BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ĐỒ ÁN TỐT NGHIỆP

NGÀNH : CÔNG NGHỆ THÔNG TIN

Sinh viên : Đoàn Quốc Anh

Giảng viên hướng dẫn: ThS. Đỗ Văn Tuyên

HẢI PHÒNG – 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

NGHIÊN CỨU TÌM HIỂU VÀ ỨNG DỤNG GRAFANA– CÔNG CỤ GIÁM SÁT MÁY CHỦ VÀ DỊCH VỤ MẠNG TẠI TRƯỜNG ĐH QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH: Công nghệ thông tin

Sinh viên : Đoàn Quốc Anh

Giảng viên hướng dẫn: ThS. Đỗ Văn Tuyên

HẢI PHÒNG – 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: Đoàn Quốc Anh **Mã SV**: 2112111038

Lớp : CT2501C

Ngành : Công nghệ thông tin

Tên đề tài:*" Nghiên cứu tìm hiểu và ứng dựng Grafana– công cụ giám sát máy chủ và dịch vụ mạng tại trường ĐH Quản lý và Công nghệ Hải Phòng "*

NHIỆM VỤ ĐỀ TÀI

1. Nội dung và các yêu cầu cần giải quyết trong nhiệm vụ đề tài tốt nghiệp

a. Mô tả tóm tắt đề tài

Đề tài tập trung vào việc nghiên cứu và ứng dụng công cụ Grafana để giám sát hiệu suất máy chủ, dịch vụ mạng tại Trường ĐH Quản lý và Công nghệ Hải Phòng. Grafana sẽ được tích hợp với các nguồn dữ liệu như Prometheus, InfluxDB để thu thập, phân tích và hiển thị dữ liệu dưới dạng dashboard trực quan, giúp nâng cao khả năng quản lý và vận hành hệ thống.

b. Nội dung hướng dẫn

- Nghiên cứu lý thuyết:
  + Tổng quan về giám sát hệ thống và vai trò của Grafana.
  + Các công nghệ liên quan (Prometheus, InfluxDB, Discord).
- Triển khai thực tế:
  + Cài đặt và cấu hình Grafana trên máy chủ.
  + Kết nối Grafana với các nguồn dữ liệu.
  + Thiết kế dashboard giám sát các chỉ số: CPU, RAM, băng thông, uptime.
- Đánh giá hiệu quả:
  + So sánh hiệu suất hệ thống trước và sau khi triển khai.
  + Phân tích ưu/ nhược điểm của giải pháp.

c. Kết quả cần đạt được

- Báo cáo chi tiết về quy trình triển khai Grafana.
- Hệ thống dashboard hoàn chỉnh giám sát máy chủ và dịch vụ mạng.
- Đề xuất cải tiến dựa trên kết quả thu thập.

d. Các yêu cầu đối với sinh viên

- Có kiến thức cơ bản về mạng máy tính và hệ thống.
- Kỹ năng lập trình cơ bản (Python, Bash).
- Khả năng tự nghiên cứu và làm việc độc lập.

`

2. Các tài liệu, số liệu cần thiết

- Tài liệu chính thức của Grafana: https://grafana.com.docs/
- Sách “Monitoring with Grafana” – Alan Hohn.
- Các bài báo khoa học về giám sát hệ thống.

3. Địa điểm thực tập tốt nghiệp

- Trường Đại học Quản lý và Công nghệ Hải Phòng

`

CÁN BỘ HƯỚNG DẪN ĐỀ TÀI TỐT NGHIỆP

Họ và tên : Đỗ Văn Tuyên

Học hàm, học vị : Thạc sỹ

Cơ quan công tác : Khoa Công nghệ thông tin

Nội dung hướng dẫn:

- Tổng quan về giám sát mạng và các công cụ giám sát phổ biến.
- Tìm hiểu về Grafana.
- Triển khai Grafana tại trường Đại học Quản lý và Công nghệ Hải Phòng.
- Đánh giá hiểu quả của Grafana.
- Đề xuất các giải pháp tối ưu hóa và nâng cao hiệu quả giám sát.

Đề tài tốt nghiệp được giao ngày 16 tháng 08 năm 2025

Yêu cầu phải hoàn thành xong trước ngày 15 tháng 11 năm 2025

Đã nhận nhiệm vụ ĐTTN	**Đã giao nhiệm vụ ĐTTN**
Sinh viên	Giảng viên hướng dẫn
Đoàn Quốc Anh	ThS. Đỗ Văn Tuyên

Hải Phòng, ngày tháng năm 2025

TRƯỞNG KHOA

`

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN TỐT NGHIỆP

Họ và tên giảng viên: Đỗ Văn Tuyên

Đơn vị công tác: Khoa Công nghệ thông tin - Trường Đại học Quản Lý và Công Nghệ Hải Phòng.

Họ và tên sinh viên : Đoàn Quốc Anh **Ngành**: Công nghệ Thông tin

Nội dung hướng dẫn : Toàn bộ đề tài

1. Tinh thần thái độ của sinh viên trong quá trình làm đề tài tốt nghiệp

2. Đánh giá chất lượng của đồ án/khóa luận (so với nội dung yêu cầu đã đề ratrong nhiệm vụ Đ.T.T.N, trên các mặt lý luận, thực tiễn, tính toán số liệu...)

3. Ý kiến của giảng viên hướng dẫn tốt nghiệp

Được bảo vệ ☐ Không được bảo vệ ☐ Điểm hướng dẫn ☐

Hải Phòng, ngày .... tháng .... năm 2025

Giảng viên hướng dẫn

(*ký và ghi rõ họ tên)*

`

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN CHẤM PHẢN BIỆN

Họ và tên giảng viên: ……

Đơn vị công tác: ……

Họ và tên sinh viên: Đoàn Quốc Anh **Ngành**: Công nghệ Thông tin

Đề tài tốt nghiệp: Nghiên cứu và tìm hiểu và ứng dụng Uptime Kuma – công cụ giám sát trạng thái website và dịch vụ mạng tại Trường Đại học Quản lý và Công nghệ Hải Phòng

1. Phần nhận xét của giảng viên chấm phản biện

……

……

……

……

……

2. Những mặt còn hạn chế

……

……

……

……

……

3. Ý kiến của giảng viên chấm phản biện

Được bảo vệ ☐ Không được bảo vệ ☐ Điểm phản biện ☐

Hải Phòng, ngày.....tháng ..... năm 2025

Giảng viên chấm phản biện

(*ký và ghi rõ họ tên*)

`

LỜI CẢM ƠN

Trong quá trình làm đồ án vừa qua được sự chỉ dẫn nhiệt tình của thầy ThS. Đỗ Văn Tuyên – Trường Đại học Quản lý và Công nghệ Hải Phòng, em đã hoàn thành đồ án của mình. Mặc dù em đã cố gắng với sự tận tâm của thầy, nhưng vì thời gian và khả năng nên đồ án của em vẫn còn không tránh được những điều thiếu sót.

Em xin chân thành và bày tỏ lòng biết ơn sâu sắc đến thầy Đỗ Văn Tuyên vì đã tận tình chỉ bảo, hướng dẫn và giành thời gian quý báu của mình cho em trong thời gian qua để em có thể hoàn thành đồ án của mình đúng thời hạn.

Em xin cảm ơn tất cả thầy cô giáo trong khoa Công nghệ thông tin vì đã truyền đạt cho em rất nhiều các kiến thức nền tảng, chuyên ngành, chuyên môn và chuyên sâu cực kì vững chắc trong những năm qua để em có thể hoàn thành được đồ án này.

Em xin cảm ơn Trường Đại học Quản lý và Công nghệ Hải Phòng vì không ngừng hỗ trợ và đào tạo những điều kiện tốt nhất trong những năm vừa qua để em có thể học và thực hiện tốt đồ án.

Em xin chân thành cảm ơn !

