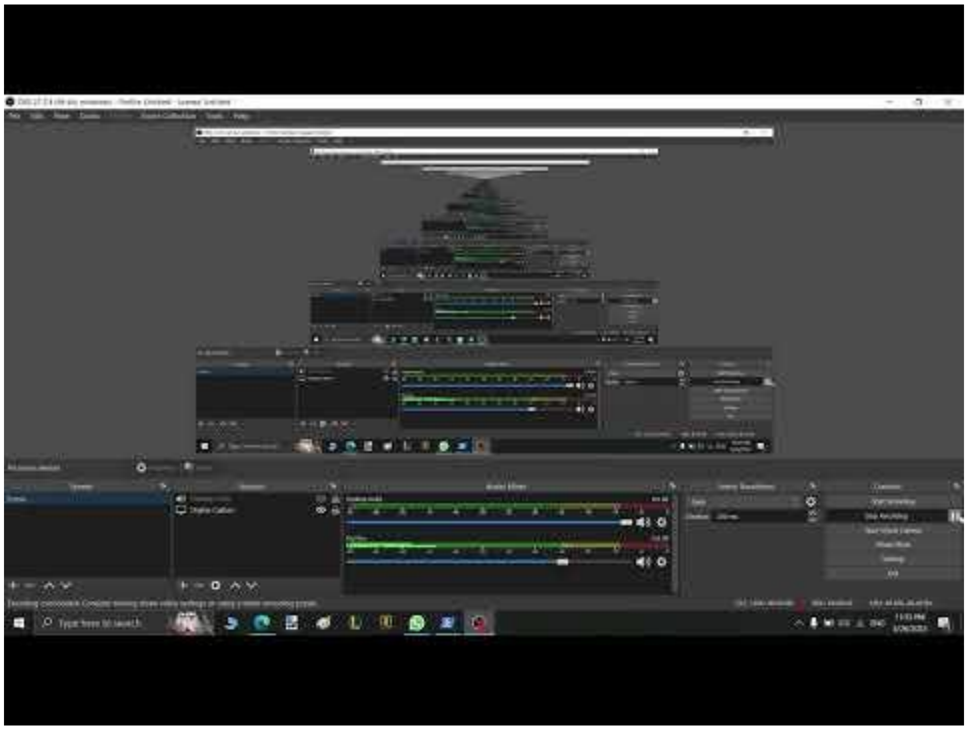


```
ubuntu@ip-172-31-15-22: ~
prometheus-2.19.0.linux-amd64/prometheus
prometheus-2.19.0.linux-amd64/LICENSE
prometheus-2.19.0.linux-amd64/prometheus.yml
prometheus-2.19.0.linux-amd64/console_libraries/
prometheus-2.19.0.linux-amd64/console_libraries/menu.lib
prometheus-2.19.0.linux-amd64/console_libraries/prom.lib
prometheus-2.19.0.linux-amd64/promtool
ubuntu@ip-172-31-15-22:~$ sudo cp prometheus-2.19.0.linux-amd64/prometheus /usr/local/bin
ubuntu@ip-172-31-15-22:~$ sudo cp prometheus-2.19.0.linux-amd64/promtool /usr/local/bin/
ubuntu@ip-172-31-15-22:~$ sudo cp -r prometheus-2.19.0.linux-amd64/consoles /etc/prometheus
ubuntu@ip-172-31-15-22:~$ sudo cp -r prometheus-2.19.0.linux-amd64/console_libraries /etc/prometheus
ubuntu@ip-172-31-15-22:~$ sudo cp prometheus-2.19.0.linux-amd64/promtool /usr/local/bin/
ubuntu@ip-172-31-15-22:~$ rm -rf prometheus-2.19.0.linux-amd64.tar.gz prometheus-2.19.0.linux-amd64
ubuntu@ip-172-31-15-22:~$ sudo nano /etc/prometheus/prometheus.yml
ubuntu@ip-172-31-15-22:~$ sudo nano /etc/systemd/system/prometheus.service
ubuntu@ip-172-31-15-22:~$ sudo chown prometheus:prometheus /etc/prometheus
ubuntu@ip-172-31-15-22:~$ sudo chown prometheus:prometheus /usr/local/bin/prometheus
ubuntu@ip-172-31-15-22:~$ sudo chown prometheus:prometheus /usr/local/bin/promtool
ubuntu@ip-172-31-15-22:~$ sudo chown -R prometheus:prometheus /etc/prometheus/consoles
ubuntu@ip-172-31-15-22:~$ sudo chown -R prometheus:prometheus /etc/prometheus/console_libraries
ubuntu@ip-172-31-15-22:~$ sudo chown -R prometheus:prometheus /var/lib/prometheus
ubuntu@ip-172-31-15-22:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-15-22:~$ sudo systemctl enable prometheus
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
ubuntu@ip-172-31-15-22:~$ sudo systemctl start prometheus
ubuntu@ip-172-31-15-22:~$ sudo nano /etc/prometheus/prometheus.yml
ubuntu@ip-172-31-15-22:~$ sudo systemctl restart prometheus
ubuntu@ip-172-31-15-22:~$ sudo nano /etc/prometheus/prometheus.yml
ubuntu@ip-172-31-15-22:~$ sudo systemctl restart prometheus
ubuntu@ip-172-31-15-22:~$ sudo systemctl restart prometheus
ubuntu@ip-172-31-15-22:~$ sudo nano /etc/prometheus/prometheus.yml
ubuntu@ip-172-31-15-22:~$ sudo systemctl restart prometheus
ubuntu@ip-172-31-15-22:~$ sudo nano _
```

