# **Day 22 Task**

## **1. Configuring Jobs and Targets**

* Task: Set up a Prometheus server to monitor multiple services running on different nodes.
* Deliverables:
  + Configure Prometheus with jobs for monitoring different services like web servers, databases, and system metrics.
* prometheus.yml

# Load rules once and periodically evaluate them according to the global 'evaluation\_interval'.

rule\_files:

# - "first\_rules.yml"

# - "second\_rules.yml"

- "alert\_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\_exporter'

static\_configs:

- targets: ['13.235.78.105:9100']

# webserver target setup

- job\_name: "nginx"

static\_configs:

- targets: ["13.235.78.105:80"]

# database target setup

- job\_name: "mysql"

static\_configs:

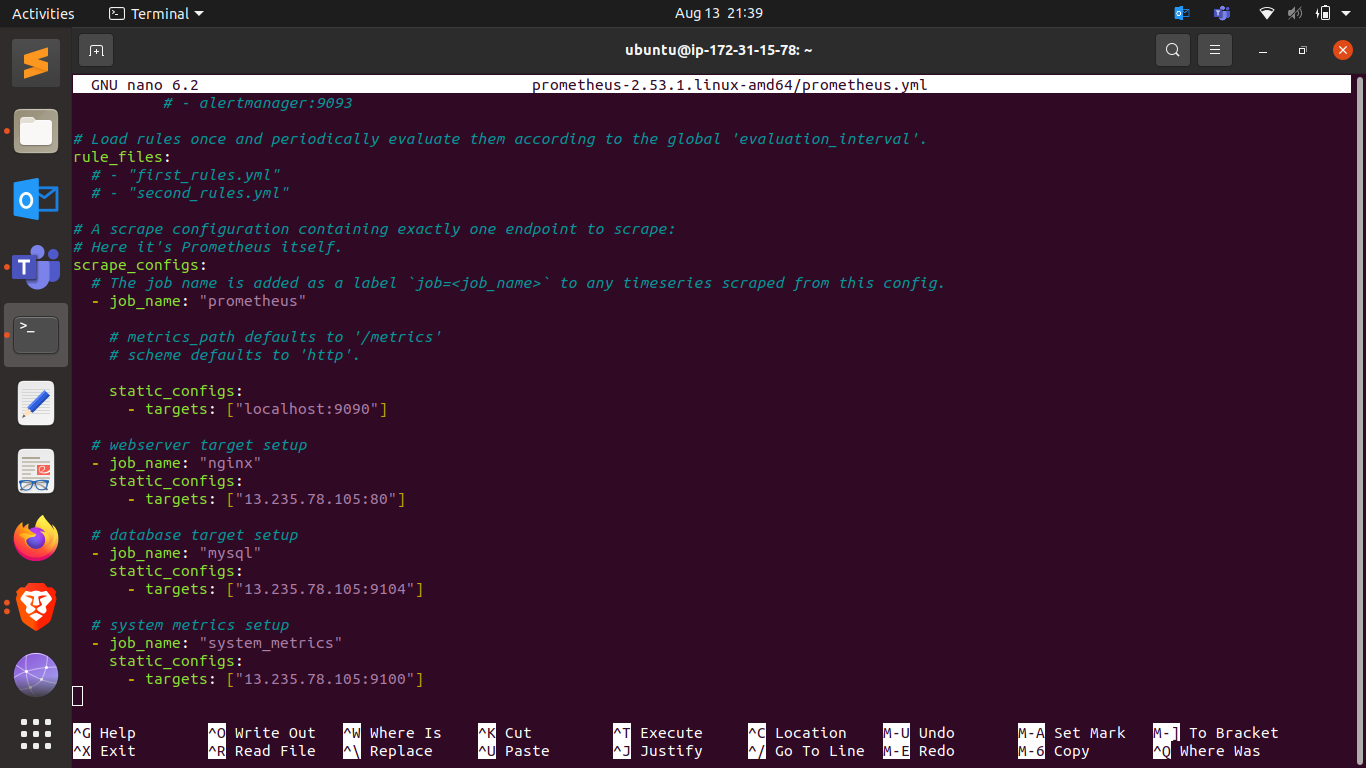
- targets: ["13.235.78.105:9104"]

# system metrics setup

- job\_name: "system\_metrics"