`

MỤC LỤC

LỜI CẢM ƠN
DANH MỤC HÌNH ẢNH
DANH MỤC BẢNG
MỞ ĐẦU 1
CHƯƠNG 1: TỔNG QUAN VỀ GIÁM SÁT MẠNG VÀ HỆ THỐNG MÁY CHỦ 4
1.1. Tìm hiểu về giám sát mạng và hệ thống máy chủ 4
1.1.1. Khái niệm 4
1.1.2. Các yếu tố cơ bản trong giám sát mạng 4
1.1.3. Chức năng của hệ thống giám sát 5
1.1.4. Đối tượng và mục tiêu giám sát 6
1.1.5. Tầm quan trọng của giám sát mạng 7
1.1.6. Các loại hình giám sát 8
1.1.7. Ưu điểm và nhược điểm của hệ thống giám sát 8
1.2. Các công cụ giám sát mạng và hệ thống máy chủ phổ biến 9
1.3. So sánh các công cụ 13
CHƯƠNG 2: TÌM HIỂU VỀ HỆ THỐNG GIÁM SÁT GRAFANA VÀ PROMETHEUS 14
2.1. Giới thiệu Grafana 14
2.2. Thông tin và tính năng Grafana: 15
2.3 Giới thiệu Prometheus 15
2.4 Thông tin và tính năng của Prometheus 16
2.5 Cách thức hoạt động 17
2.6 Cách thức hoạt động từng bước: 19
2.7 Các phương thức giám sát trong hệ thống (Grafana & Prometheus) 22
2.8 Ưu điểm và nhược điểm của hệ thống giám sát 23
2.8.1 Ưu điểm 23
2.8.2 Nhược điểm 24

`

2.9 So sánh Grafana với các công cụ khác 26
CHƯƠNG 3: TRIỂN KHAI HỆ THỐNG GIÁM SÁT GRAFANA TẠI NHÀ TRƯỜNG. 29
3.1. Mục tiêu triển khai 29
3.2. Dịch vụ giám sát trong nhà trường 29
3.3. Mô hình triển khai thực tế 30
3.4 Triển khai môi trường và hạ tầng 32
3.5 Cấu hình thu thập dữ liệu 34
3.6 Cấu hình trực quan hóa trên Grafana 36
3.6.1. Kết nối Data Source: Thiết lập kết nối tới Prometheus 36
3.6.2. Xây dựng dashboard giám sát hiệu năng server và giám sát dịch vụ website (dashboard tổng) 39
3.7 Thiết lập hệ thống cảnh báo 43
3.7.1 Cấu hình kênh thông báo (Contact Point) 43
3.7.2 Thiết lập Quy tắc cảnh báo (Alert Rules) 45
3.8 Kịch bản kiểm thử và kết quả 47
3.8.1. Kịch bản giả lập sự cố mất kết nối với website HPU và kết quả 47
3.8.2. Kịch bản giả lập sự cố CPU quá tải và kết quả 50
Chương 4: KẾT QUẢ ĐẠT ĐƯỢC VÀ ĐÁNH GIÁ HIỆU QUẢ CỦA GRAFANA 54
4.1. Kết quả triển khai hệ thống giám sát 54
4.1.1 Kết quả thu được từ Prometheus 54
4.1.2. Kết quả từ Grafana Dashboard 55
4.1.3. Kết quả về hệ thống cảnh báo (Alerting System) 56
4.2. Đánh giá hiệu quả giám sát bằng Grafana 59
4.2.1 Tính trực quan và dễ quan sát 59
4.2.2 Độ chính xác của dữ liệu 59
4.2.3 Khả năng phát hiện và phản ứng với sự cố 59
4.2.4 Lợi ích mang lại cho hệ thống 59
4.3 Hạn chế của hệ thống giám sát 60
`

CHƯƠNG 5: ĐỀ XUẤT CÁC GIẢI PHÁP TỐI ƯU HÓA VÀ NÂNG CAO HIỆU QUẢ GIÁM SÁT. 61
5.1. Tối ưu hóa hệ thống Prometheus – Grafana 61
5.1.1 Tối ưu Prometheus 61
5.1.2 Tối ưu Grafana 61
5.2 Mở rộng hệ thống giám sát 61
5.2.1. Bổ sung các Exporter cần thiết 61
5.2.2. Giám sát nhiều server (Multi-node Monitoring) 61
5.3. Tự động hóa cảnh báo 62
5.3.1. Tích hợp thêm kênh cảnh báo 62
5.3.2. Cảnh báo nâng cao 62
5.4. Nâng cao độ tin cậy và tính ổn định 62
5.4.1. Prometheus HA (High Availability) 62
5.4.2. Tối ưu hạ tầng phần cứng 62
5.4.3. Tăng cường bảo mật 63
5.5. Hướng phát triển trong tương lai 63
KẾT LUẬN 64
TÀI LIỆU THAM KHẢO 66

`

DANH MỤC HÌNH ẢNH

Hình 1. Phần mềm giám sát mạng Grafana 10
Hình 2. Phần mềm giám sát mạng Zabbix 10
Hình 3. Phần mềm giám sát mạng Nagios 11
Hình 4. Phần mềm giám sát mạng PRTG Network 12
Hình 5. Phần mềm giám sát mạng Grafana 14
Hình 6. Phần mềm thu thập và lưu trữ dữ liệu Prometheus 16
Hình 7. Cách thức hoạt động 18
Hình 8. Quy trình cài đặt và cấu hình hệ thống giám sát Grafana và Prometheus 22
Hình 9: Mô hình triển khai thực tế 31
Hình 10: Trạng thái Docker Engine (đã được active) 32
Hình 11: Phiên bản của docker và docker compose 33
Hình 12: Hình ảnh các container đang hoạt động 33
Hình 13: Cấu hình Prometheus Server 35
Hình 14: Cấu hình mục tiêu giám sát 35
Hình 15: Giao diện Prometheus Target Health 36
Hình 16: Giao diện đăng nhập Grafana 37
Hình 17: Giao diện chính Grafana 37
Hình 18: Thêm Data Source 38
Hình 19: Cấu hình URL 38
Hình 20: Nguồn dữ liệu đang hoạt động 39
Hình 21: Panel CPU Usage (%) 40
Hình 22: Panel RAM Usage (%) 41
Hình 23: Panel Network Traffic 42
Hình 24: Panel Website Uptime 42
Hình 25: Dashboard tổng 43
Hình 26: Nơi chưa Webhook URL của kênh Discord 44
Hình 27: Xác nhận kết nối tới Discord Webhook đã được thiết lập thành công 44
Hình 28: Thiết lập cảnh báo Website Down thành công 45
Hình 29: Thiết lập cảnh báo CPU quá tải thành công 46
`

Hình 30: Dashboard chuyển trạng thái sang DOWN màu đỏ..... 48
Hình 31: Kênh Discord nhận tin nhắn FIRING (WEBSITE_HPU_DOWN) 48
Hình 32: Trạng thái trên Dashboard chuyển về UP màu xanh..... 49
Hình 33: Kênh Discord nhận tin nhắn RESOLVED (WEBSITE_HPU_DOWN) 50
Hình 34: Biểu đồ CPU Usage vượt mức 90% 51
Hình 35: Kênh Discord nhận tin nhắn FIRING (CPU_OVERLOAD) 51
Hình 36: Biểu đồ CPU Usage giảm dần dưới mức 90%..... 53
Hình 37: Kênh Discord nhận tin nhắn RESOLVED (CPU_OVERLOAD..... 53
Hình 38: Chứng minh Prometheus sẵn sàng trực quan hóa dữ liệu trên Grafana. 54
Hình 39: Chứng minh Dashboard giám sát tổng thể hoạt động ổn định 55
Hình 40: Chứng minh Dashboard phát hiện và cảnh báo sự cố Website DOWN 57
Hình 41: Chứng minh tin nhắn cảnh báo FIRING được gửi về kênh Discord... 57
Hình 42: Chứng minh Dashboard khôi phục trở lại Website UP..... 58
Hình 43: Chứng minh tin nhắn RESOLVED được gửi về Discord..... 58

`

DANH MỤC BẢNG

Bảng 1 Các lý do cần giám sát 7
Bảng 2. So sánh các công cụ 13
Bảng 3: Các phương thức giám sát 23
Bảng 4: So sánh Grafana với các công cụ khác 27
Bảng 5: Kết quả giám sát từ Grafana 56

`

MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh công nghệ thông tin ngày càng phát triển, việc đảm bảo hệ thống mạng và các dịch vụ trực tuyến hoạt động ổn định, liên tục là một yêu cầu quan trọng đối với các tổ chức, đặc biệt là các cơ sở giáo dục. Việc giám sát trạng thái hệ thống giúp phát hiện sự cố kịp thời, giảm thiểu rủi ro và nâng cao chất lượng vận hành.

Tại trường Đại học Quản lý và Công nghệ Hải Phòng, hệ thống Website và dịch vụ mạng đóng vai trò quan trọng trong quản lý, giảng dạy và hỗ trợ sinh viên. Tuy nhiên, công tác giám sát hiện tại chủ yếu dừng ở việc kiểm tra trạng thái UP/DOWN và còn thiếu khả năng phân tích hiệu năng chuyên sâu theo thời gian.

Grafana và Prometheus là bộ đôi công cụ mã nguồn mở, nổi bật với khả năng thu thập dữ liệu chuỗi thời gian (Time-series data) và trực quan hóa mạnh mẽ. Việc ứng dụng hệ thống này giúp nâng cấp công tác giám sát lên một tầm cao mới: từ giám sát trạng thái sang giám sát hiệu năng thực tế (CPU, RAM, Network Latency).

Vì vậy, em chọn đề tài **"*Nghiên cứu tìm hiểu và ứng dụng Grafana– công cụ giám sát máy chủ và dịch vụ mạng tại trường ĐH Quản lý và Công nghệ Hải Phòng*"** nhằm tìm hiểu, triển khai thử nghiệm và xây dựng một Dashboard giám sát toàn diện cho nhà trường.

