# Báo cáo

***Sinh viên thực hiện****: Trương Công Toàn*

***Đề tài*** *: Tìm hiểu về Prometheus và Grafana*

***Mô tả****: Tìm hiểu về Prometheus và Grafana, các chức năng của chúng và tiến hành cài đặt cũng như demo ví dụ với Prometheus và Grafana*

Mục lục

[Tiêu đề 1](#_Toc87146417)

[Mô tả 1](#_Toc87146418)

[**I.** **Giới thiệu về Prometheus** 2](#_Toc87146419)

[**1.** **Khái niệm** 2](#_Toc87146420)

[**1.1 Khái niệm về Prometheus:** 2](#_Toc87146421)

[**1.2 Time series database** 2](#_Toc87146422)

[**2.** **Cách thức hoạt động của Prometheus** 3](#_Toc87146423)

[**2.1. Các thành phần chính trong hệ thống Prometheus** 3](#_Toc87146424)

[**2.2.** **Cách thức hoạt động của Prometheus** 4](#_Toc87146425)

[**2.2.1. Một số tính năng của Prometheus** 5](#_Toc87146426)

[**2.2.2. Một số thuật ngữ quan trọng với Prometheus** 5](#_Toc87146427)

[**2.3. Khi nào nên áp dụng Prometheus** 6](#_Toc87146428)

[**3.** **Cài đặt Prometheus trên môi trường Linux** 7](#_Toc87146429)

[**3.1 Cài đặt Prometheus** 7](#_Toc87146430)

[**3.2 Cài đặt Node Exporter** 11](#_Toc87146431)

[**II . Tìm hiểu về Grafana** 14](#_Toc87146432)

[**1. Khái niệm** 14](#_Toc87146433)

[**2. Các tính năng của Grafana** 15](#_Toc87146434)

[**3. Cài đặt và sử dụng Grafana** 15](#_Toc87146435)

[**III.** **Kết luận** 25](#_Toc87146436)

[**IV. Tài liệu tham khảo** 26](#_Toc87146437)

1. **Giới thiệu về Prometheus**
2. **Khái niệm**

**1.1 Khái niệm về Prometheus:**

Prometheus là một hệ thống giám sát (monitoring system) với cơ sở dữ liệu theo thời gian (An opensource time series) và một bộ công cụ cảnh báo (alerting toolkit) mã nguồn mở.

Prometheus là một open-source systems monitoring và alerting ban đầu được xây dựng tại SoundCloud. Vào năm 2012 nhiều công ty, tổ chức đã đứng ra bảo trợ cho Prometheus, và project này cực kỳ và phát và có rất nhiều người dùng. Hiện tại nó không còn là một project độc lập mà được phát triển bởi rất nhiều công ty khác nhau. Nó sử dụng mã nguồn GoLang của google. Hiện tại thì Prometheus 100% là open source và có Apache 2 License on GitHub.

Tính năng quan trọng nhất của Prometheus là thu thập thông số, dữ liệu từ các mục tiêu (dịch vụ) được nhắm đến theo khoảng thời gian nhất định đã được cài đặt trước. Ngoài ra còn các API khác thể hiện được kết quả, đánh giá bằng biểu thức quy tắc và đưa ra cảnh báo. Prometheus còn cung cấp một ngôn ngữ truy vấn rất mạnh PromQL, cực kì hữu ích khi giao tiếp với các dịch vụ monitor khác.

**1.2 Time series database**

Dữ liệu sử dụng trong Prometheus là time series database, đây là một khái niệm sử dụng để mô tả những database chuyên dụng được sinh ra cho việc tối ưu lưu trữ dữ liệu theo các mốc thời gian, có thể kể đến như các thông số về tình trạng server, lượng tài nguyên RAM, CPU tiêu thụ, …

Các thông số này được lưu dưới dạng các mốc thời gian, từ những dữ liệu đó ta có thể xử lý để mô phỏng thành các biểu đồ giúp dễ dàng quan sát sự biến đổi của dữ liệu theo các mốc thời gian nhờ đó có thể nắm bắt được tình trạng của server để phát hiện những vấn đề và đưa ra cách giải quyết một cách kịp thời.

Cho một ví dụ: Server của chúng ta bị chậm đi trông thấy trong 3 ngày gần đây. Thông qua dữ liệu query từ Prometheus và visualize thành biếu đồ, chúng ta dễ dàng thấy server đột nhiên CPU luôn tăng cao trong những ngày này. Bạn bắt đầu đi kiểm tra các process và phát hiện những process lạ bị hacker cài cắm vào để thực hiện mục đích xấu như lợi dụng để đào coin. 

1. **Cách thức hoạt động của Prometheus**

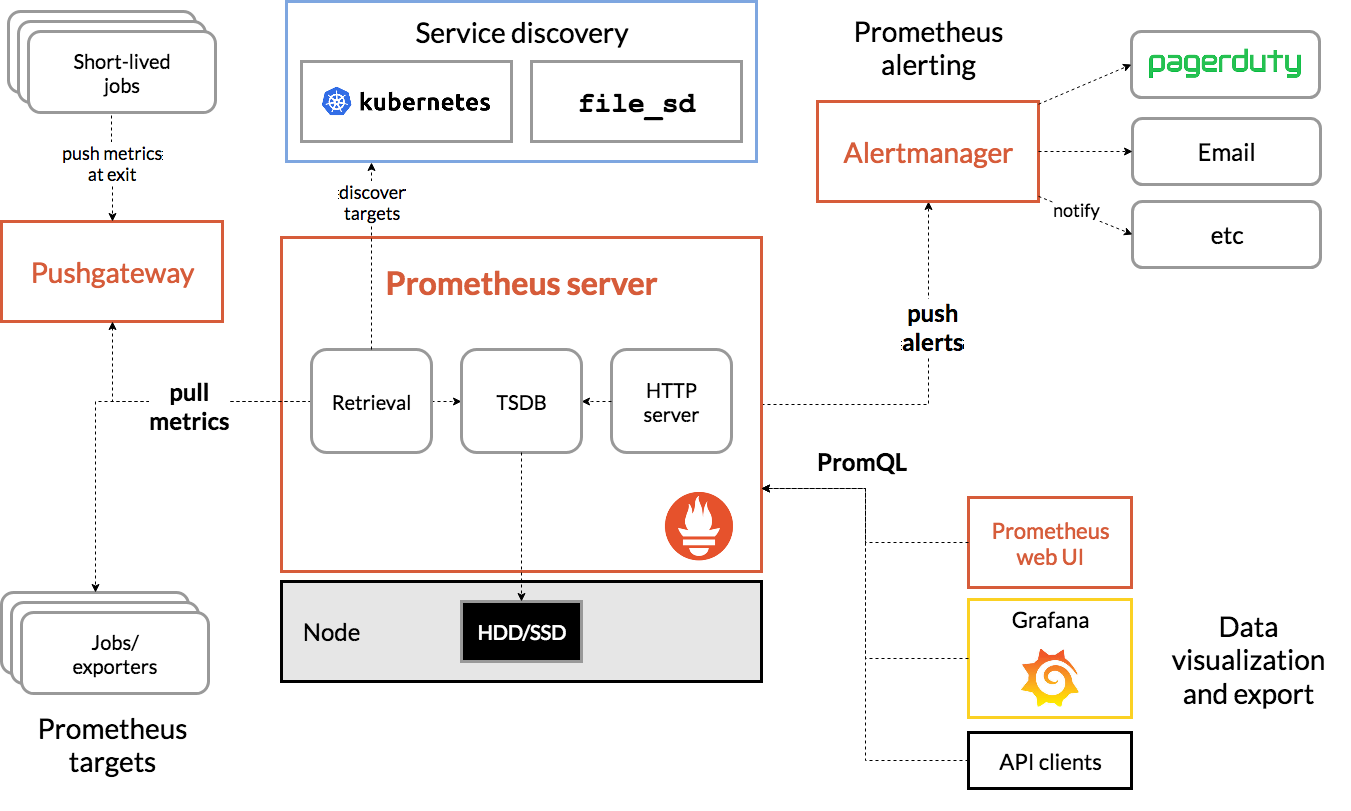
**2.1. Các thành phần chính trong hệ thống Prometheus**

* Máy chủ Prometheus.
* Thư viện kết nối đến máy chủ Prometheus cho các ứng dụng. (Có thể sử dụng dạng JSON nếu không có thư viện hỗ trợ)
* Push Gateway Prometheus: sử dụng để hỗ trợ các job có thời gian thực hiện ngắn (tạm thời). Đơn giản là các tác vụ công việc này không tồn tại lâu đủ để Prometheus chủ động lấy dữ liệu. Vì vậy là mà các dữ liệu chỉ số (metric) sẽ được đẩy về Push Gateway rồi đẩy về Prometheus Server.
* Exporter hỗ trợ giám sát các dịch vụ hệ thống và gửi về Prometheus theo chuẩn Prometheus mong muốn.
* AlertManager: dịch vụ quản lý, xử lý các cảnh báo (alert).