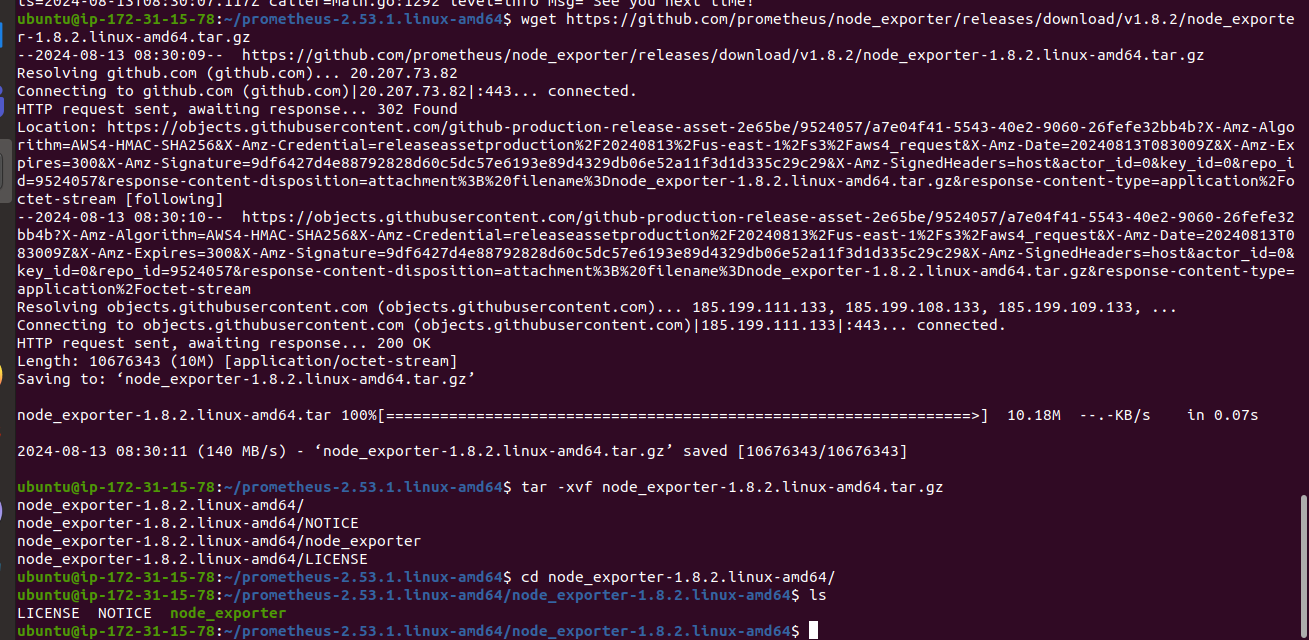
static\_configs:

- targets: ["13.235.78.105:9100"]



## **2. Using Exporters (Node Exporter)**

* **Task:** Use Node Exporter to monitor system-level metrics like CPU, memory, disk usage, and network statistics.
* Deliverables:
  + Install and configure Node Exporter on all nodes.
  + Ensure Node Exporter metrics are being correctly scraped by Prometheus.

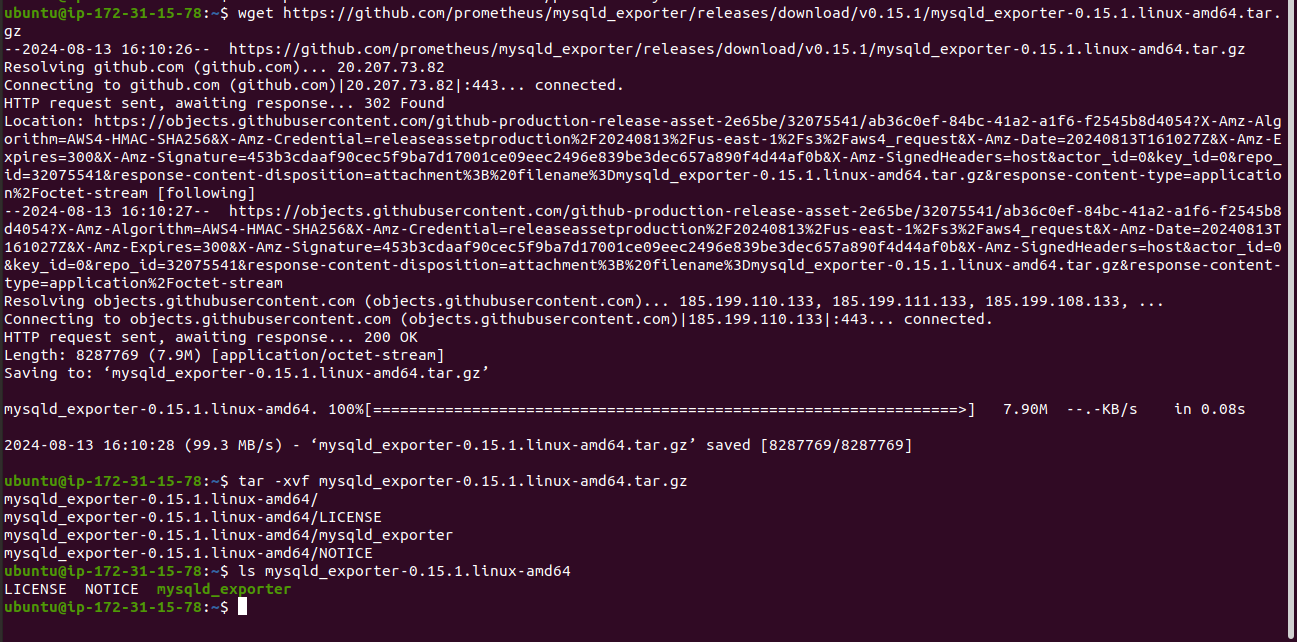


### **3. Hands-on Exercise: Setting Up Exporters**

* **Task:** Configure at least two different types of exporters (e.g., Node Exporter and MySQL Exporter) and integrate them with Prometheus.

Deliverables:

* + Demonstrate successful data collection from both exporters.
  + Create a basic Prometheus dashboard to visualize key metrics from these exporters.