2. Hiện trạng hệ thống giám sát tại nhà trường

Tại Trường Đại học Quản lý và Công nghệ Hải Phòng, hệ thống CNTT đang ngày càng được mở rộng với các dịch vụ như website trường, hệ thống đào tạo trực tuyến, máy chủ nội bộ, email, và các dịch vụ khác phục vụ giảng viên, sinh viên và cán bộ.

Tuy nhiên, công tác giám sát các hệ thống này hiện nay chủ yếu dựa vào kiểm tra thủ công hoặc các phương pháp đơn giản, chưa có công cụ chuyên biệt hỗ trợ thu thập các chỉ số hiệu năng (Performance Metrics) như CPU, RAM, I/O của máy chủ theo thời gian thực. Điều này dẫn đến việc chậm trễ trong phát hiện sự cố, thiếu dữ liệu lịch sử để phân tích nguyên nhân gốc rễ, gây ảnh hưởng đến hoạt động của nhà trường và trải nghiệm người dùng.

3. Nhu cầu thực tế

Từ thực trạng trên, nhà trường rất cần một giải pháp giám sát hệ thống chuyên nghiệp, có khả năng:

- Giám sát hiệu năng chuyên sâu (CPU, RAM, Disk, Network) theo dữ liệu chuỗi thời gian.
- Trực quan hóa Dashboard linh hoạt (Grafana) và có khả năng tùy biến cao.
- Cảnh báo tự động, chính xác dựa trên các ngưỡng hiệu năng và có khả năng tích hợp lên đa kênh như Telegram, Discord, Email...
- Không yêu cầu chi phí bản quyền, dễ triển khai và bảo trì.

Bộ đôi Grafana và Prometheus đáp ứng đầy đủ các yêu cầu trên, với ưu điểm nổi bật là cung cấp khả năng truy vấn phức tạp bằng ngôn ngữ PromQL, giúp chuyển đổi dữ liệu thô thành các chỉ số có ý nghĩa và trực quan.

4. Mục tiêu đề tài

Đề tài "*Nghiên cứu tìm hiểu và ứng dụng Grafana– công cụ giám sát máy chủ và dịch vụ mạng tại trường ĐH Quản lý và Công nghệ Hải Phòng*" hướng đến các mục tiêu cụ thể sau:

- Nghiên cứu tổng quan về các công cụ giám sát máy chủ và dịch vụ mạng hiện nay, đặc biệt là Grafana.
- Tìm hiểu nguyên lý hoạt động, các tính năng nổi bật và cách triển khai Grafana.

- Thực hiện triển khai hệ thống Grafana và Prometheus trong môi trường mạng thực tế của nhà trường.
- Xây dựng Dashboard hoàn chỉnh gồm 6 mục giám sát: CPU, RAM, Disk, Network, Website Uptime và Website Latency.
- Đánh giá tính trực quan của Dashboard và hiệu quả của hệ thống trong việc thu thập dữ liệu và cảnh báo sự cố.
- Đề xuất các giải pháp cải tiến và ứng dụng mở rộng sau khi thử nghiệm thành công.

5. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu: Công cụ giám sát Grafana và hệ thống máy chủ, dịch vụ mạng tại trường Đại học Quản lý và Công nghệ Hải Phòng.

Phạm vi nghiên cứu: Tập trung vào cài đặt, cấu hình và xây dựng dashboard giám sát hiệu năng cho một máy chủ và dịch vụ website cụ thể tại trường (ví dụ: website chính thức của nhà trường).

6. Phương pháp nghiên cứu

Tìm hiểu tài liệu: Nghiên cứu tài liệu chính thức và các bài viết liên quan đến Grafana.

Thực nghiệm: Cài đặt và triển khai Hệ thống Grafana/Prometheus trên môi trường mạng thực tế. Phương pháp triển khai được ưu tiên là sử dụng Docker Container trên một máy chủ ảo (VPS) để đảm bảo tính cô lập, linh hoạt và dễ dàng tái tạo môi trường.

Phân tích và đánh giá: Theo dõi kết quả hoạt động, phân tích dữ liệu chuỗi thời gian để đánh giá hiệu quả của dashboard trong việc phát hiện và xử lý sự cố.

CHƯƠNG 1: TỔNG QUAN VỀ GIÁM SÁT MẠNG VÀ HỆ THỐNG MÁY CHỦ

1.1. Tìm hiểu về giám sát mạng và hệ thống máy chủ

1.1.1. Khái niệm

- Giám sát mạng (Network Monitoring) là quá trình theo dõi, thu thập và phân tích dữ liệu về tình trạng hoạt động của các thành phần trong hệ thống mạng và máy chủ. Thông qua việc giám sát, người quản trị có thể nắm bắt được hiệu năng, độ ổn định, khả năng đáp ứng và phát hiện sớm các sự cố tiềm ẩn để đưa ra biện pháp xử lý kịp thời.
- Hoạt động giám sát mạng thường bao gồm việc theo dõi tài nguyên hệ thống (CPU, RAM, dung lượng đĩa, băng thông mạng), trạng thái thiết bị (Router, Switch, Server), cũng như tình trạng của các dịch vụ mạng (Web, Email, Database, …).
- Việc này giúp đảm bảo hệ thống luôn hoạt động ổn định, hạn chế tối đa thời gian gián đoạn và nâng cao chất lượng dịch vụ.
- Hệ thống giám sát mạng thường được xây dựng dựa trên các công nghệ thu thập dữ liệu tự động, lưu trữ và hiển thị thông tin dưới dạng biểu đồ, báo cáo, hoặc cảnh báo thời gian thực. Nhờ đó, người quản trị dễ dàng theo dõi và đánh giá toàn bộ hiệu suất của hạ tầng CNTT trong doanh nghiệp hoặc tổ chức.

1.1.2. Các yếu tố cơ bản trong giám sát mạng

- Để triển khai một hệ thống giám sát hiệu quả, bao gồm cả máy chủ và dịch vụ mạng, cần xác định rõ các yếu tố cốt lõi mang tính nền tảng, được phân thành ba nhóm chính:

a. Đối tượng giám sát:

- Đây là việc xác định phạm vi cần theo dõi, bao gồm:
  - Hệ thống máy chủ: Giám sát tài nguyên cốt lõi (CPU, RAM, Disk I/O).

- Thiết bị mạng: Giám sát router, switch và chất lượng đường truyền (băng thông, độ trễ).
- Dịch vụ ứng dụng: Giám sát trạng thái hoạt động (Uptime), hiệu suất và tỷ lệ lỗi của các ứng dụng nội bộ.

b. Công cụ và giải pháp hỗ trợ:

- Là các phương tiện kỹ thuật dùng để thực hiện việc giám sát, được chia thành hai loại:
  - Phần mềm thương mại: Các giải pháp có bản quyền, cung cấp sự hỗ trợ chuyên nghiệp như SolarWinds, ManageEngine OpManager …
  - Phần mềm mã nguồn mở: Các công cụ linh hoạt, tiết kiệm chi phí (Grafana, Uptime Kuma, …) phù hợp cho việc tùy chỉnh theo cấu trúc mạng đặc thù.

c. Yếu tố con người và quy trình vận hành:

- Công nghệ cần được vận hành bởi quy trình rõ ràng. Yếu tố này bao gồm:
  - Con người: Đội ngũ quản trị hệ thống cần có kiến thức để phân tích dữ liệu và phản ứng với cảnh báo.
  - Quy trình: Thiết lập quy trình vận hành tiêu chuẩn cho việc xử lý sự cố, đảm bảo sự nhất quán và kịp thời khi hệ thống có vấn đề.

1.1.3. Chức năng của hệ thống giám sát

- Hệ thống giám sát hiện đại cần thực hiện chuỗi các chức năng kỹ thuật nhằm đảm bảo quy trình quản lý hiệu suất và sự cố diễn ra liên tục, tự động. Các chức năng cốt lõi bao gồm:

  - **Thu thập và lưu trữ dữ liệu:** Chức năng nền tảng là thu thập các chỉ số hiệu suất (metrics), dữ liệu nhật ký (logs) và lưu trữ chúng trong Cơ sở dữ liệu chuỗi thời gian (time-series database). Việc này đảm bảo dữ liệu được ghi nhận liên tục, chính xác theo thời gian thực, là cơ sở cho các phân tích tiếp theo.