**Một số exporter có thể kể qua như:**

* Prometheus: chính bản thân prometheus cũng có một built-in exporter, export các metrics về service prometheus ra tại URI: http://prometheus.lc:9090/metrics
* cAdvisor : export các metrics của các docker service, các process trên server.
* Node Exporter: export các metrics một con node (hiểu là một server) như CPU, RAM của node, dung lượng ổ đĩa, số lượng request tới node đấy, .etc.
* Postgres Exporter, giúp đọc dữ liệu từ các bảng trong Postgres và export ra cho Prometheus
* HAProxy Exporter

**2.2.** **Cách thức hoạt động của Prometheus**

 Trên đây là mô hình Prometheus thường thấy. Cái hay của Prometheus đó là nó chủ động đi lấy dữ liệu theo dõi chứ không đợi các dịch vụ trả về. Vì vậy luôn đảm bảo phát hiện sớm những vấn đề có thể gây lỗi nghiêm trọng trong hệ thống

Theo lịch trình đã cài đặt Prometheus sẽ tự động thực hiện quá trình lấy các thông số/số liệu (metric) từ các dịch vụ, mục tiêu được chỉ định qua kết nối trực tiếp hoặc thông qua dịch vụ Pushgateway trung gian. Sau đó Prometheus sẽ lưu trữ các dữ liệu thu thập được ở máy chủ đang chạy. Tiếp đến là kiểm tra các rule đã được cài đặt để xử lý các dữ liệu theo nhu cầu cũng như kiểm tra và thực hiện các cảnh báo.

**2.2.1. Một số tính năng của Prometheus**

* Mô hình dữ liệu Time series đặc biệt phù hợp để theo dõi số liệu theo thời gian.
* Có ngôn ngữ truy vấn riêng PromQL rất mạnh mẽ.
* Tích hợp tốt với rất nhiều nền tảng ứng dụng.
* Yêu cầu hạ tầng để triển khai đơn giản.
* Hỗ trợ Push Gateway (khi làm việc với các dịch vụ tồn tại trong thời gian ngắn và hủy sau khi hoàn thành).
* Có hỗ trợ Service discovery hoặc cấu hình tĩnh để tìm, nhắm mục tiêu cần theo dõi.

**2.2.2. Một số thuật ngữ quan trọng với Prometheus**

* Time-series Data: là một chuỗi các điểm dữ liệu, thường bao gồm các phép đo liên tiếp được thực hiện từ cùng một nguồn trong một khoảng thời gian.
* Alert: một cảnh báo (alert) là kết quả của việc đạt điều kiện thoả mãn một rule cảnh báo được cấu hình trong Prometheus. Các cảnh báo sẽ được gửi đến dịch vụ Alertmanager.
* Alertmanager: chương trình đảm nhận nhiệm vụ tiếp nhận, xử lý các hoạt động cảnh báo.
* Client Library: một số thư viện hỗ trợ người dùng có thể tự tuỳ chỉnh lập trình phương thức riêng để lấy dữ liệu từ hệ thống và đẩy dữ liệu metric về Prometheus.
* Endpoint: nguồn dữ liệu của các chỉ số (metric) mà Prometheus sẽ đi lấy thông tin.
* Exporter: là một chương trình được sử dụng với mục đích thu thập, chuyển đổi các metric không ở dạng kiểu dữ liệu chuẩn Prometheus sang chuẩn dữ liệu Prometheus. Sau đấy exporter sẽ expose web service api chứa thông tin các metrics hoặc đẩy về Prometheus.
* Instance: một instance là một nhãn (label) dùng để định danh duy nhất cho một target trong một job.
* Job: là một tập hợp các target chung một nhóm mục đích. Ví dụ: giám sát một nhóm các dịch vụ database… thì ta gọi đó là một job
* PromQL: là viết tắt của Prometheus Query Language, ngôn ngữ này cho phép bạn thực hiện các hoạt động liên quan đến dữ liệu metric.
* Sample: là một giá trị đơn lẻ tại một thời điểm thời gian trong khoảng thời gian time series.
* Target: một target là định nghĩa một đối tượng sẽ được Prometheus đi lấy dữ liệu (scrape). Ví dụ như: nhãn nào sẽ được sử dụng cho đối tượng, hình thức chứng thực nào sử dụng hoặc các thông tin cần thiết để quá trình đi lấy dữ liệu ở đối tượng được diễn ra.

**2.3. Khi nào nên áp dụng Prometheus**

Prometheus rất thích hợp với những hệ thống Microservices và có các dịch vụ Listening. Một hệ thống theo dõi chủ động sẽ giúp người quản trị phát hiện sớm những dấu hiệu cảnh báo xấu.

Với những công việc liên quan đến Queue Job, mối nguy cơ luồng xử lý bị loop hoặc stop rất lớn. Lý do có thể đến từ tài nguyên phần cứng hoặc phần mềm được cài đặt không chính xác. Khi đó việc xem log của dịch vụ đó rất khó hoặc phụ thuộc vào may mắn. Với Prometheus các thông tin luôn được cập nhật và khi xảy ra lỗi, bạn vẫn có thể xem lại dữ liệu theo dõi một cách dễ dàng qua API của Prometheus.

Chúng ta cần chú ý đến một số đặc điểm của Prometheus như:

* Prometheus là 100% mã nguồn mở. Chúng ta có thể coi mã nguồn tại Git : **https://github.com/prometheus/prometheus/**
* Phần lớn các core tính năng của Prometheus được viết bằng ngôn ngữ Go. Một số còn lại thì được viết bằng Java, Python hoặc Ruby.
* Prometheus không dùng để lấy dữ liệu log, thay vì vậy thì Prometheus là dịch vụ giám sát, thu thập và xử lý dữ liệu dạng metric (thông số).
* Prometheus sử dụng cơ chế đi lấy (pull) dữ liệu từ máy chủ remote là chính, chứ không sử dụng cơ chế đợi remote đẩy (push) dữ liệu lên ngoại trừ trường hợp sử dụng PushGateway.
* Prometheus sử dụng chương trình cảnh báo Alertmanager để xử lý và gửi cảnh báo đi.
* Về phần giao diện biểu đồ (đồ thị) thì Prometheus sử dụng mã nguồn Grafana để tích hợp hiển thị.
* Metric của Prometheus sử dụng chuẩn OpenMetrics.
* Prometheus hỗ trợ 3 hình thức cài đặt các thành phần hệ thống gồm : Docker Image, cài đặt từ source với Golang và file chương trình chạy sẵn đã được biên dịch sẵn.

1. **Cài đặt Prometheus trên môi trường Linux**

**3.1 Cài đặt Prometheus**

Để cài đặt Prometheus đầu tiên chúng ra cài tải phiên bản Prometheus thích hợp về máy.

1. Chúng ta có thể tải thông qua wget bằng lệnh như sau:



1. Tiếp theo chúng ta giải nén thư mục **prometheus-2.20.0-rc.1.linux-amd64.tar.gz** mà chúng ta đã tải về:



Sau khi giải nén ta thu được thư mục với nội dung như sau:

Ở đây có một số file chúng ta cần chú ý như:

* prometheus.yml: Đây là file cấu hình của Prometheus, bằng cách chỉnh sửa file này bạn có thể tuỳ chỉnh Prometheus server theo cách của bạn. Các bạn có thể tham khảo thêm tại Prometheus configuration.
* prometheus: Là một binary file. Bạn có thể thực thi file này để khởi chạy Prometheus.
* promtool: Là một tool giúp bạn kiểm tra các thiết lập cho Prometheus.

1. Một cách đơn giản chúng ta hoàn toàn có thể chạy trực tiếp file prometheus. Nhưng không ai làm như vậy, chúng ta cùng thiết lập để chạy Prometheus như một service trên môi trường Linux.