4th Link: Prometheus Alert manager:

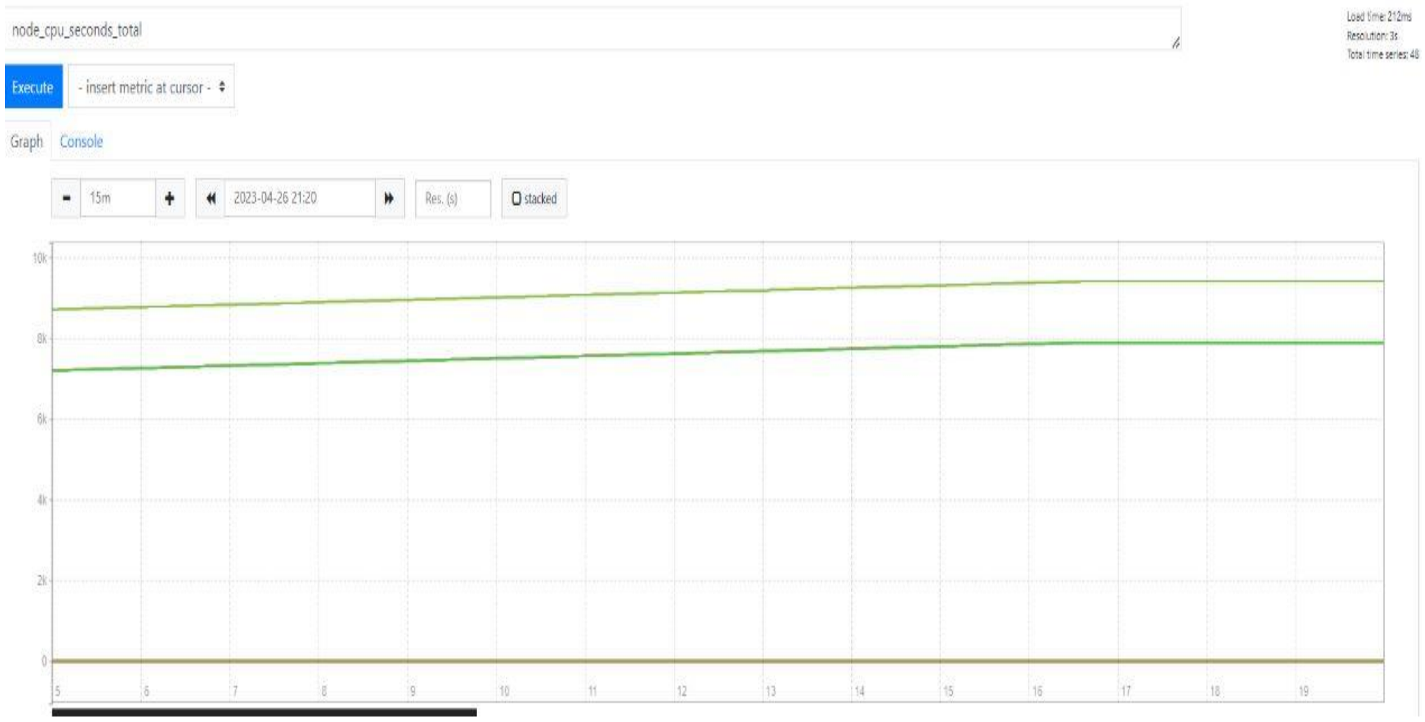
https://codewizardly.com/prometheus-on-aws-ec2-part4/

Following this link, we will show its output: Click Image below.