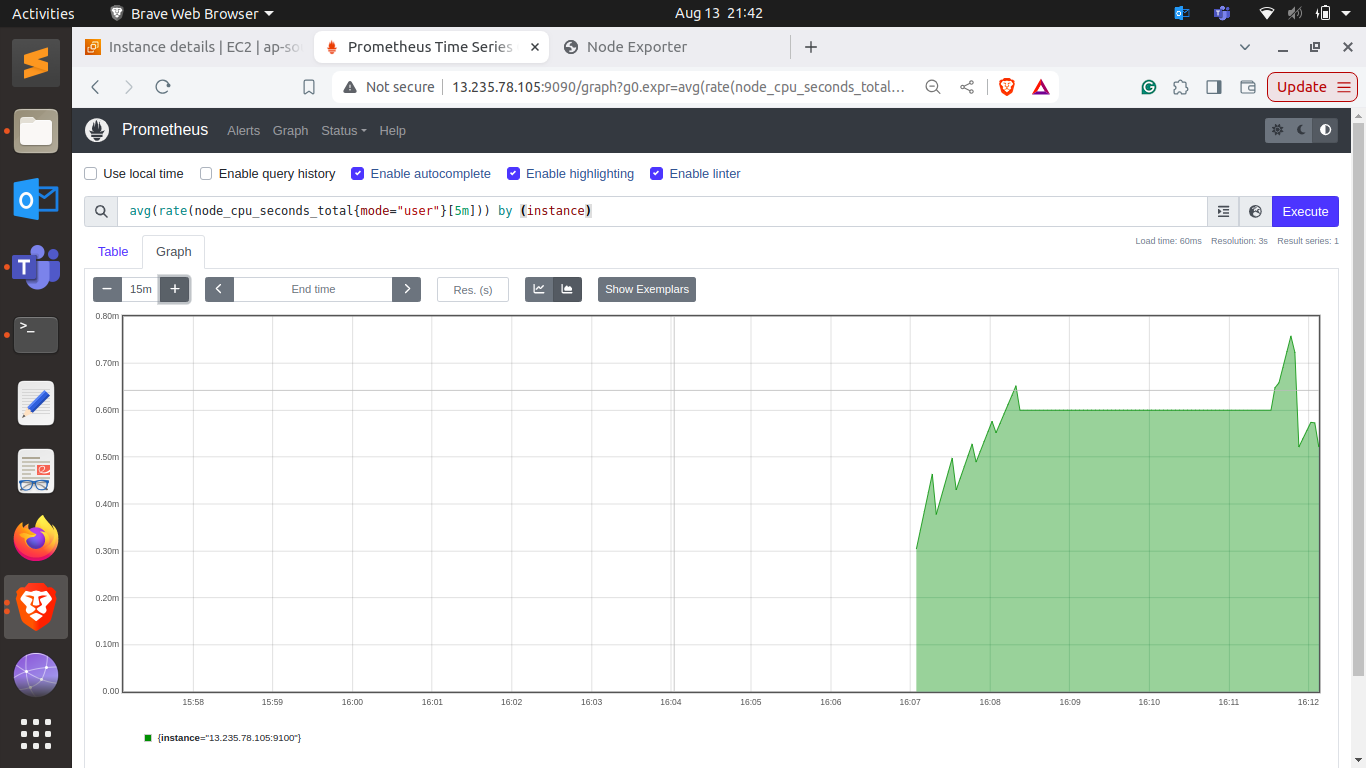


## **4. Introduction to PromQL**

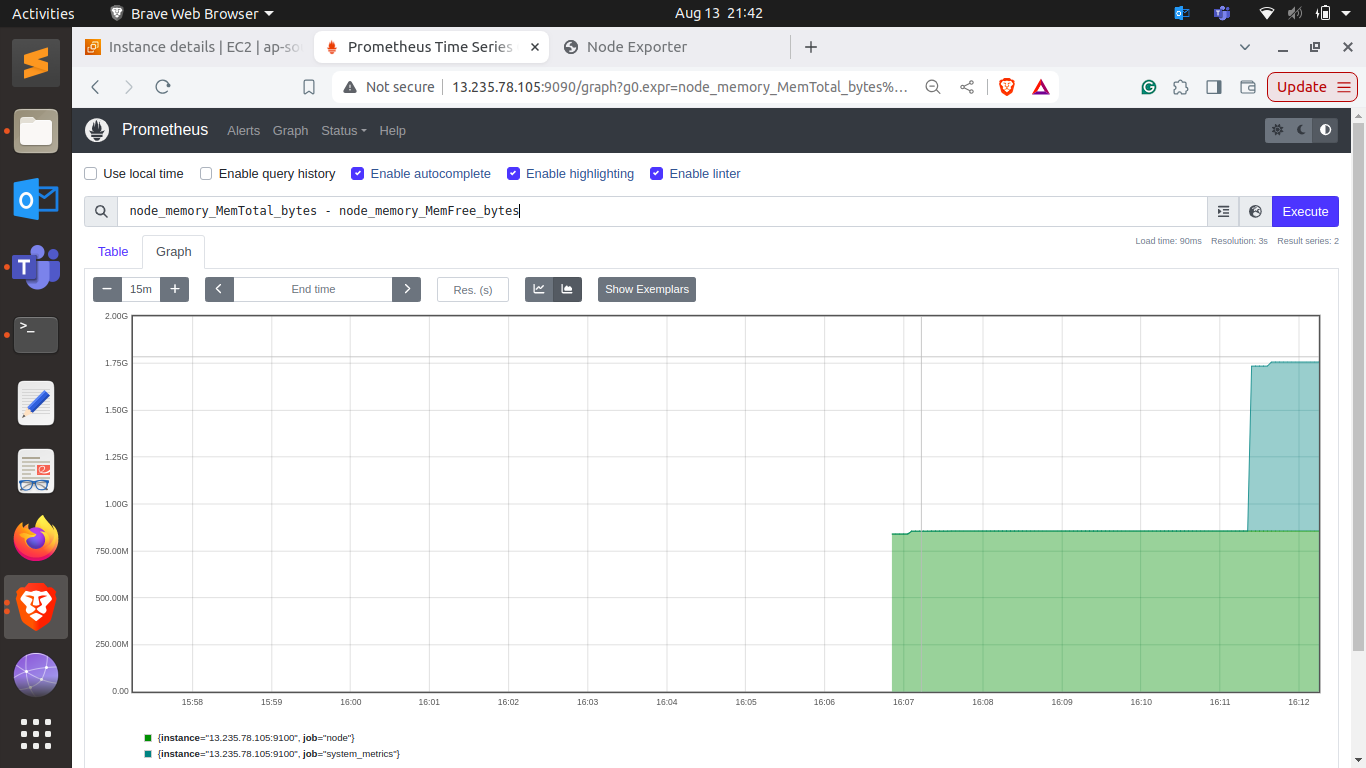
* **Task:** Learn and implement basic PromQL queries to extract meaningful data from the metrics collected.
* Deliverables:
  + Write basic queries to retrieve metrics like average CPU usage, memory consumption, and disk I/O over time.