Để bảo mật chúng ta tạo một user và một group để quản lý Prometheus:



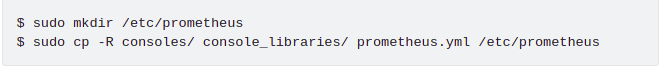
1. Tiếp theo chúng ta di chuyển các file binary trong thư mục đã giải nén về **/usr/local/bin:**



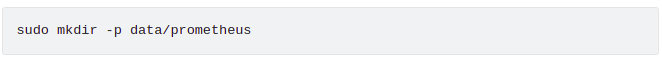
1. Cấp quyền cho **user** chúng ta đã tạo đối với prometheus file:

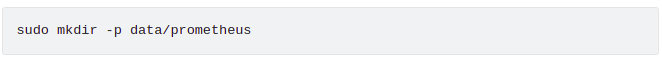


1. Tạo 1 thư mục prometheus trong **/etc** và di chuyển các tài nguyên cần thiết:

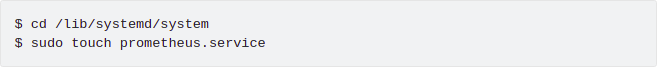


1. Tạo một thư mục để lưu trữ dữ liệu cho Prometheus và cấp quyền có các thư mục vừa tạo:





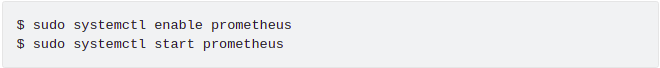
1. Chúng ta tạo một service để thực thi Prometheus. Di chuyển đến **/lib/systemd/system** và tạo 1 file prometheus.service:



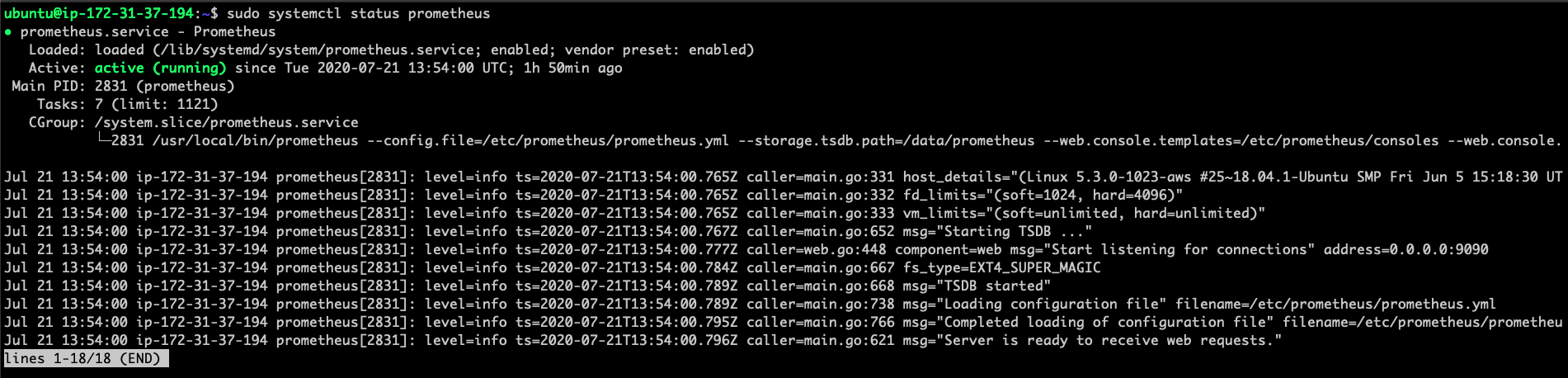
1. Định nghĩa file **prometheus.service** như sau:



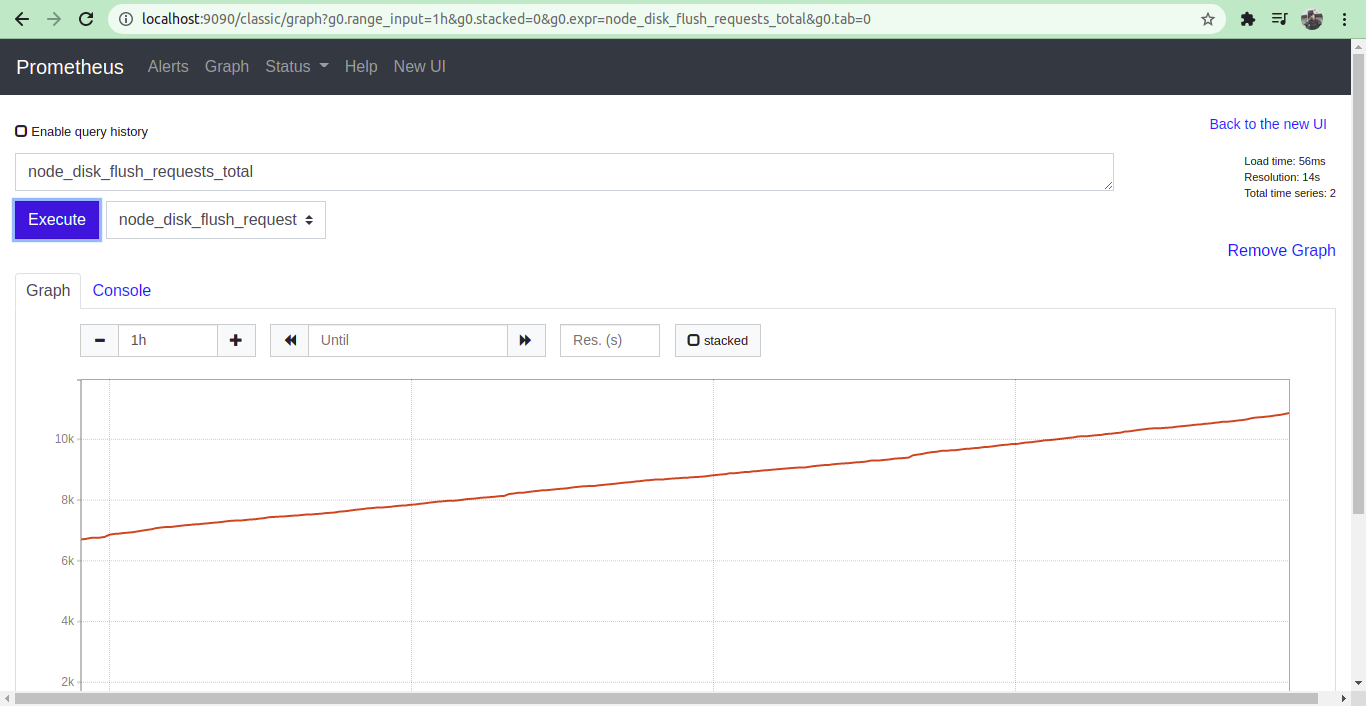
(10) Như vậy khi service start chương trình prometheus trong thư mục **/usr/local/bin** sẽ được thực thi với các param tương ứng đã được định nghĩa. Ta lưu lại file và thực hiện **start service**:

(11) Check trạng thái của service với câu lệnh:

Nếu trạng thái như sau tức là service chúng ta đã khởi chạy thành công:

Nếu service khởi chạy thất bại các bạn có thể sử dụng lệnh để có thể lấy thông tin về lỗi đã xảy ra và fix: **$ journalctl -xe** 

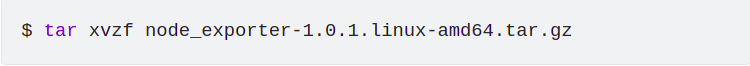
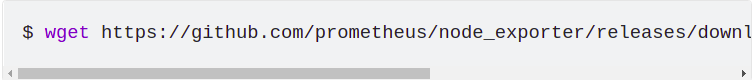
Sau khi service đã khởi chạy các bạn có thể truy cập vào địa chỉ [**http://localhost:9090/graph**](http://localhost:9090/graph)**.** Mặc định Prometheus sẽ tự monitoring chính bản thân nó, chúng ta có thể thực thi các query và có thể xem các biểu đồ tương ứng:



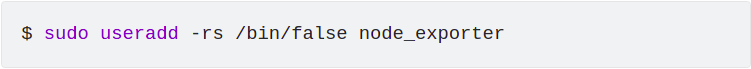
**3.2 Cài đặt Node Exporter**

Việc thiết lập cài đặt Node Exporter khá giống với việc chúng ta cài đặt Prometheus như trên. Chúng ta cùng nhau cài đặt Node Exporter như một service chạy trên hệ điều hành Linux của chúng ta. Đầu tiên chúng ta cần tải phiên bản mới nhất của Node Exporter về. Các bạn có thể tìm kiếm phiên bản của Node Exporter tại <https://prometheus.io/download/>**.**

1. Tải file node exporter về máy :

Tiếp theo là giải nén thư mục chúng ta vừa mới tải về:

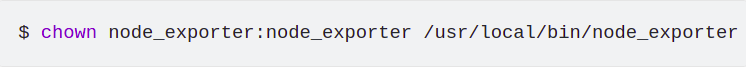
1. Tạo một user cho việc quản lý exporter:

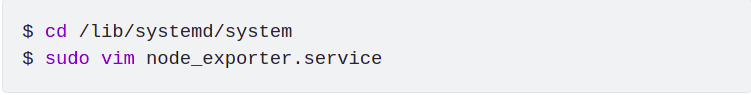


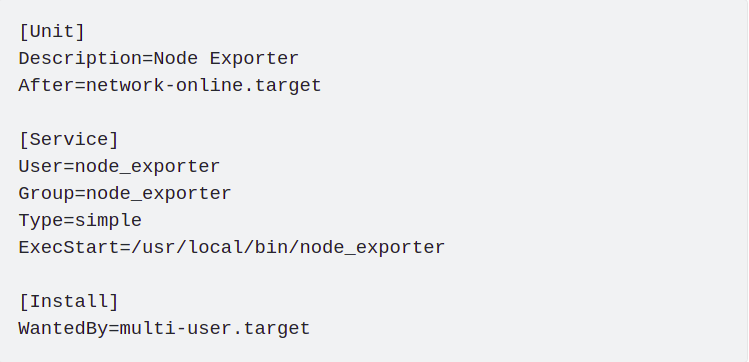
1. Copy binary file trong thư mục đã giải nén tới địa chỉ **/usr/local/bin:**