- **Trực quan hóa:** Chức năng thiết yếu giúp chuyển đổi dữ liệu thô thành các biểu đồ, đồ thị trực quan (dashboard). Chức năng này giúp người quản trị dễ dàng nắm bắt tình trạng hệ thống, phát hiện xu hướng bất thường, và đánh giá hiệu suất của máy chủ và dịch vụ mạng chỉ qua giao diện hiển thị.
- **Cảnh báo:** Cơ chế tự động hóa việc phát hiện sự cố. Hệ thống sẽ kiểm tra các chỉ số so với ngưỡng đã thiết lập và tự động kích hoạt, gửi thông báo đến đội ngũ kỹ thuật khi ngưỡng bị vi phạm, đảm bảo thời gian phản ứng nhanh nhất.
- **Phân tích và báo cáo:** Sử dụng dữ liệu lịch sử đã lưu trữ để thực hiện phân tích nguyên nhân gốc rễ sau sự cố. Chức năng này còn hỗ trợ tạo các báo cáo định kỳ về mức độ sử dụng tài nguyên và tính khả dụng của dịch vụ, phục vụ cho việc lập kế hoạch nâng cấp.

1.1.4. Đối tượng và mục tiêu giám sát

- Trong bối cảnh hạ tầng công nghệ thông tin ngày càng phức tạp, việc triển khai một hệ thống giám sát hiệu quả là điều kiện cần thiết để đảm bảo tính liên tục và chất lượng dịch vụ. Bước đầu tiên và quan trọng nhất trong quy trình này là xác định rõ đối tượng giám sát và mục tiêu giám sát.

- Đối tượng giám sát cần bao trùm toàn bộ các lớp của hệ thống, từ phần cứng vật lý, máy chủ, thiết bị mạng cho đến các dịch vụ ứng dụng cung cấp cho người dùng. Việc xác định mục tiêu sẽ giúp định hướng chiến lược thu thập dữ liệu, đảm bảo dữ liệu giám sát thu thập được là thiết yếu cho công tác quản lý và ra quyết định.

Cần giám sát gì (Đối tượng giám sát)	Tại sao (mục tiêu giám sát)
Tính khả dụng của hạ tầng (Router, Switch, Server, …)	Đảm bảo các thành phần chủ chốt luôn trong trạng thái hoạt động và có thể truy cập được, ngăn chặn sự gián đoạn nghiêm trọng.
Tài nguyên Hệ thống (CPU, RAM, Disk I/O, …)	Phát hiện điểm nghẽn để can thiệp tài nguyên kịp thời, duy trì hiệu suất xử lý ổn định.
Hiệu suất dịch vụ (tốc độ phản hồi, tỷ lệ lỗi)	Đảm bảo chất lượng trải nghiệm người dùng đối với các ứng dụng nghiệp vụ chính.
Dữ liệu logs và lưu lượng Mạng	Cung cấp thông tin chi tiết để phân tích nguyên nhân xảy ra sự cố.
An toàn và bảo mật	Phát hiện xâm nhập, tấn công mạng (như DDoS) hoặc các hành vi đáng ngờ để bảo vệ dữ liệu và đảm bảo tính toàn vẹn của hệ thống.

Bảng 1 Các lý do cần giám sát

- Khi một hệ thống được triển khai và đưa vào vận hành, dữ liệu giám sát thu thập được là tài sản quý giá. Việc phát hiện nhanh chóng các bất thường liên quan đến thiết bị, dịch vụ hay tài nguyên hệ thống... thông qua các cảnh báo tự động sẽ giúp đội ngũ kỹ thuật có giải pháp sửa chữa, thay thế hoặc phản ứng kịp thời. Từ đó, đảm bảo hệ thống luôn ổn định, thông suốt và nâng cao hiệu quả vận hành tổng thể toàn bộ hệ thống.

1.1.5. Tầm quan trọng của giám sát mạng

- Tầm quan trọng của giám sát hệ thống mạng và máy chủ vượt ra ngoài khuôn khổ khắc phục sự cố; nó là một yếu tố chiến lược đảm bảo tính liên tục và hiệu quả hoạt động của bất kỳ tổ chức nào. Trong môi trường công nghệ thông tin hiện đại, nơi sự gián đoạn dịch vụ có thể gây ra những hậu quả nghiêm trọng, giám sát chủ động đã trở thành một yêu cầu bắt buộc.
- Việc giám sát hệ thống mạng và máy chủ mang lại lợi ích chiến lược và vận hành, bao gồm:

- **Chuyển đổi từ phản ứng sang chủ động:** Giúp đội ngũ vận hành phát hiện các xu hướng suy giảm hiệu suất và các dấu hiệu bất thường (anomalies) để can thiệp kịp thời, trước khi lỗi hệ thống xảy ra.
- **Tăng cường tính liên tục và độ tin cậy:** Giảm thiểu đáng kể thời gian ngừng hoạt động, đảm bảo các dịch vụ thiết yếu luôn sẵn sàng.
- **Tối ưu hóa tài nguyên và chi phí:** Cung cấp dữ liệu chính xác về mức độ sử dụng tài nguyên, hỗ trợ việc lập kế hoạch năng lực và ra quyết định đầu tư nâng cấp hệ thống một cách hiệu quả.
- **Hỗ trợ phân tích và ra quyết định:** Dữ liệu lịch sử là cơ sở vững chắc để phân tích nguyên nhân xảy ra xự cố và cải thiện hiệu suất lâu dài.
- **Đảm bảo an ninh và bảo mật:** Giám sát lưu lượng và các sự kiện logs để phát hiện sớm các hành vi truy cập hoặc tấn công mạng đáng ngờ.

1.1.6. Các loại hình giám sát

- Giám sát Chủ động (Active Monitoring): là phương pháp mà hệ thống giám sát chủ động gửi truy vấn hoặc tạo giao tiếp tới đối tượng cần theo dõi để kiểm tra trạng thái và hiệu suất.

- Giám sát Thụ động (Passive Monitoring): là phương pháp mà hệ thống giám sát lắng nghe và nhận dữ liệu được gửi tự động từ chính các đối tượng được theo dõi, mà không cần tạo ra truy vấn.

1.1.7. Ưu điểm và nhược điểm của hệ thống giám sát

- Hệ thống giám sát là công cụ không thể thiếu, tuy nhiên, việc triển khai nó luôn đi kèm với những đánh đổi về tài nguyên và kỹ thuật:

❖ **Ưu điểm:**

- Tăng tính ổn định và thời gian hoạt động: Phát hiện sớm các dấu hiệu suy giảm hiệu suất và sự cố, giúp giảm thiểu đáng kể thời gian ngừng hoạt động.
- Hỗ trợ ra quyết định: Cung cấp dữ liệu lịch sử và thời gian thực để phân tích hiệu suất, xác định nút thắt cần tháo gỡ và hỗ trợ việc lập kế hoạch nâng cấp.

- Minh bạch hóa hoạt động: Trực quan hóa trạng thái hệ thống thông qua các dashboard, giúp quản trị viên dễ dàng theo dõi và báo cáo.
- Cải thiện bảo mật: Phát hiện các lưu lượng truy cập hoặc hoạt động đáng ngờ.

❖ **Nhược điểm:**

- Tốn kém tài nguyên: Yêu cầu tài nguyên máy chủ (CPU, RAM, Disk, …) đáng kể để thu thập, lưu trữ và xử lý lượng lớn dữ liệu chuỗi thời gian (time-series data).
- Độ phức tạp trong cấu hình: Việc thiết lập các công cụ, tích hợp nhiều nguồn dữ liệu và cấu hình ngưỡng cảnh báo chính xác đòi hỏi kỹ năng kỹ thuật cao.
- Chi phí bảo trì: Hệ thống giám sát cần được bảo trì và điều chỉnh liên tục để phù hợp với sự thay đổi và mở rộng của hạ tầng.

1.2. Các công cụ giám sát mạng và hệ thống máy chủ phổ biến

- Có rất nhiều giải pháp giám sát mạng trên thị trường:

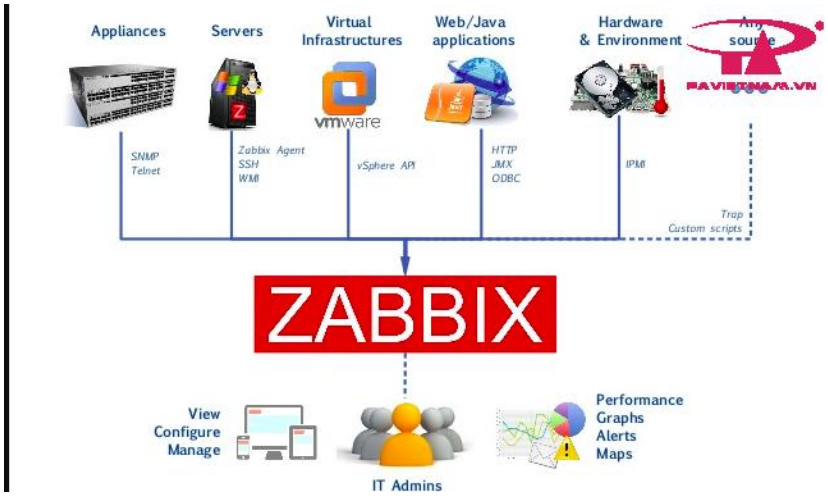
❖ **Phần mềm giám sát mạng Grafana**

- **Grafana** là một nền tảng mã nguồn mở hàng đầu được thiết kế chuyên biệt cho việc trực quan hóa, phân tích và cảnh báo trên dữ liệu chuỗi thời gian (time-series data). Grafana đóng vai trò là giao diện hiển thị thông minh, không tự thu thập dữ liệu mà kết nối và hợp nhất dữ liệu từ nhiều nguồn khác nhau như Prometheus, InfluxDB, …
- Nó cho phép người dùng xây dựng các dashboard trực quan và thiết lập hệ thống cảnh báo mạnh mẽ để giám sát toàn diện hệ thống.