**Graph of average CPU usage**

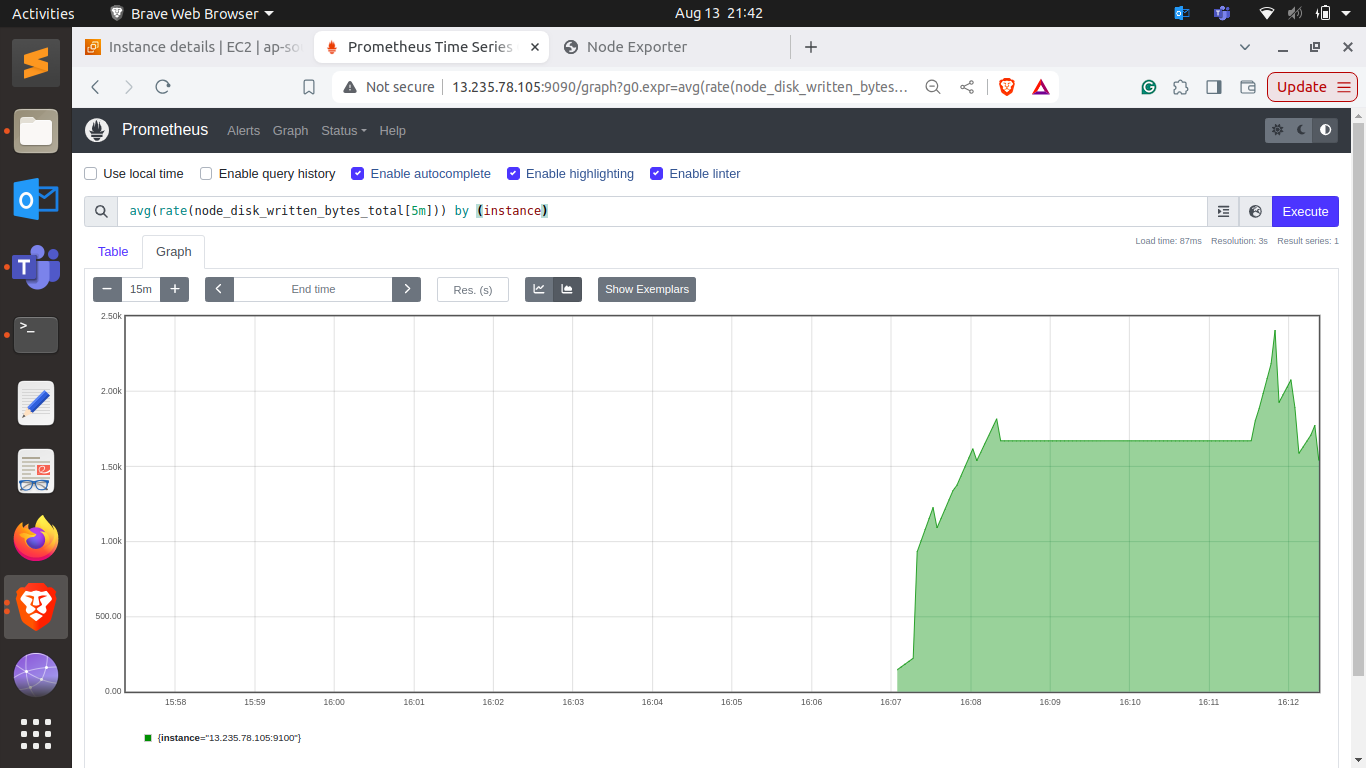
avg(rate(node\_cpu\_seconds\_total{mode="user"}[5m])) by (instance)