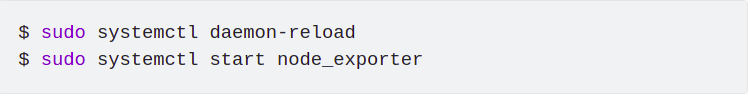
1. Thiết lập quyền cho binary file:

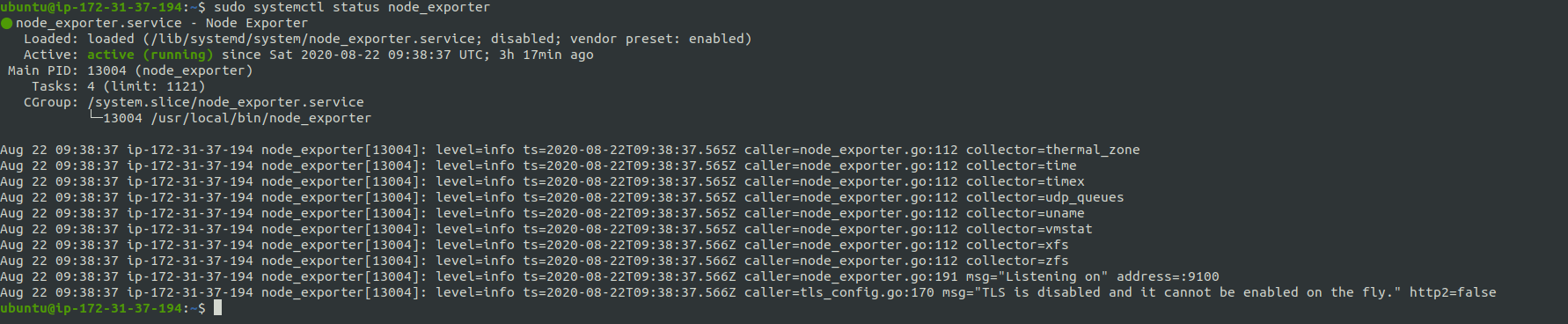


1. Tạo một service cho việc chạy Node Exporter
2. Thêm nội dung như sau vào service file:



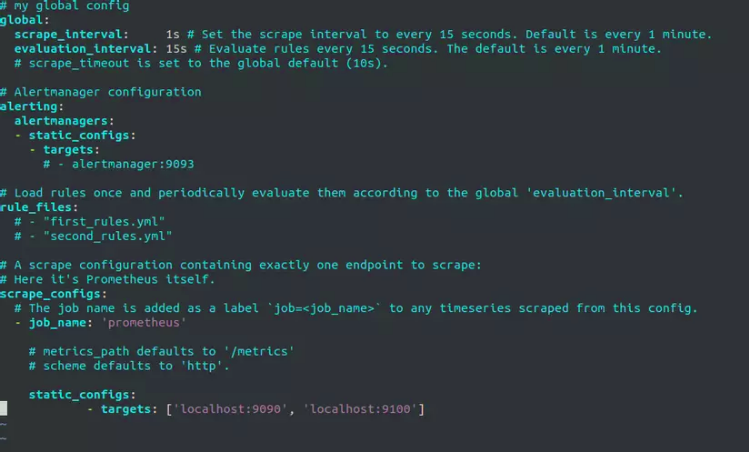
1. Khởi chạy service chúng ta vừa mới tạo:

Kiểm tra service với lệnh: **sudo systemctl status node\_exporter**



Như hình bên trên là service của chúng ta đã hoạt động. Mặc định của Node Exporter sẽ hoạt động tại cổng 9100.Chúng ta có thể kiểm tra bằng cách truy cập vào địa chỉ <http://localhost:9100/metric>

1. Sau khi Node Exporter đã hoạt động chúng ta cần chỉnh sửa file cấu hình của Prometheus là prometheus.yml.Thay đổi thiết lập **global > scrape\_interval** về 1s để có thể cào dữ liệu mỗi giây và static\_configs thêm địa chỉ để lấy dữ liệu về từ Node Exporter.

Như vậy chúng ta đã hoàn thành công việc thiết lập Node Exporter và xuất dữ liệu cho Prometheus. Dữ liệu chúng ta đã có, để có thể sử dụng dữ liệu và hiển thị trên một dashboard - một công cụ có tên là Grafana.

**II . Tìm hiểu về Grafana**

**1. Khái niệm**

Là một công cụ trực quan hóa, Grafana là một thành phần phổ biến trong các ngăn xếp giám sát, thường được sử dụng kết hợp với cơ sở dữ liệu chuỗi thời gian như InfluxDB , Prometheus và Graphite ; các nền tảng giám sát như Sensu, Icinga , Checkmk , Zabbix , Netdata , và PRTG ; SIEM như Elasticsearch và Splunk ; và các nguồn dữ liệu khác.

Nói dễ hình dung là sau khi lấy được metric từ các thiết bị, grafana sẽ sử dụng metric đó để phân tích và tạo ra dashboard mô tả trực quan các metric cần thiết cho việc monitoring ví dụ như cpu, ram, disk, network, iops, session.

Việc xây dựng Dashboard nó là một phần quan trọng trong việc monitor của hệ thống. Grafana support rất nhiều giải pháp monitor khác nhau.

**2. Các tính năng của Grafana**

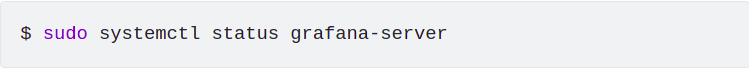
1. Nhiều tùy chọn hiển thị vì chúng tôi sẽ có quyền truy cập vào nhiều loại đồ họa, tiện ích bảng điều khiển và nhiều tùy chọn khác để hiển thị số liệu và hồ sơ.
2. Nó có một hệ thống cảnh báo trong đó mỗi số liệu sẽ theo một quy trình đánh giá trong một khoảng thời gian xác định và, nếu có bất kỳ sự khác biệt nào, một cảnh báo sẽ được tạo ra từ tình huống này.
3. Tùy chọn thông báo mở rộng, chúng được đưa ra khi cảnh báo sửa đổi trạng thái của nó và có thể nhận thông báo qua email hoặc thông qua các nền tảng như Slack, PagerDuty, VictorOps, OpsGenie hoặc webhook.
4. Nó có bảng điều khiển động mà chúng ta có thể tạo bảng điều khiển động và có thể sử dụng lại bằng cách sử dụng các biến mẫu khác nhau trong đó các menu thả xuống và nhiều hành động có thể được thêm vào.
5. Các nguồn dữ liệu hỗn hợp cho phép trộn các loại dữ liệu khác nhau trong một biểu đồ để có khái niệm thông tin tốt hơn.