Hình 1. Phần mềm giám sát mạng Grafana

❖ Phần mềm giám sát mạng Zabbix

- **Zabbix** là một công cụ mã nguồn mở nổi tiếng giải quyết cho ta các vấn đề về giám sát – là phần mềm sử dụng các tham số của một mạng, tình trạng và tính toàn vẹn của Server cũng như các thiết bị mạng.
- Zabbix sử dụng một cơ chế thống báo linh hoạt và khả năng tuỳ biến cao cho phép người dùng cấu hình email hoặc sms để cảnh báo dựa trên sự kiện được ta thiết lập sẵn. Ngoài ra Zabbix cung cấp báo cáo và dữ liệu chính xác dựa trên cơ sở dữ liệu.
- Điều này khiến cho Zabbix trở nên lý tưởng hơn, thích hợp phục vụ cho hệ thống mạng tầm trung và lớn của các doanh nghiệp hiện tại với mức chi phí đầu tư vừa phải.
- Tất cả báo cáo, thống kê cũng như cấu hình thông số của Zabbix có thể dễ dàng truy cập qua giao diện web tinh tế đẹp mắt.



Hình 2. Phần mềm giám sát mạng Zabbix

❖ **Phần mềm giám sát mạng Nagios**

- **Nagios** là một ứng dụng phần mềm mã nguồn mở và miễn phí dành cho các hệ thống máy tính. Nó được sử dụng để giám sát hệ thống, mạng và cơ sở hạ tầng.
- Ứng dụng phần mềm này được viết bằng ngôn ngữ C, chủ yếu được thiết kế để chạy trên hệ điều hành Linux. Tuy nhiên nó cũng có thể chạy với hệ điều hành Unix và Windows.
- Nagios được sử dụng để giám sát liên tục các hệ thống, ứng dụng, dịch vụ và quy trình kinh doanh, … Trong trường hợp xảy ra sự cố, Nagios có thể thông báo cho nhân viên kỹ thuật về sự cố, cho phép họ bắt đầu các quy trình khắc phục trước khi sự cố ngừng hoạt động ảnh hưởng đến quy trình kinh doanh, người dùng cuối hoặc khách hàng.

Hình 3. Phần mềm giám sát mạng Nagios

❖ **Phần mềm giám sát mạng PRTG Network**

- Một công cụ giám sát mạng nổi tiếng và được ưa thích rộng rãi như vậy là PRTG. PRTG được định nghĩa là "Trình biểu thị đồ thị lưu lượng bộ định tuyến Paessler".
- Nó là một phần mềm giám sát mạng không có tác nhân (agentless network monitoring software) - Agentless là giải pháp không yêu cầu cài đặt Agent, phân tích mạng dựa trên giám sát package, được sử dụng để giám sát tính sẵn sàng của mạng và hiệu suất.

- PRTG cũng thu thập số liệu thống kê từ các máy chủ như bộ chuyển mạch, bộ định tuyến và máy chủ**.** Phần mềm giám sát dễ sử dụng và có giao diện đồ họa trực quan. PRTG Network Monitor theo dõi hiệu suất mạng, thông lượng, băng thông và các yếu tố khác.

Hình 4. Phần mềm giám sát mạng PRTG Network

So sánh các công cụ

Công cụ	Ưu điểm	Nhược điểm
Grafana	Giao diện trực quan tuyệt vời, khả năng tùy chỉnh dashboard cao, kết nối đa dạng với hầu hết các nguồn dữ liệu (Prometheus, Zabbix, SQL, v.v.), mạnh mẽ về cảnh báo.	Không tự thu thập/lưu trữ dữ liệu, yêu cầu phải tích hợp với các hệ thống backend khác để hoạt động.
Zabbix	Mạnh mẽ, hỗ trợ giám sát chi tiết toàn bộ hệ thống (all-in-one), có sẵn cơ chế thu thập và lưu trữ.	Khó cấu hình, giao diện phức tạp hơn so với Grafana, đòi hỏi tài nguyên lớn khi hệ thống mở rộng.
Nagios	Hỗ trợ giám sát toàn diện, là nền tảng mã nguồn mở lâu đời, đáng tin cậy.	Giao diện cũ, cấu hình phức tạp qua file văn bản, cần nhiều tài nguyên.
PRTG Network	Chuyên giám sát hạ tầng, hiển thị biểu đồ trực quan, dễ cài đặt cho môi trường Windows.	Có giới hạn về số lượng "cảm biến" (sensors) trong phiên bản miễn phí, chi phí cao khi mở rộng.

Bảng 2. So sánh các công cụ

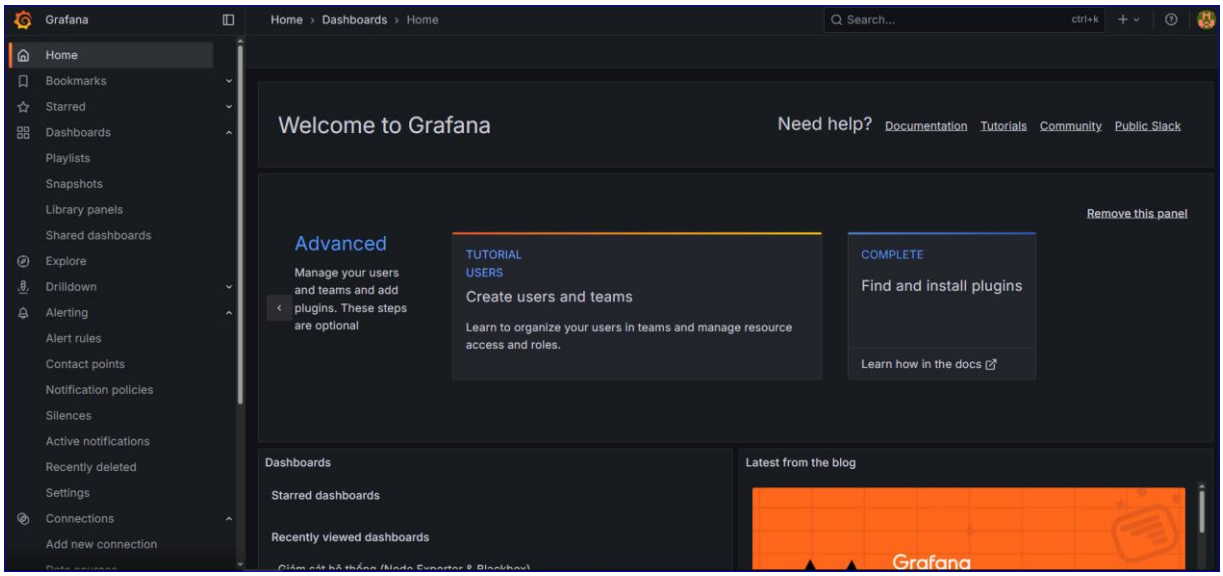
So sánh nhanh:

- Grafana: Trực quan hóa dữ liệu hàng đầu, tích hợp linh hoạt với mọi nguồn dữ liệu (Data Source Agnostic), phù hợp nhất cho vai trò hiển thị Dashboard và cảnh báo.
- Nagios, Zabbix: Mạnh mẽ, phù hợp giám sát toàn bộ hệ thống (All-in-one), tích hợp khả năng thu thập và lưu trữ dữ liệu.
- PRTG: Giao diện trực quan, tập trung vào giám sát hiệu suất mạng và thiết bị hạ tầng.

CHƯƠNG 2: TÌM HIỂU VỀ HỆ THỐNG GIÁM SÁT GRAFANA VÀ PROMETHEUS

2.1. Giới thiệu Grafana

- **Grafana** là một nền tảng mã nguồn mở (Open-Source) hàng đầu, được thiết kế chuyên biệt cho việc trực quan hóa, phân tích và tương tác với dữ liệu chuỗi thời gian (time-series data). Được phát triển và duy trì bởi Grafana Labs, công cụ này đã nhanh chóng trở thành tiêu chuẩn công nghiệp cho việc xây dựng các Dashboard, giúp người quản trị hệ thống dễ dàng theo dõi hiệu suất và tình trạng sức khỏe của toàn bộ hạ tầng.
- Bản chất của Grafana là một giao diện người dùng (User Interface - UI) thông minh, hoạt động độc lập với nguồn dữ liệu. Điều này đồng nghĩa với việc Grafana không tự thu thập hay lưu trữ dữ liệu; thay vào đó, nó kết nối với các nguồn dữ liệu bên ngoài (data source) thông qua các API chuyên biệt.
- Vai trò chính của Grafana là thống nhất dữ liệu từ nhiều nguồn khác nhau (như Prometheus, Zabbix, SQL, v.v.) và chuyển đổi chúng thành các biểu đồ, đồ thị sinh động, phục vụ cho công tác giám sát chủ động và phân tích lỗi.