**Graph of memory consumption**



**Graph of disk I/O over time**



## **5. Basic Queries (Selectors, Functions, Operators)**

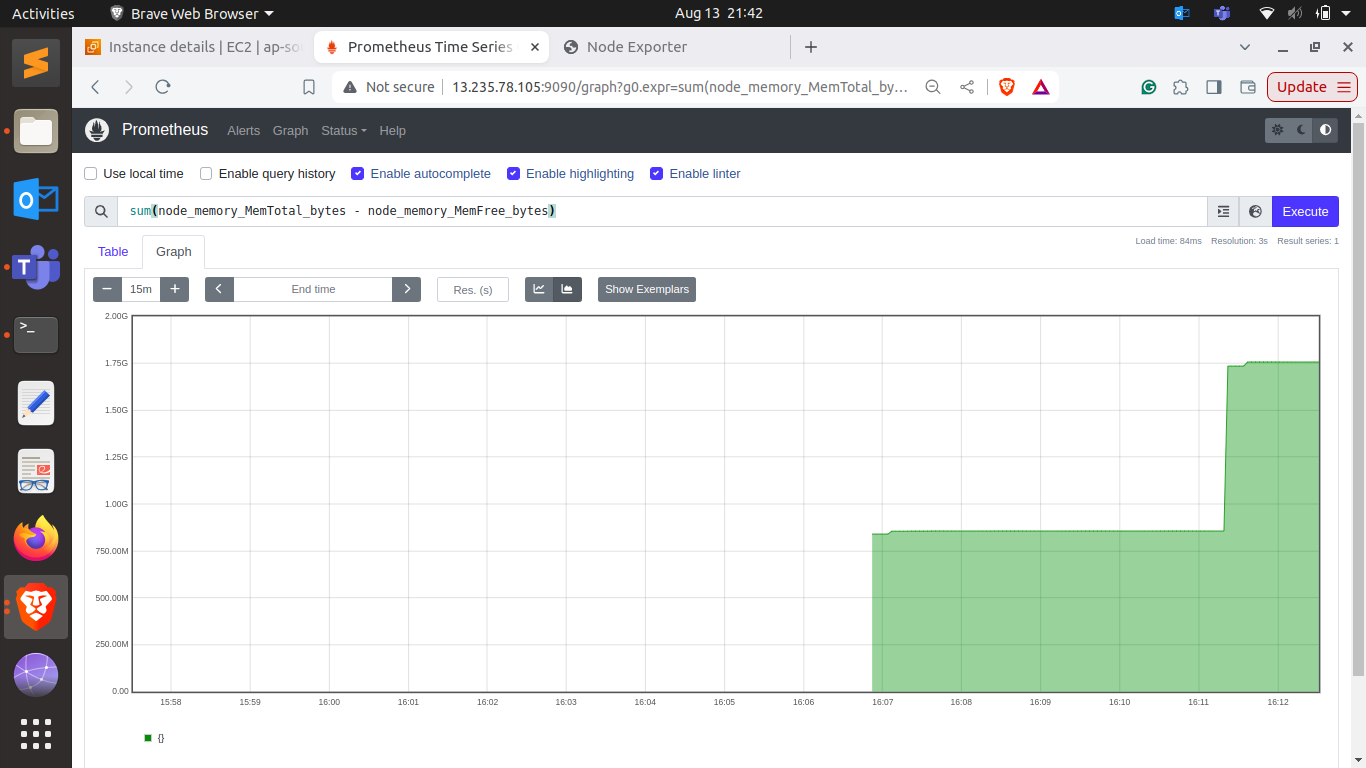
* **Task:** Create PromQL queries using selectors, functions, and operators to filter and manipulate time-series data.
* Deliverables:
  + Write PromQL queries to calculate the 95th percentile of CPU usage.
  + Use functions like rate(), increase(), and histogram\_quantile() to perform more complex analysis.

## **6. Advanced Queries and Aggregations**

* **Task:** Perform advanced data aggregation using PromQL.
* Deliverables:
  + Write queries to calculate the total memory usage across all nodes.
  + Aggregate data to find the maximum disk space usage among all nodes.

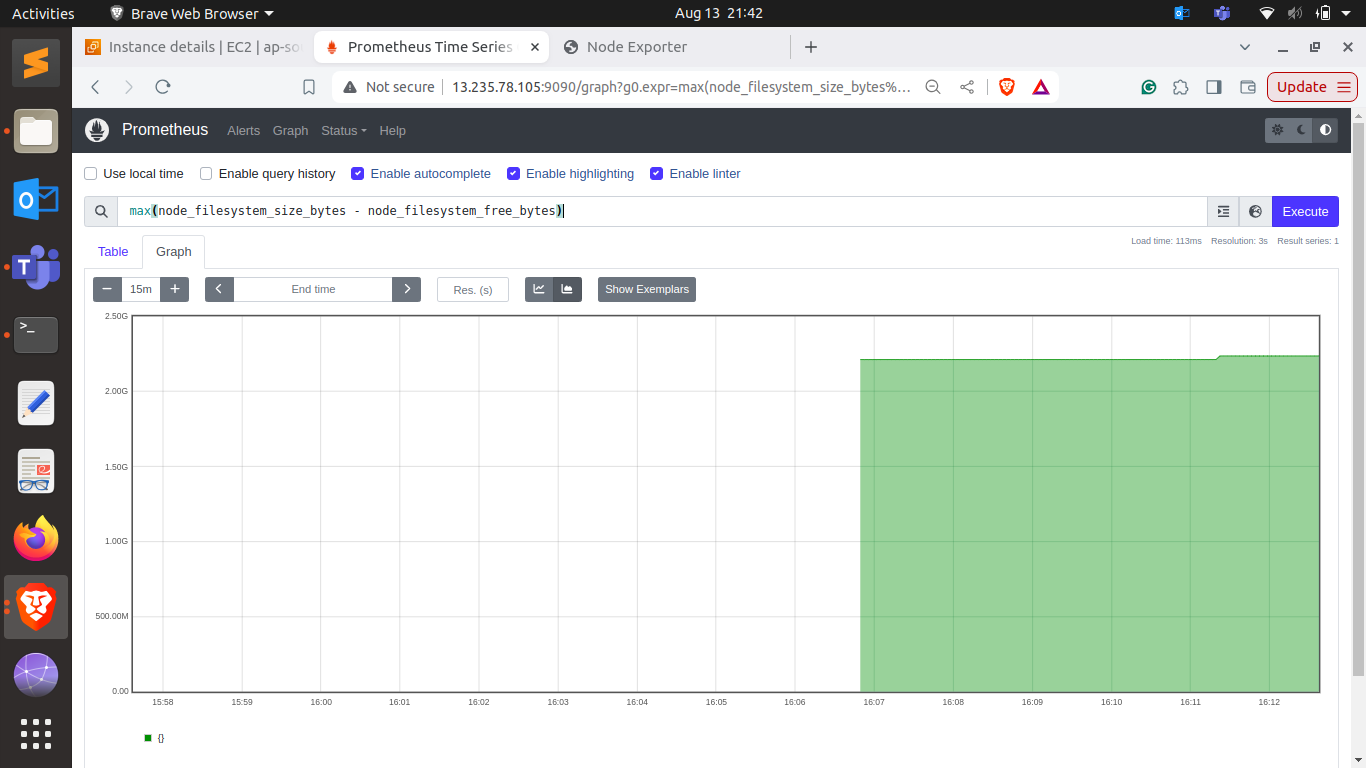
**Graph of total memory usage across all nodes**