**3. Cài đặt và sử dụng Grafana**

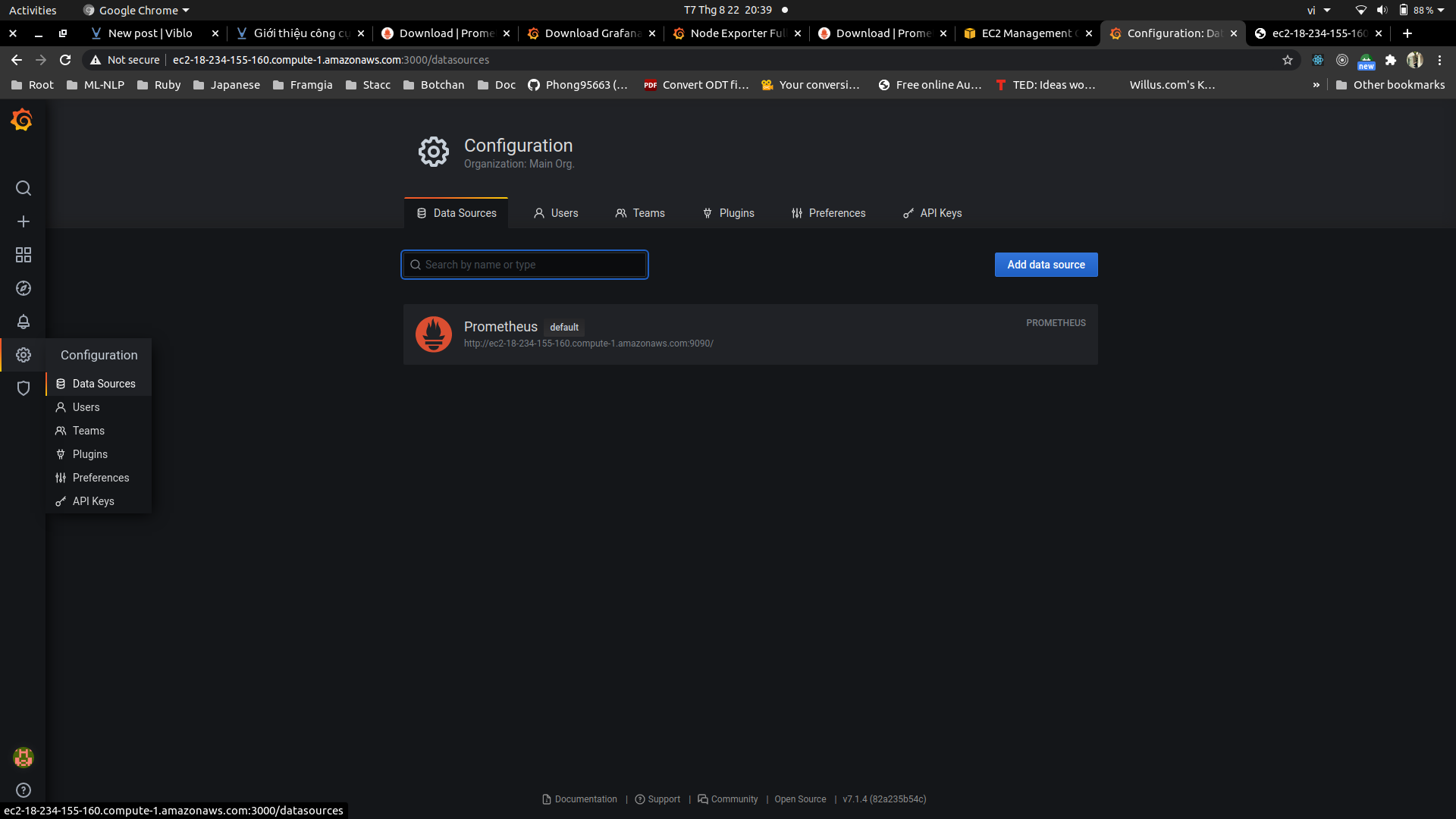
Để cài đặt Grafana hết sức đơn giản, chúng ta có thể tại Grafana về tại **https://grafana.com/grafana/download** thực hiện cài đặt theo hướng dẫn và start service.

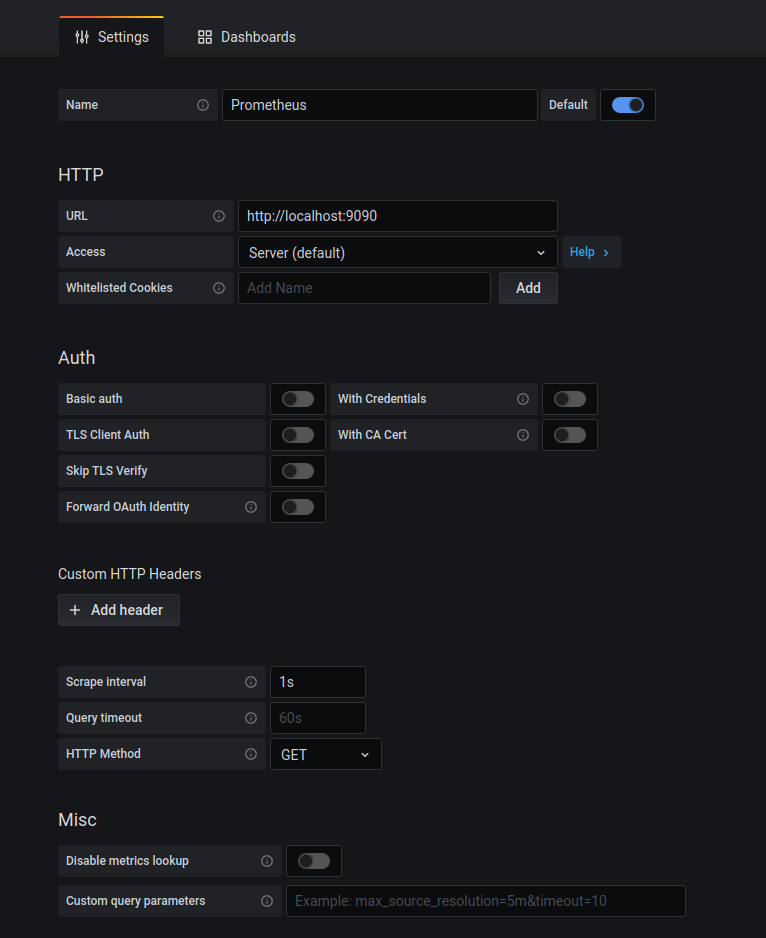
các lệnh được thực hiện lần lượt như sau :

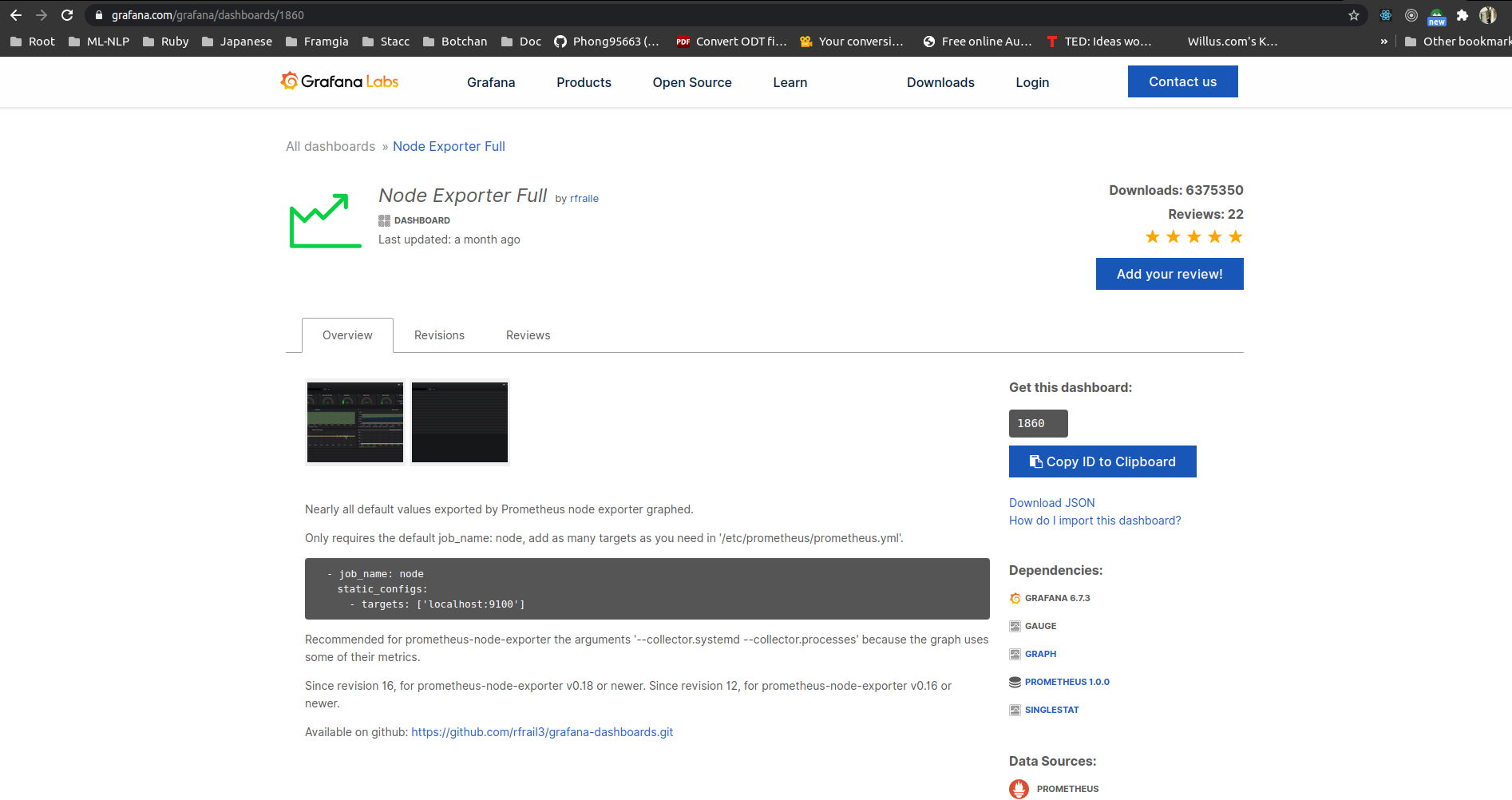
Tiếp theo chúng ta tiến hành start services để khởi chạy Grafana:

Mặc định Grafana sẽ khởi chạy tại cổng 3000. Chúng ta sẽ truy cập tại địa chỉ **http://localhost:3000**. Một trang web hiện ra và thực hiện đăng nhập với tài khoản mặc định là **admin** và mật khẩu là **admin**. Sau khi đăng nhập lần đầu chúng ta thực hiện việc thay đổi mật khẩu mặc định và chúng ta vào được giao diện quản lý của Grafana. 