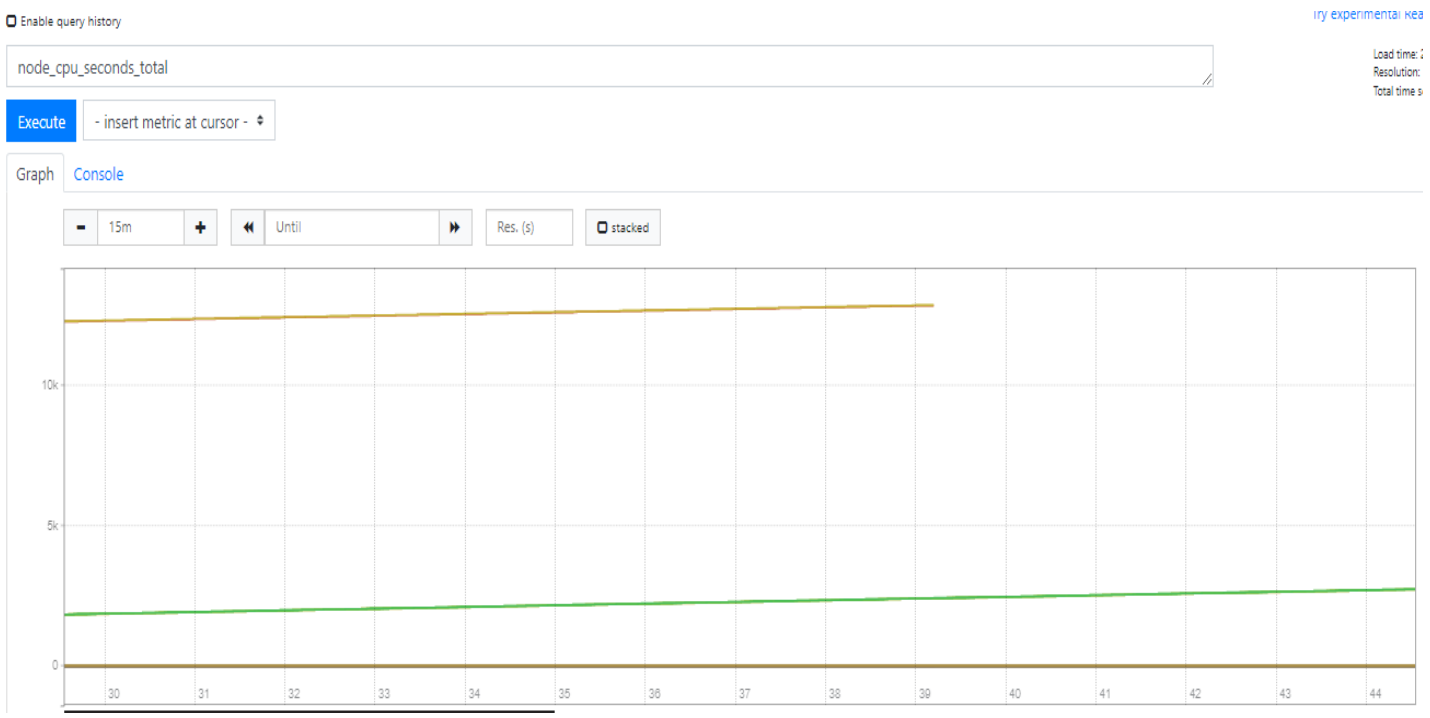


Display Prometheus portal with KPIs (CPUs/MEM usage), and EC2 instances (Targets) monitoring.