sum(node\_memory\_MemTotal\_bytes - node\_memory\_MemFree\_bytes)



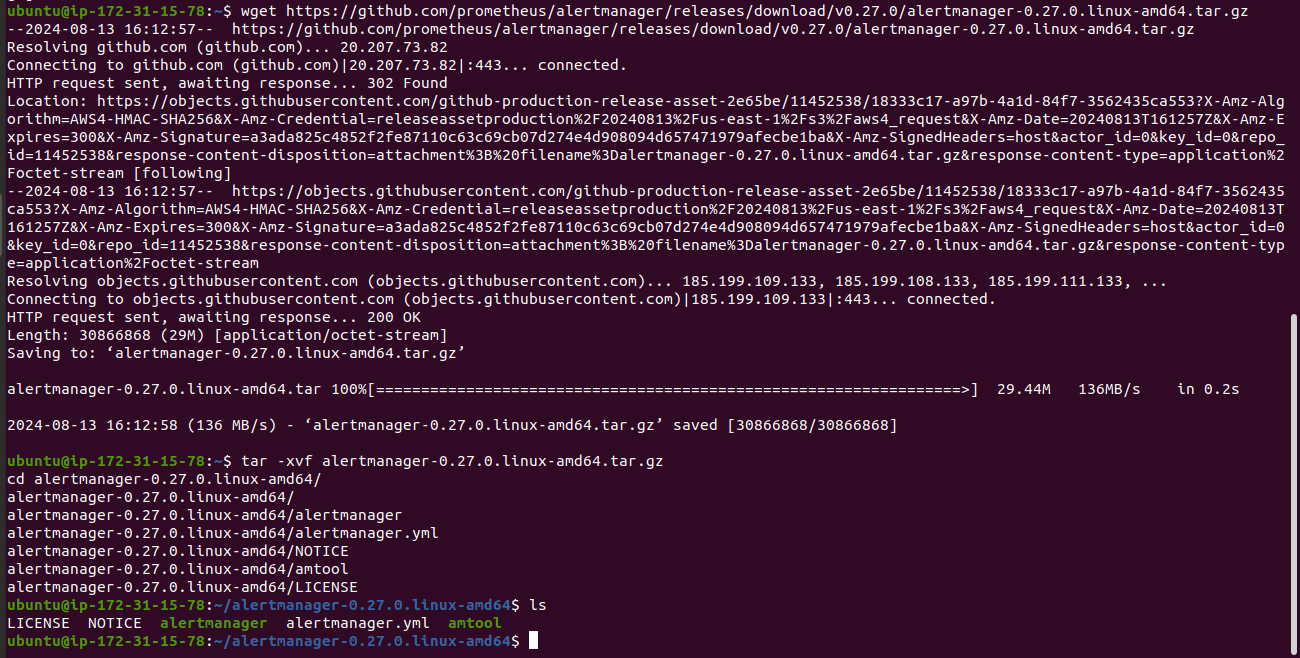
**Graph of maximum disk space usage among all nodes**

max(node\_filesystem\_size\_bytes - node\_filesystem\_free\_bytes)



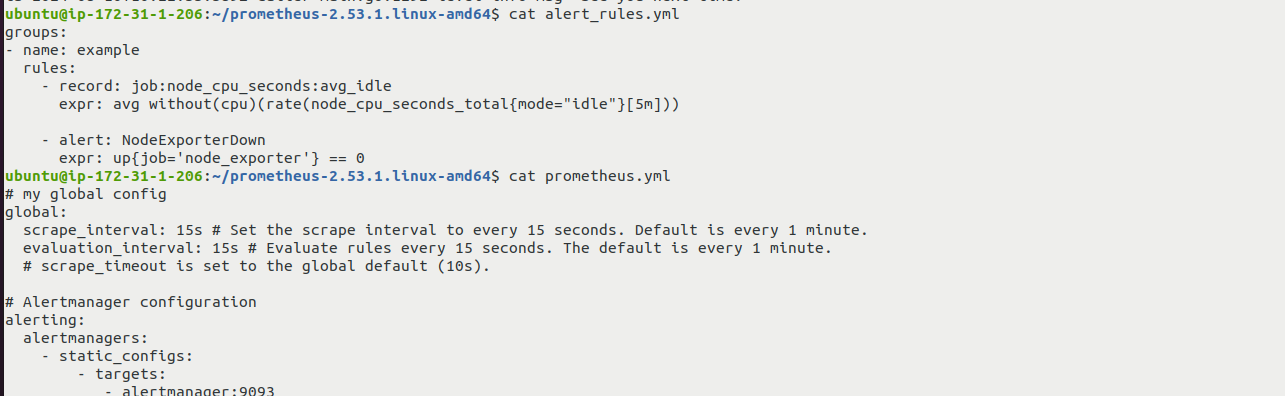
## **7. Configuring Alertmanager**

* **Task:** Set up Alertmanager to handle alerts generated by Prometheus.
* Deliverables:
  + Configure Alertmanager with Prometheus.
  + Create routing rules to manage alert notifications based on severity and service type.