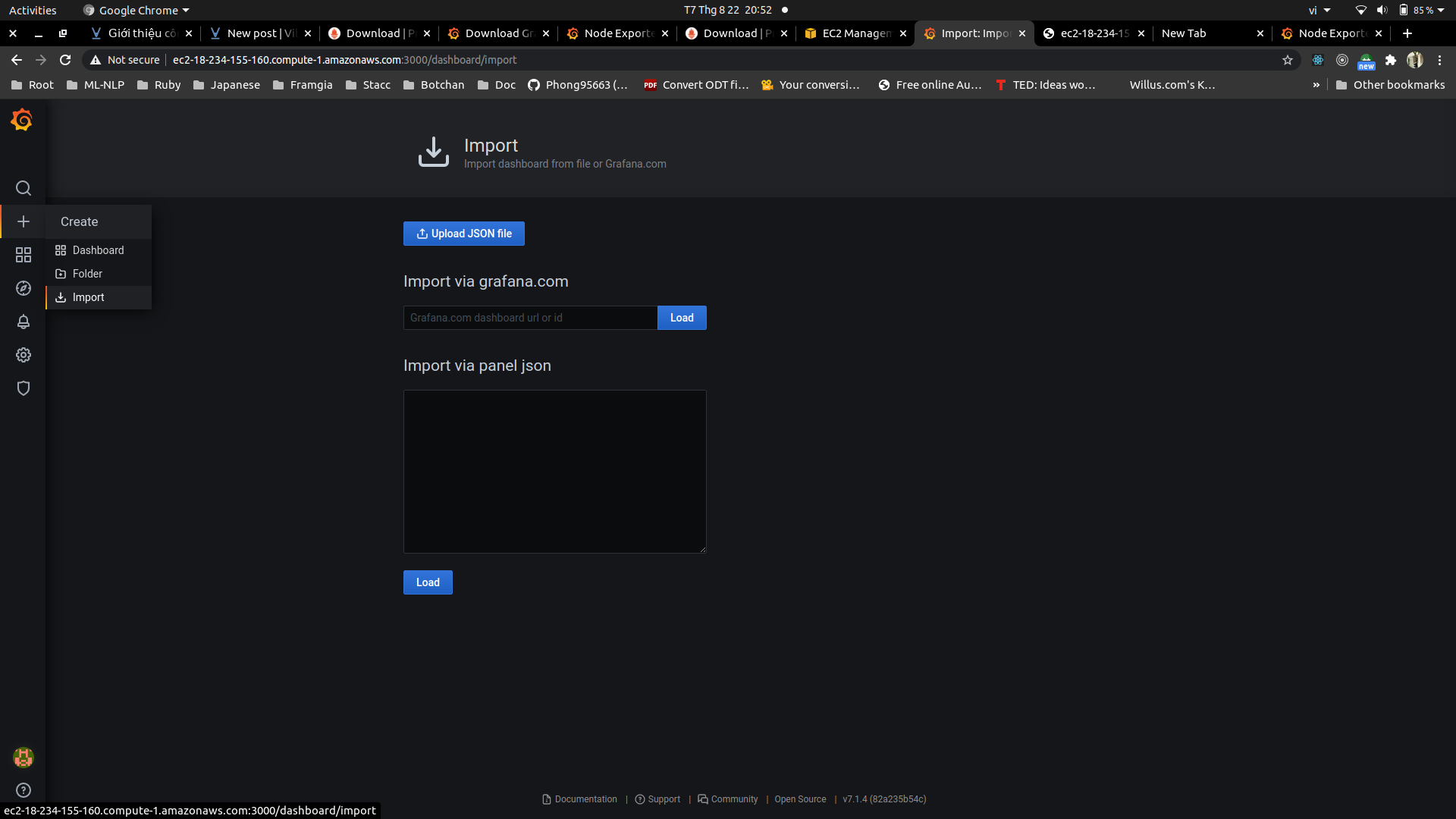
Chúng ta vào Configuration và thực hiện add Data source từ Prometheus và thiết lập.