Hình 5. Phần mềm giám sát mạng Grafana

2.2. Thông tin và tính năng Grafana:

- Các tính năng của Grafana được xây dựng để cung cấp một nền tảng quan sát toàn diện, linh hoạt và mạnh mẽ:

- **Khả năng kết nối đa dạng:** Grafana có thể kết nối và trực quan hóa dữ liệu từ hầu hết các nguồn dữ liệu phổ biến như Prometheus (Metrics), Loki (Logs), InfluxDB, các cơ sở dữ liệu SQL (MySQL, PostgreSQL) và các dịch vụ Cloud khác.
- **Giao diện dashboard trực quan và linh hoạt:** Cung cấp giao diện người dùng trực quan, nhanh chóng, hỗ trợ tùy chỉnh cao thông qua nhiều loại Panel khác nhau (Biểu đồ đường, Gauge, Heatmap, Bảng...).
- **Tính năng mẫu hóa:** Cho phép xây dựng các dashboard động bằng cách sử dụng các biến (Variables), giúp tái sử dụng một Dashboard cho việc giám sát không giới hạn số lượng máy chủ hoặc dịch vụ.
- **Cơ chế cảnh báo nâng cao:** Hỗ trợ thiết lập các quy tắc cảnh báo phức tạp, đa điều kiện và gửi thông báo qua nhiều kênh phổ biến như Email, Discord, Microsoft Teams …
- **Trực quan hóa dữ liệu thời gian thực:** Có khả năng hiển thị dữ liệu gần thời gian thực (Near Real-time) với độ phân giải tùy chỉnh, giúp theo dõi các sự kiện đang diễn ra.
- **Tính năng mở rộng:** Sở hữu thư viện Plugin phong phú, cho phép bổ sung các loại Panel mới, Data Source mới hoặc các tính năng nghiệp vụ chuyên biệt khác.
- **Hỗ trợ đa ngôn ngữ:** Cung cấp nhiều tùy chọn ngôn ngữ cho giao diện người dùng.
- **Xác thực mạnh mẽ:** Hỗ trợ các phương thức xác thực cho tổ chức như LDAP**,** OAuth, SAML và xác thực đa yếu tố (2FA/Multi-factor Authentication).
- **Tích hợp quản lý:** Hỗ trợ cấu hình Proxy và Reverse Proxy, giúp tích hợp Grafana vào môi trường mạng và bảo mật của tổ chức một cách an toàn.

2.3 Giới thiệu Prometheus

- **Prometheus** là một bộ công cụ giám sát và cảnh báo hệ thống mã nguồn mở (Open-Source), ban đầu được phát triển bởi SoundCloud vào năm 2012. Hiện nay, Prometheus là một dự án độc lập và được duy trì bởi Cloud Native Computing

Foundation (CNCF), trở thành tiêu chuẩn vàng trong việc giám sát các hệ thống phân tán và kiến trúc vi dịch vụ (microservices).

- Khác với các hệ thống giám sát truyền thống, Prometheus hoạt động theo mô hình Pull-based (kéo dữ liệu). Thay vì đợi các thiết bị gửi dữ liệu về, Prometheus chủ động kết nối đến các mục tiêu giám sát (targets) theo chu kỳ định trước để thu thập các chỉ số (metrics).
- Toàn bộ dữ liệu thu thập được sẽ được lưu trữ trong một cơ sở dữ liệu chuỗi thời gian (Time Series Database - TSDB) hiệu năng cao do chính Prometheus quản lý, cho phép truy vấn và phân tích dữ liệu lịch sử một cách nhanh chóng và chính xác.

Hình 6. Phần mềm thu thập và lưu trữ dữ liệu Prometheus

2.4 Thông tin và tính năng của Prometheus

- Các tính năng của Prometheus được thiết kế tối ưu cho độ tin cậy và khả năng mở rộng trong môi trường giám sát hiện đại:

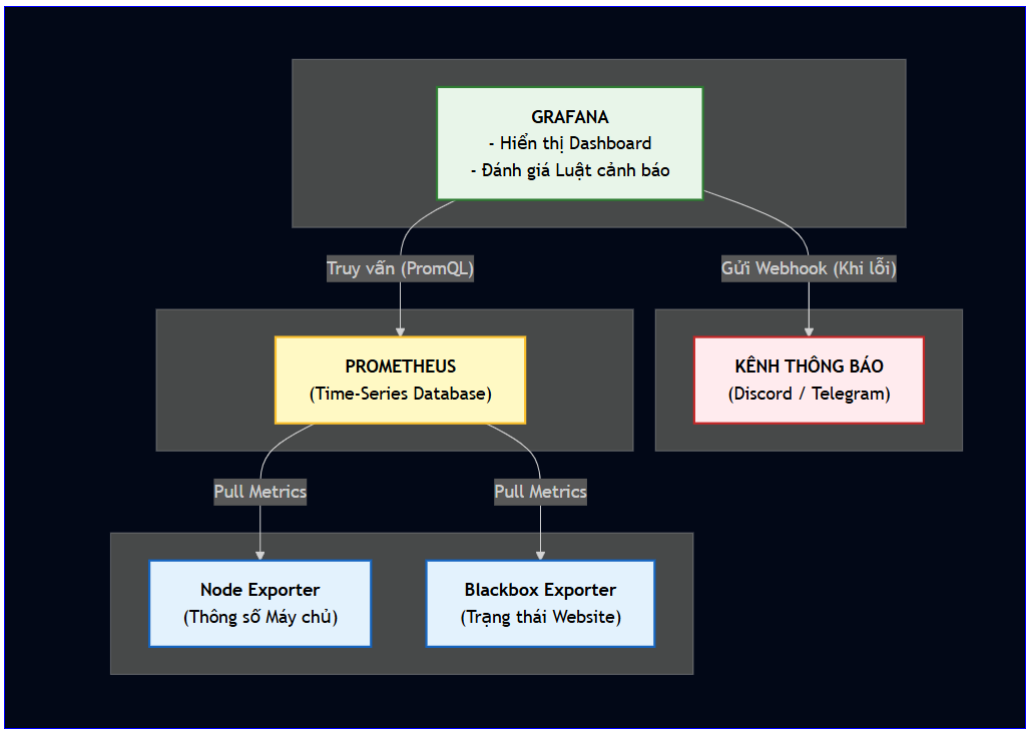
- **Mô hình dữ liệu đa chiều (Multi-dimensional Data Model):** Dữ liệu được lưu trữ dưới dạng chuỗi thời gian (time-series), được định danh bởi tên chỉ số (metric name) và các cặp khóa-giá trị (key-value pairs) gọi là nhãn (labels). Điều này cho phép phân tích dữ liệu linh hoạt theo nhiều chiều hướng khác nhau.
- **Ngôn ngữ truy vấn mạnh mẽ (PromQL):** Prometheus cung cấp ngôn ngữ truy vấn riêng biệt là PromQL (Prometheus Query Language).

PromQL cho phép người dùng thực hiện các phép toán tổng hợp, lọc, và tính toán phức tạp trên dữ liệu time-series để tạo ra các biểu đồ và cảnh báo chính xác.

- **Cơ chế thu thập dữ liệu chủ động (Pull Model):** Prometheus chủ động thu thập dữ liệu qua giao thức HTTP, giúp việc giám sát trở nên đơn giản hơn khi không cần cài đặt agent phức tạp đẩy dữ liệu đi, đồng thời dễ dàng phát hiện nếu một dịch vụ mục tiêu bị "chết" (không thể pull được).
- **Khả năng tự động phát hiện mục tiêu (Service Discovery):** Hỗ trợ tích hợp với các hệ thống quản lý dịch vụ như Kubernetes, Consul, EC2... để tự động phát hiện và giám sát các dịch vụ mới mà không cần cấu hình thủ công liên tục.
- **Lưu trữ cục bộ hiệu quả:** Prometheus không phụ thuộc vào hệ thống lưu trữ phân tán phức tạp; các node máy chủ Prometheus hoạt động độc lập và tự chủ về lưu trữ, giúp hệ thống dễ vận hành và ít điểm lỗi hơn.
- **Hỗ trợ Push Gateway:** Đối với các tác vụ ngắn hạn (short-lived jobs) không thể chờ Prometheus đến kéo dữ liệu, Prometheus hỗ trợ push gateway để các tác vụ này chủ động đẩy metrics lên trước khi kết thúc.
- **Hệ thống cảnh báo (Alerting):** Quản lý các quy tắc cảnh báo (alert rules) dựa trên PromQL và gửi thông báo đến Alertmanager để xử lý, gộp nhóm và định tuyến thông báo đến Email, Slack, Discord, …

2.5 Cách thức hoạt động

- Như đã đề cập, Grafana chỉ đóng vai trò là giao diện hiển thị và không tự lưu trữ dữ liệu. Do đó, để hoạt động được, Grafana cần kết hợp với một cơ sở dữ liệu mạnh mẽ phía sau. Trong đồ án này sử dụng Prometheus làm hệ thống thu thập và lưu trữ dữ liệu trung tâm.