## **8. Writing Alerting Rules**

* **Task:** Write custom alerting rules in Prometheus to trigger alerts based on specific conditions.
* Deliverables:
  + Create alerting rules for high CPU usage, memory leaks, and disk space running low.
  + Ensure alerts are correctly generated and sent to Alertmanager.

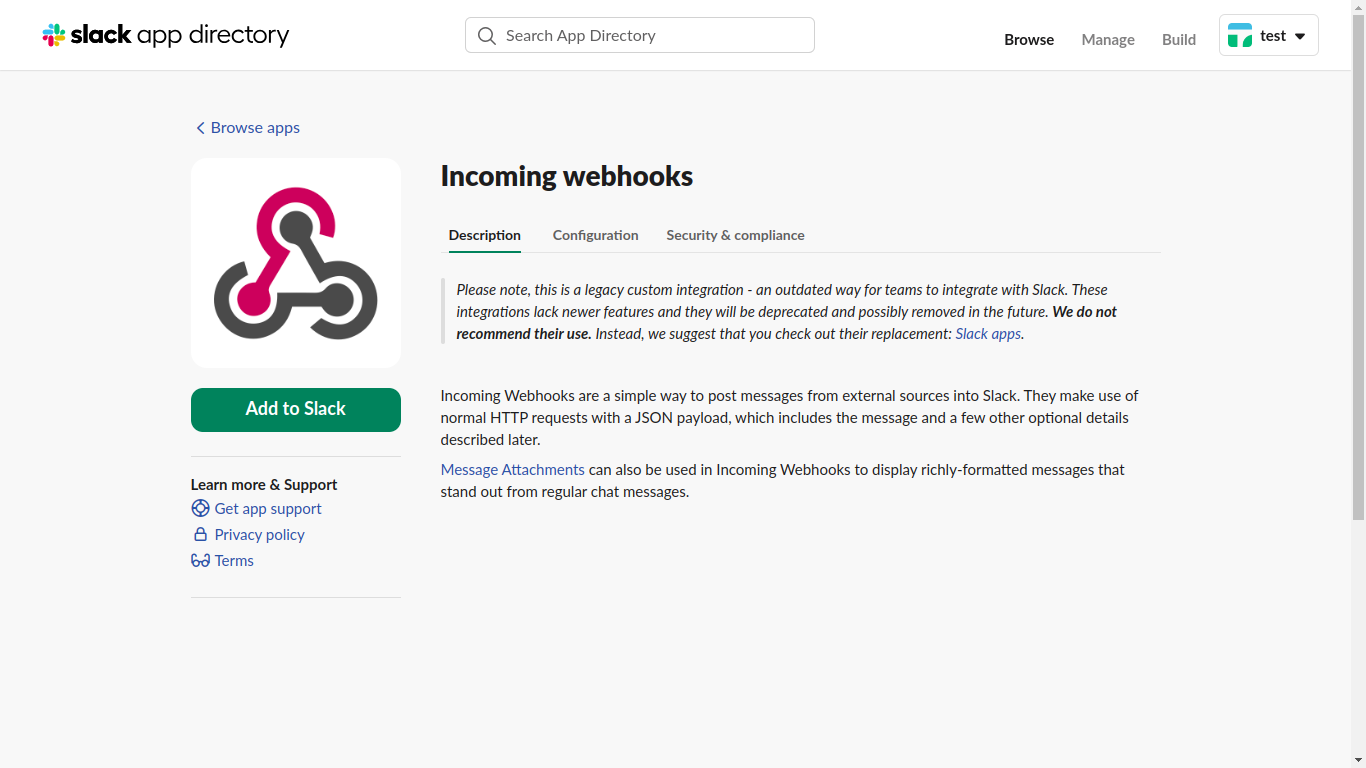


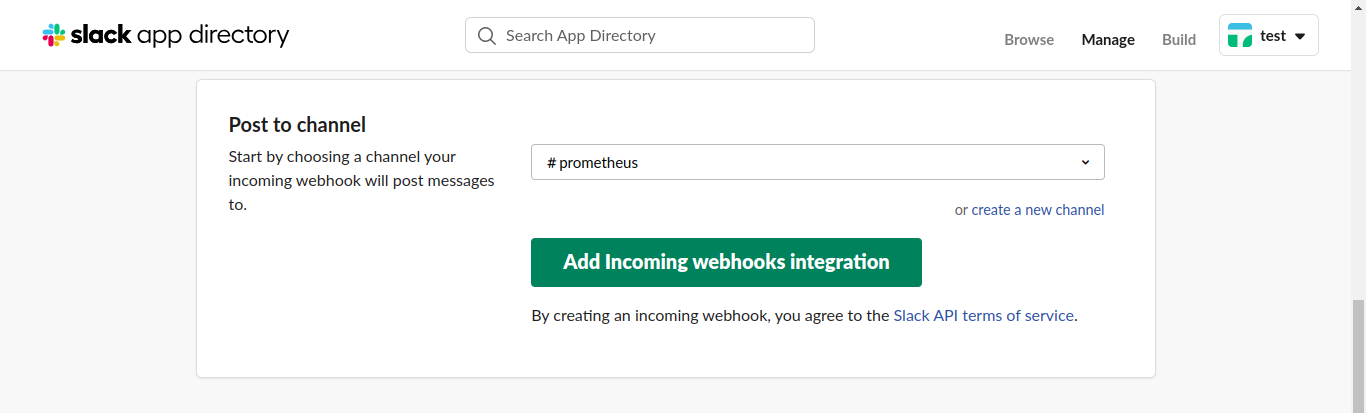
## **9. Setting Up Notification Channels (Email, Slack, etc.)**