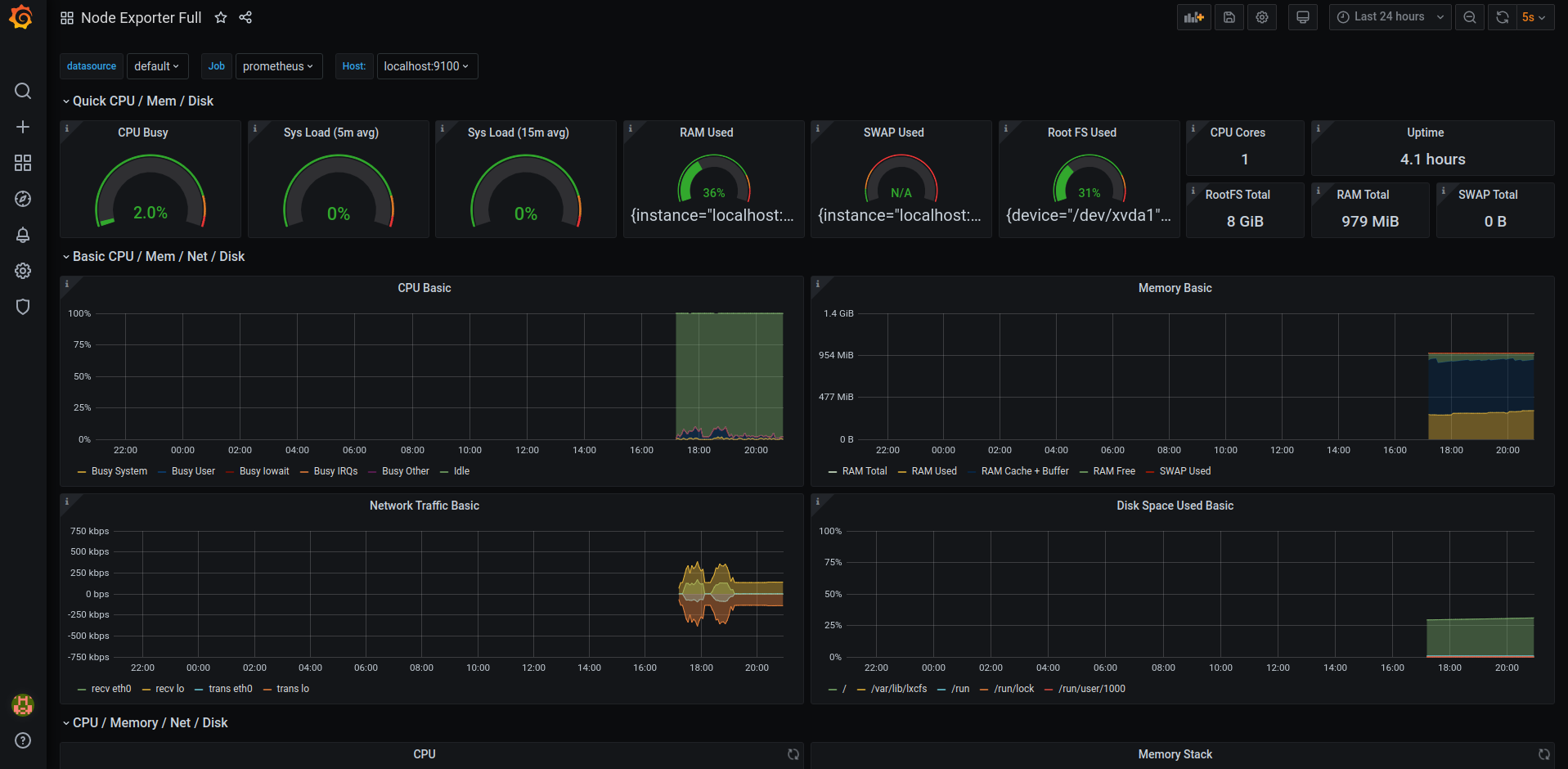


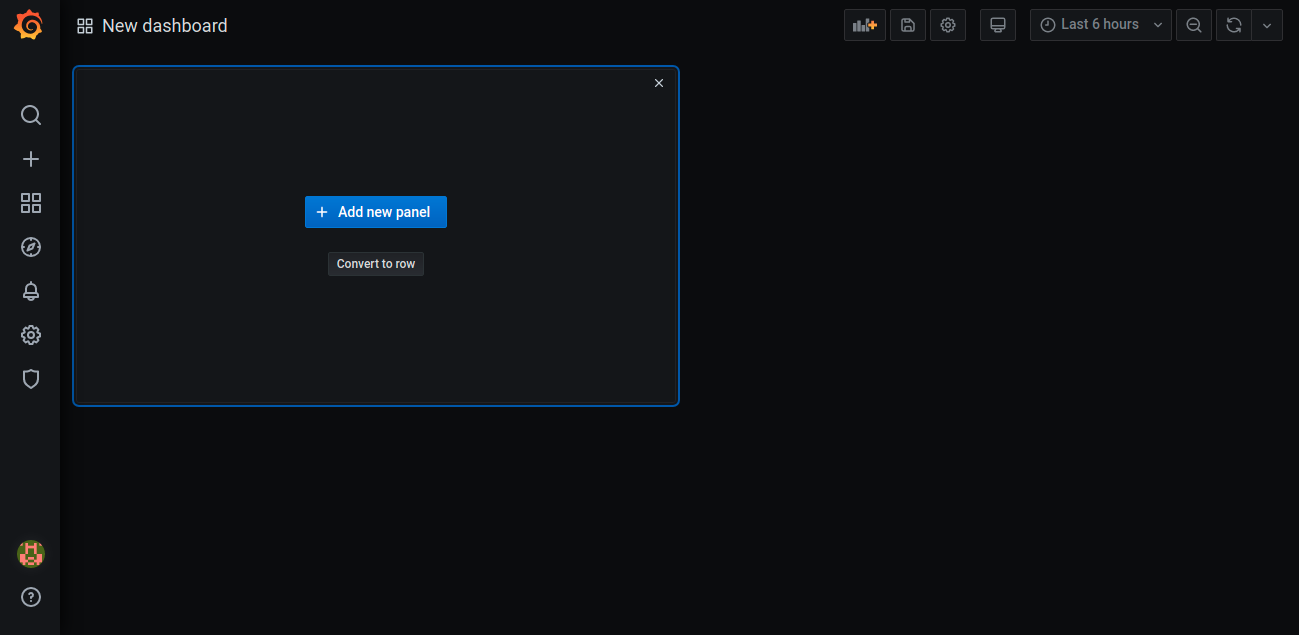
Tiếp theo chúng ta tạo một Dashboard cho Node Exporter một cách đơn giản nhất là sử dụng những Dashboard đã được xây dựng sẵn. Có rất nhiều Dashboard đã được xây dựng sẵn với nhiều mục đich chúng ta có thể tìm kiếm được tại đây [**https://grafana.com/grafana/dashboards**](https://grafana.com/grafana/dashboards) **.**Chúng ta có thế sử dụng một Dashboard cho Node Exporter có tên là Node Exporter Full các bạn có thể tìm kiếm tại địa chỉ trên. Dashboard này có ID 1860 chúng ta có thể lấy được dễ dàng thông qua URL.



Để import Dashboard bạn click vào nút dấu cộng và chọn import. Nhập ID của Dashboard và chọn Load một màn hình config hiện ra, chúng ta chọn Data Source từ **Prometheus** đã tạo từ trước và nhấn vào import.

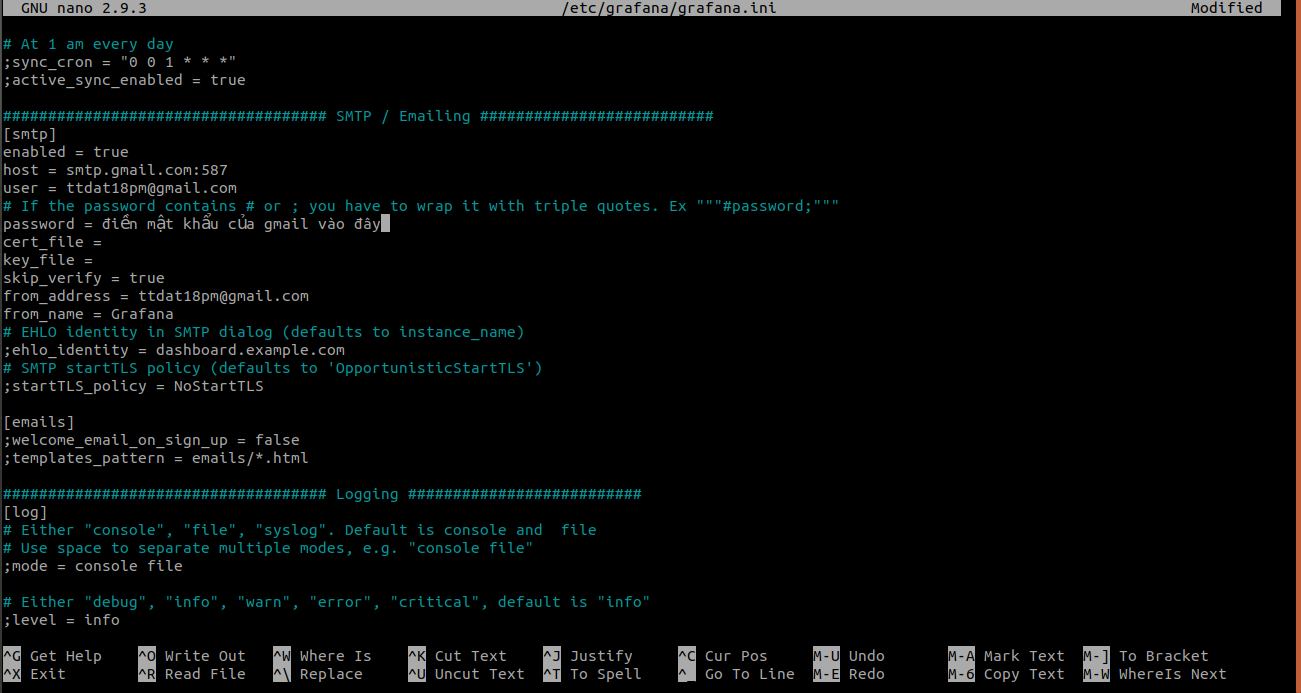
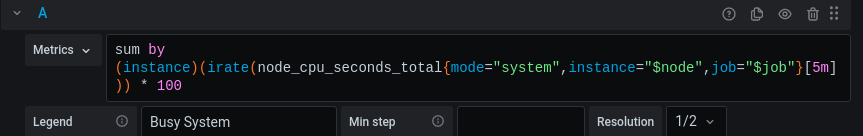
Và kết quả thu được như sau : 

Ngoài ra để tạo được dashboard thì chúng ta chọn create và chọn Add new panel để có thể tùy chỉnh.

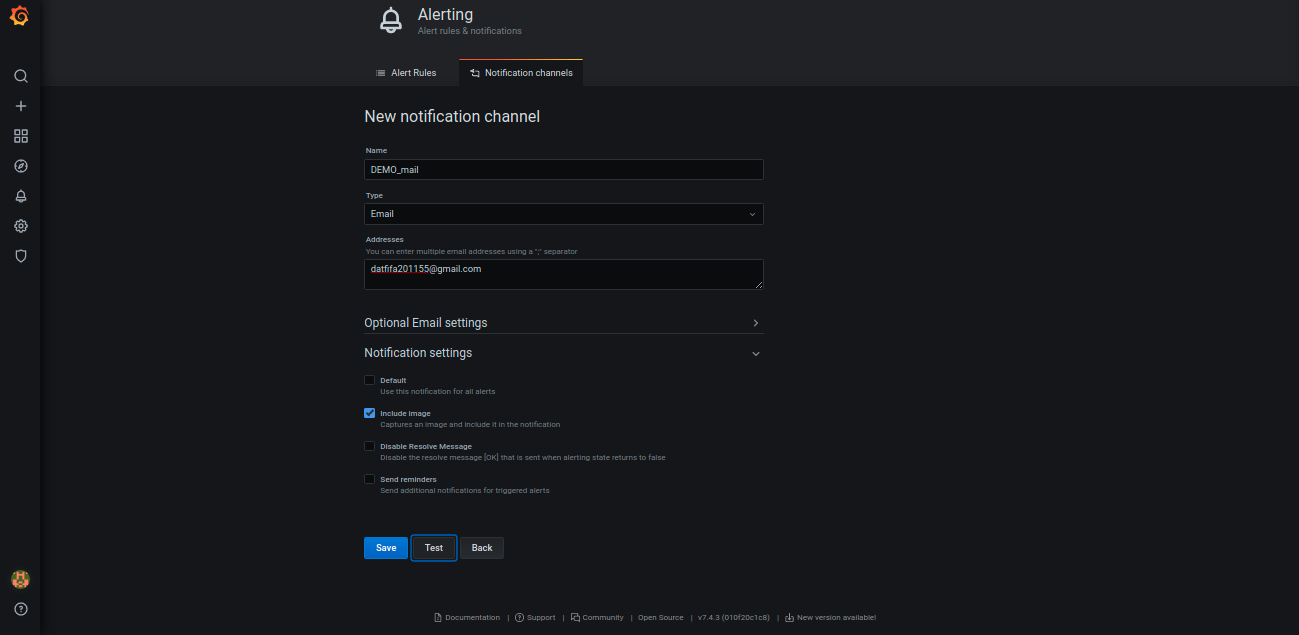
* **Cấu hình Alert (cảnh báo)**

Tiếp theo chúng ta cần cấu hình cảnh báo cho grafana, có 1 điều cần chú ý là: khi sử dụng dashboard được cấu hình sẵn thì chúng ta sẽ không cấu hình alert được,lý do là khi gọi các metric có sử dụng biến để có thể phân biệt được thời gian cũng như của server nào, vì thế nếu chúng ta có nhiều server thì chúng ta phải tạo ra nhiều khung hiển thị để hiển thị từng thông tin tài nguyên của server đó và bỏ đi phần biến lúc gọi metric để có thể cấu hình alert.