Hình 7. Cách thức hoạt động

- Sơ đồ hoạt động gồm các thành phần sau:
  - **Nguồn dữ liệu (Data Sources / Exporters):**
    - Là các thành phần chịu trách nhiệm thu thập dữ liệu thô từ các đối tượng cần giám sát.
    - Node Exporter: Được cài đặt trên máy chủ để thu thập các thông số về phần cứng và hệ điều hành như: mức sử dụng CPU, dung lượng RAM, ổ cứng và lưu lượng mạng.
    - Blackbox Exporter: Thực hiện việc thăm dò từ bên ngoài để kiểm tra tính sẵn sàng (uptime) và độ trễ của các dịch vụ thông qua các giao thức như HTTP, HTTPS, DNS, TCP...
  - **Hệ thống thu thập (Prometheus):**
    - Đóng vai trò là trung tâm lưu trữ cơ sở dữ liệu chuỗi thời gian (Time Series Database).
    - Thực hiện cơ chế Pull (kéo dữ liệu) để định kỳ lấy dữ liệu từ các Exporters (Node Exporter và Blackbox Exporter) về lưu trữ và xử lý.

- Cung cấp ngôn ngữ truy vấn (PromQL) để trích xuất dữ liệu phục vụ cho việc hiển thị.

- **Hệ thống hiển thị và Cảnh báo (Grafana):**
  - Là giao diện người dùng (Dashboard) chính, nơi người quản trị theo dõi trạng thái hệ thống.
  - Kết nối tới Prometheus để truy vấn dữ liệu và trực quan hóa dưới dạng các biểu đồ, đồ thị sinh động.
  - Cho phép cấu hình các quy tắc cảnh báo (Alert Rules) dựa trên ngưỡng giá trị (ví dụ: CPU > 90% hoặc Website bị Down).

- **Hệ thống nhận cảnh báo (Alerting System):**
  - Là các kênh tiếp nhận thông báo cuối cùng khi có sự cố xảy ra.
  - Khi Grafana phát hiện vi phạm quy tắc cảnh báo, nó sẽ gửi thông báo đến người quản trị thông qua các kênh tích hợp như Email, Telegram, Webhook…

2.6 Cách thức hoạt động từng bước:

- **Truy vấn dữ liệu (Query):** Grafana tự động gửi các yêu cầu truy vấn (Queries) đến nguồn dữ liệu đã cấu hình (ở đây là Prometheus) để kéo về các chỉ số (metrics) hoặc dữ liệu log cần thiết cho việc hiển thị và đánh giá.
- **Xử lý và trực quan hóa (Processing & Visualization):** Grafana tiếp nhận dữ liệu đã kéo về và xử lý chúng (ví dụ: tính toán tổng hợp, chuyển đổi đơn vị), sau đó chuyển đổi dữ liệu thô thành các thành phần trực quan (Panels) như biểu đồ và đồ thị, hiển thị trên Dashboard.
- **Đánh giá luật cảnh báo (Rule Evaluation):** Grafana liên tục chạy các luật cảnh báo (Alert Rules) đã được thiết lập, đánh giá giá trị hiện tại của dữ liệu truy vấn được so với các ngưỡng cảnh báo (thresholds).
- **Kích hoạt và gửi cảnh báo (Alerting & Notification):** Khi luật cảnh báo bị vi phạm (thay đổi trạng thái từ OK sang Alerting), Grafana sẽ kích hoạt hệ

thống cảnh báo và gửi thông báo tức thời đến các kênh liên lạc đã cấu hình (Webhook, Email, Discord, …) để thông báo sự cố.

2.6 Quy trình cài đặt và cấu hình cơ bản

- Quy trình triển khai hệ thống giám sát hiệu năng dựa trên Grafana và Prometheus trên nền tảng Docker được thực hiện theo một trình tự tiêu chuẩn gồm 5 bước chính. Quy trình này đảm bảo tính hệ thống, khả năng mở rộng và dễ dàng trong việc quản lý, bảo trì.

Bước 1: Chuẩn bị môi trường triển khai (Environment Setup) Đây là bước khởi tạo hạ tầng cơ sở cho hệ thống.

- **Hạ tầng:** Sử dụng một máy chủ (VPS hoặc máy chủ vật lý) chạy hệ điều hành Linux (như Ubuntu Server hoặc CentOS) để đảm bảo tính ổn định và tối ưu hóa tài nguyên.
- **Nền tảng Container hóa:** Cài đặt Docker Engine để cung cấp môi trường chạy ứng dụng dưới dạng container cô lập, giúp tối ưu hiệu suất so với máy ảo truyền thống. Đồng thời, cài đặt Docker Compose để quản lý và điều phối việc khởi chạy nhiều container cùng lúc thông qua file cấu hình.

Bước 2: Thiết kế kiến trúc và Cấu hình dịch vụ (Service Orchestration) Hệ thống được định nghĩa dưới dạng mã (Infrastructure as Code) thông qua file điều phối docker-compose.yml, thay vì cài đặt thủ công từng phần mềm rời rạc.

- **Khai báo dịch vụ:** Xác định các container thành phần cần thiết bao gồm: Prometheus (thu thập dữ liệu), Grafana (hiển thị), Node Exporter (giám sát tài nguyên máy chủ) và Blackbox Exporter (giám sát dịch vụ website).
- **Thiết lập mạng (Network):** Cấu hình mạng nội bộ ảo để các dịch vụ có thể giao tiếp bảo mật với nhau mà không cần mở cổng công khai không cần thiết.

- **Quản lý lưu trữ (Volumes):** Cấu hình các phân vùng lưu trữ bền vững (Persistent Volumes) cho Prometheus và Grafana để đảm bảo dữ liệu lịch sử và cấu hình Dashboard không bị mất khi khởi động lại hệ thống.

Bước 3: Cấu hình thu thập dữ liệu (Data Collection Configuration) Đây là bước thiết lập logic hoạt động cho "bộ não" Prometheus thông qua file cấu hình prometheus.yml.

- **Định nghĩa Job:** Khai báo các tác vụ (Jobs) giám sát cụ thể. Mỗi Job sẽ chỉ định địa chỉ (IP/Port) của các Exporter mục tiêu.
- **Chu kỳ thu thập (Scrape Interval):** Thiết lập tần suất Prometheus lấy dữ liệu từ các Exporter (ví dụ: mỗi 15 giây hoặc 1 phút), đảm bảo độ mịn của dữ liệu phù hợp với nhu cầu giám sát.

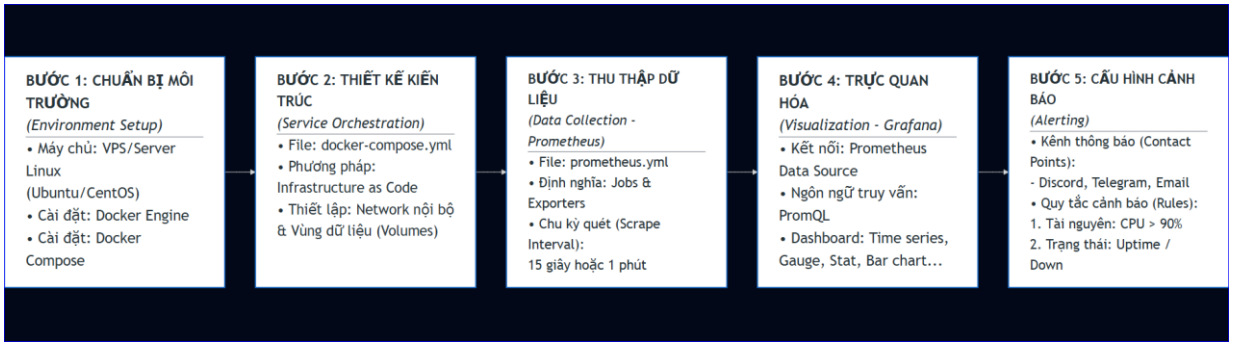
Bước 4: Cấu hình hiển thị và Trực quan hóa (Visualization Setup) Sau khi dữ liệu đã được thu thập về Prometheus, bước này tập trung vào việc biến dữ liệu thô thành thông tin có ý nghĩa trên Grafana.