* **Task:** Integrate Alertmanager with multiple notification channels and Slack.
* Deliverables:
  + Set up Email notifications for critical alerts.
  + Integrate Slack for real-time alerts and notifications.

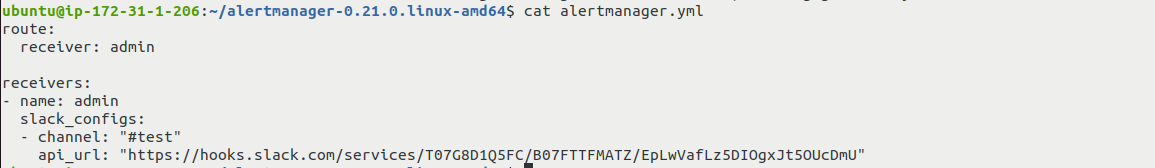
**Creating Prometheus Channel on Slack**

**Adding Incoming webhooks**





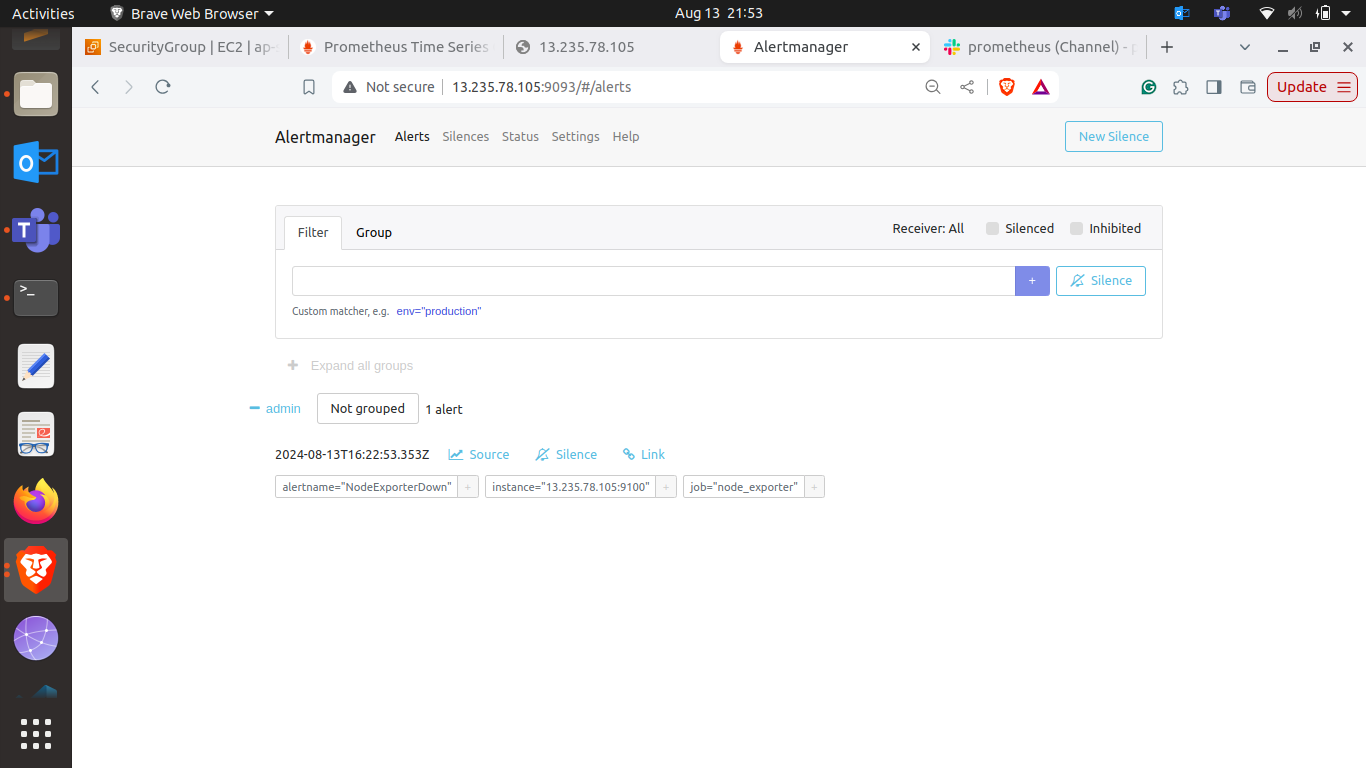
Setting up alertmanager.yml



## **10. Hands-on Exercise: Creating Alerts**

* **Task:** Test the entire alerting pipeline by creating and triggering custom alerts.
* Deliverables:
  + Simulate a scenario where a node exceeds 90% CPU usage and ensure alerts are triggered and sent to both Email and Slack.
  + Validate the alerts in both notification channels.

Alert recieved on Alert Manager and forwarder to slack



Alert shown in slack on prometheus channel