We have 3 Nodes running, So for CPU Usage we chose "node_cpu_seconds_total" graph below as it appears we have 3 lines in the graph:



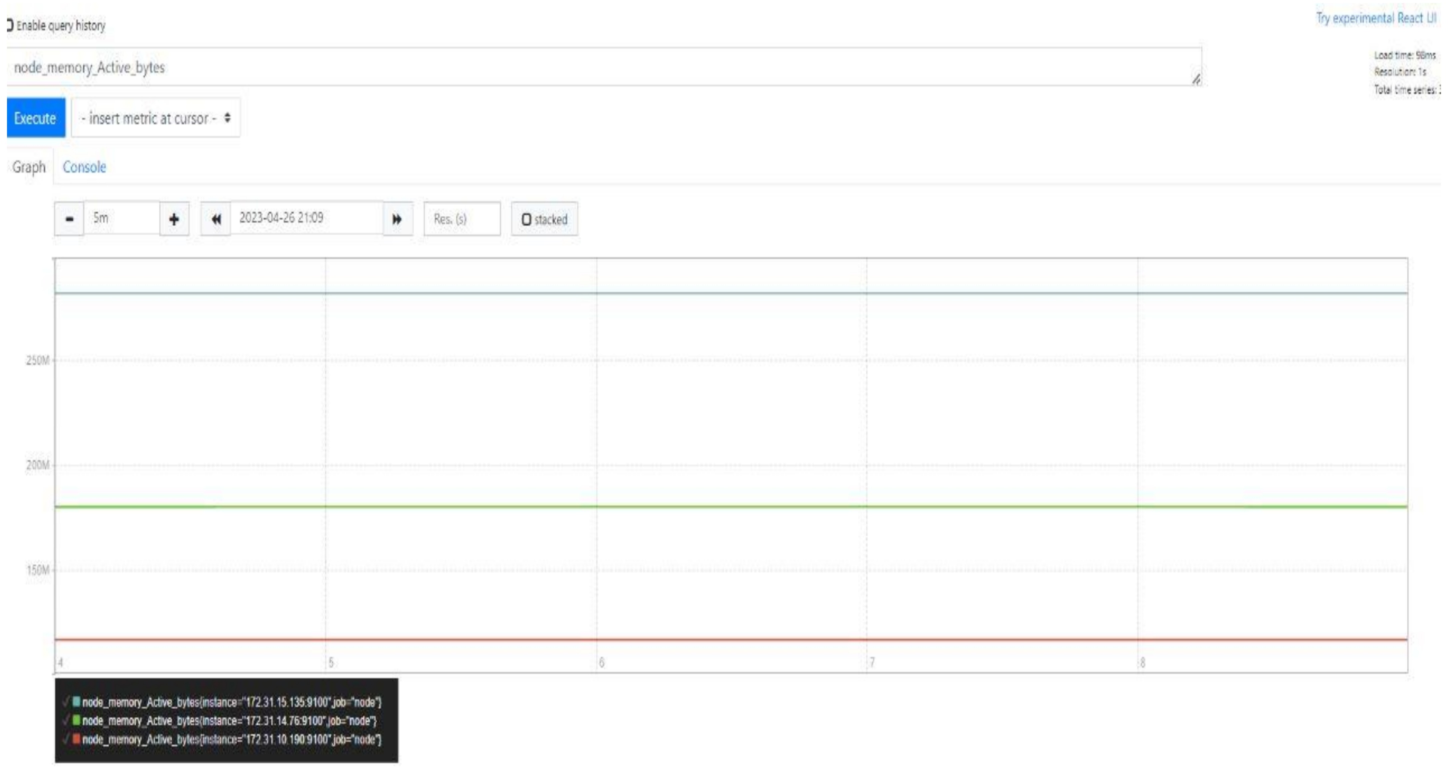
If we “Stop” one of the instances, the upper line will not continue respect to time as it is in the graph below:



Enable query history

```
node_cpu_seconds_total
```

For Memory Usage, we chose “node_memory_Active_bytes” graph below:



```
node_memory_Active_bytes
```

Load time: 98ms
Resolution: 1s
Total time series: 3

✓ 🟦 node_memory_Active_bytes{instance="172.31.15.135:9100",job="node"}
✓ 🟩 node_memory_Active_bytes{instance="172.31.14.76:9100",job="node"}
✓ 🟥 node_memory_Active_bytes{instance="172.31.10.190:9100",job="node"}

Conclusion

Node-exporter measures multiple metrics such as memory, disk space, CPU, and network traffic. By monitoring these metrics with a tool like Prometheus, you can manage distributed systems over a network, no matter how complex or large and avoid bottlenecks in the virtual or physical nodes helps avoid slow-down and outages that are difficult to diagnose at a pod or container level.