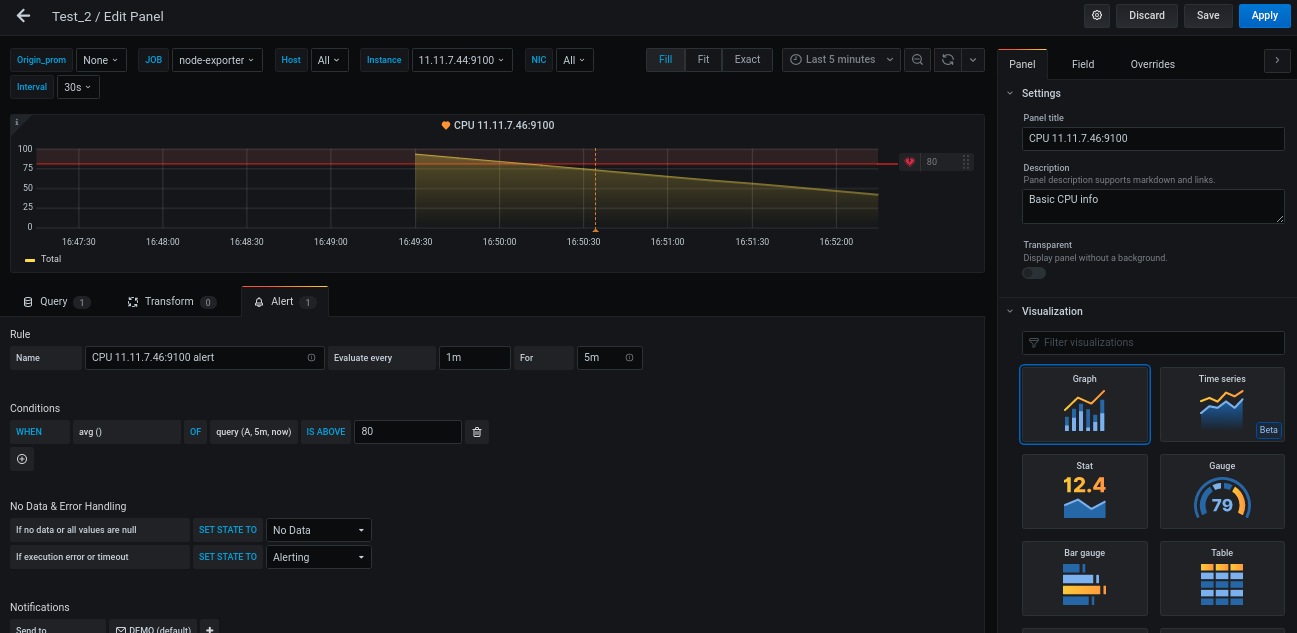
Chúng ta cần phải cấu hình lại grafana.ini để có thể gửi cảnh báo về mail của chúng ta, đường dẫn để truy cập file grafana.ini là: **/etc/grafana/grafana.ini**

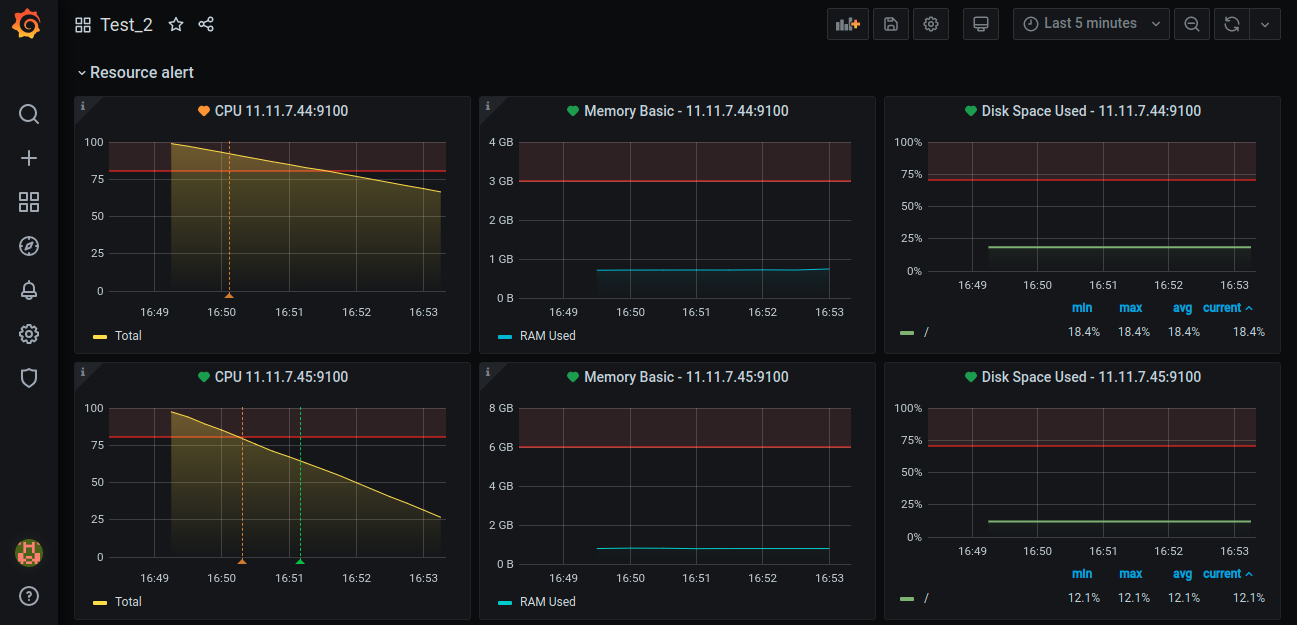
Ta cần thêm 1 số thông tin ở phần smtp, cần chú ý là gmail mà chúng ta nhập vào ở đây, grafana sẽ dùng gmail đó để viết và gửi cảnh báo cho chúng ta, sau khi chỉnh sửa thì chạy lệnh sudo service **grafana-server restart** để restart lại grafana.

Trước khi thiết lập cảnh báo thì chúng ta cần tạo 1 kênh thông báo với 1 email cụ thể để có thể gửi cảnh báo đến email đó. Ta ấn test để kiểm tra thử email có được gửi thành công hay không.

Và đây là email chúng ta nhận được

Để cấu hình cảnh báo thì chúng chọn Alert và chọn Create Alert để bắt.

Tại đây chúng ta điền vào các thông số mà chúng ta quy định, ví dụ như trong ảnh, hệ thống sẽ cảnh báo khi CPU vượt ngưỡng 1, sau đó chúng ta chọn test rule để kiêm tra alert chúng ta cài đặt có hoạt động không, sao đó chọn save để có thể lưu lại

Khi chúng ta thiết lập cảnh báo thành công là phía trên mỗi khung dữ liệu điều có hình trái tim, màu xanh là an toàn các tài nguyên thấp hơn mức quy định, màu vàng là khi tài nguyên vượt mức quy định, nhưng chưa đủ thời gian gây ra nguy hiểm mà chúng ta thiết lập, màu đỏ là vượt qua thời gian an toàn thì hệ thống sẽ gửi cảnh báo qua email của chúng ta kèm theo hình ảnh khi vượt ngưỡng an toàn

1. **Kết luận**

Qua bài báo cáo này em đã biết thêm được nhiều kiến thức khái niệm về Prometheus, những đặc điểm của Prometheus, tại sao nên sử dụng Prometheus để thực hiện giám sát, cũng như nắm thêm kiến thức về Các thành phần chính trong hệ thống Prometheus, một số thuật ngữ quan trọng Prometheus. Đồng thời cũng đã tìm hiểu về cách cài đặt cũng như cấu hình cần thiết để Prometheus có thể hoạt động .Bên cạnh đó em cũng biết thêm được các định nghĩa về Grafana là gì ? các tính năng và lợi ích mà Grafana mang lại, hiểu được cách cài đặt và sử dụng kết hợp giữa Grafana và Prometheus.

Trong quá trình xây dựng bản báo cáo em cũng gặp một số vấn đề khó khăn , điển hình như đây là vấn đề khá mới đối với bản thân , tài liệu tìm kiếm phần đa là được viết bằng tiếng Anh và tiếng Nhật , hơn thế nữa kỹ năng xây dựng báo cào cùng với lời văn còn yếu mong nhận được sự góp ý nhiều hơn từ các anh chị đánh giá.

# **IV. Tài liệu tham khảo**

1. Tài liệu tham khảo về Monitoring Linux Processes using Prometheus and Grafana :

[https://devconnected.com/](%20%20https:/devconnected.com/)

1. Cách thức cài đặt Prometheus

https://prometheus.io/

1. Grafana tutorial :

<https://www.udemy.com/course/grafana-tutorial-g/>