- **Kết nối Data Source:** Thiết lập Prometheus làm nguồn dữ liệu chính trong Grafana.
- **Xây dựng Dashboard:** Sử dụng ngôn ngữ truy vấn PromQL để trích xuất dữ liệu và hiển thị lên các Panel trực quan như: Biểu đồ đường (Time series) cho CPU/RAM, Biểu đồ trạng thái (Stat) cho Uptime, và Đồng hồ đo (Gauge) cho dung lượng đĩa.

Bước 5: Cấu hình Cảnh báo tự động (Alerting Configuration) Bước cuối cùng nhằm hoàn thiện tính chủ động của hệ thống giám sát.

- **Kênh thông báo (Contact Points):** Cấu hình các kênh nhận cảnh báo như Discord, Telegram hoặc Email để gửi thông tin sự cố đến người quản trị.

- **Quy tắc cảnh báo (Alert Rules):** Thiết lập các điều kiện kích hoạt cảnh báo dựa trên hai nhóm chỉ số chính:
    - Cảnh báo hiệu năng: Khi tài nguyên hệ thống bị quá tải (ví dụ: CPU > 90%).
    - Cảnh báo trạng thái (Uptime/Down): Khi dịch vụ hoặc website không phản hồi hoặc phản hồi lỗi (Website Down), giúp phát hiện ngay lập tức sự cố gián đoạn dịch vụ



Hình 8. Quy trình cài đặt và cấu hình hệ thống giám sát Grafana và Prometheus

2.7 Các phương thức giám sát trong hệ thống (Grafana & Prometheus)

- Để đảm bảo khả năng giám sát toàn diện từ tài nguyên máy chủ đến trạng thái website, hệ thống sử dụng kết hợp các công cụ thu thập dữ liệu chuyên biệt với phương thức hoạt động cụ thể như sau:

Công cụ thu thập	Loại hình giám sát	Mô tả kỹ thuật	Ứng dụng thực tế
Node Exporter	Giám sát tài nguyên máy chủ	Thu thập các số liệu chi tiết từ nhân hệ điều hành Linux để đánh giá sức khỏe và hiệu năng xử lý của phần cứng.	**1. CPU:** Theo dõi tỷ lệ phần trăm vi xử lý đang hoạt động. **2. RAM:** Theo dõi dung lượng bộ nhớ đã sử dụng. **3. Băng thông:** Đo lường tốc độ truyền và nhận dữ liệu của mạng.
Blackbox Exporter	Giám sát tính sẵn sàng	Thực hiện cơ chế thăm dò từ bên ngoài thông qua giao thức HTTP để kiểm tra trạng thái phản hồi của trang web.	**4. Uptime:** Xác định trạng thái hoạt động của website (đang chạy hay bị gián đoạn) theo thời gian thực.

Bảng 3: Các phương thức giám sát

2.8 Ưu điểm và nhược điểm của hệ thống giám sát

- Việc kết hợp Grafana và Prometheus tạo nên một kiến trúc giám sát hiện đại, tận dụng được thế mạnh của cả hai công cụ: khả năng thu thập dữ liệu mạnh mẽ của Prometheus và khả năng trực quan hóa linh hoạt của Grafana. Tuy nhiên, bên cạnh những thế mạnh về công nghệ, việc triển khai công cụ này cũng đi kèm với những yêu cầu kỹ thuật và thách thức vận hành đặc thù cần được xem xét kỹ lưỡng.

2.8.1 Ưu điểm

- **Kiến trúc phân tán và linh hoạt:** Hệ thống tách biệt hoàn toàn lớp thu thập dữ liệu (Prometheus) và lớp hiển thị (Grafana).
    - Độc lập nguồn dữ liệu (Data Source Agnostic): Grafana không ép buộc lưu trữ dữ liệu theo định dạng cố định mà có thể kết nối đồng thời tới nhiều nguồn (Prometheus, MySQL, Loki) trên cùng một

Dashboard. Điều này tạo ra khả năng quản lý tất cả dữ liệu từ hạ tầng đến ứng dụng trên một màn hình duy nhất.

- Tính ổn định: Nếu giao diện Grafana gặp lỗi, dữ liệu nền tảng vẫn được Prometheus thu thập liên tục mà không bị gián đoạn.

- **Khả năng tùy biến hiển thị chuyên sâu:** Grafana cung cấp quyền kiểm soát chi tiết đối với việc hiển thị dữ liệu, vượt trội hơn các biểu đồ tĩnh truyền thống.
    - Đa dạng Panel: Hỗ trợ Heatmaps, Histograms, Geomaps và Node Graph để vẽ sơ đồ mạng.
    - Biến (Variables): Tính năng tạo biến giúp Dashboard trở nên "động" (Dynamic). Người dùng có thể chuyển đổi việc giám sát giữa hàng trăm máy chủ khác nhau chỉ bằng một thao tác chọn menu thả xuống mà không cần thiết kế lại từ đầu.
- **Hiệu năng xử lý và ngôn ngữ truy vấn mạnh mẽ:**
    - Hiệu năng cao: Mô hình lưu trữ chuỗi thời gian (Time-series) của Prometheus được tối ưu hóa để xử lý hàng triệu điểm dữ liệu, cho phép giám sát tần suất cao (ví dụ: 10 giây/lần) mà không làm quá tải tài nguyên.
    - PromQL linh hoạt: Khi kết hợp với Prometheus, Grafana cho phép thực thi các câu lệnh PromQL phức tạp ngay trên lớp hiển thị mà không cần thay đổi cấu trúc dữ liệu gốc.
- **Hệ sinh thái cộng đồng phong phú**:
    - Là phần mềm mã nguồn mở phổ biến nhất trong lĩnh vực quan sát, Grafana sở hữu kho thư viện dashboard khổng lồ.
    - Người quản trị có thể tải về và sử dụng ngay các mẫu dashboard được cộng đồng tối ưu sẵn (ví dụ: Node Exporter Full, Blackbox Exporter), giúp tiết kiệm đến 90% thời gian cấu hình.

2.8.2 Nhược điểm

- **Rào cản kỹ thuật về ngôn ngữ truy vấn:**

- Grafana không phải là công cụ dành cho người dùng phổ thông không có nền tảng kỹ thuật. Để xây dựng được các Dashboard chuyên sâu và có giá trị phân tích cao, người quản trị bắt buộc phải am hiểu và thành thạo ngôn ngữ truy vấn PromQL (Prometheus Query Language).
- Việc viết sai câu lệnh truy vấn không chỉ dẫn đến hiển thị dữ liệu sai lệch mà còn có thể gây quá tải cho hệ thống thu thập dữ liệu. Đây là một rào cản lớn so với các công cụ sử dụng giao diện "kéo-thả" đơn giản.

- **Độ phức tạp trong vận hành và quản trị:**
  - Khác với các giải pháp giám sát "nguyên khối" (All-in-one) như PRTG hay Zabbix – nơi mọi tính năng được đóng gói sẵn trong một bộ cài đặt duy nhất, hệ thống Grafana - Prometheus hoạt động dựa trên kiến trúc vi dịch vụ (Microservices). Có nghĩa là kiến trúc phần mềm chia ứng dụng lớn thành nhiều dịch vụ nhỏ, độc lập, mỗi dịch vụ xử lý một chức năng cụ thể. Các dịch vụ này giao tiếp với nhau qua API
  - Điều này đòi hỏi người quản trị phải quản lý và duy trì nhiều thành phần rời rạc (các Docker container, các file cấu hình YAML riêng biệt cho từng dịch vụ). Việc cấu hình, nâng cấp hoặc khắc phục sự cố do đó sẽ phức tạp hơn và yêu cầu sự hiểu biết sâu sắc về sự tương tác giữa các thành phần.
- **Hạn chế trong lưu trữ dữ liệu dài hạn:**
  - Prometheus được thiết kế để theo dõi dữ liệu "nóng" (dữ liệu quan trọng, hay sử dụng) trong thời gian ngắn, nên mặc định nó chỉ giữ lại dữ liệu trong khoảng 15-30 ngày, sau đó dữ liệu sẽ bị xóa đi để tiết kiệm dung lượng và tối ưu hóa hiệu suất.
  - Grafana chỉ đóng vai trò hiển thị dữ liệu từ một nguồn (như Prometheus), còn mọi dữ liệu lịch sử đều được lưu trữ bên trong Prometheus. Do đó, nếu Prometheus bị mất dữ liệu, Grafana không

có dữ liệu để lấy và hiển thị, dẫn đến việc không thể xem lại được lịch sử đã mất.

- Để đáp ứng nhu cầu lưu trữ và phân tích dữ liệu lịch sử trong dài hạn (vài tháng hoặc vài năm), hệ thống cần phải tích hợp thêm các giải pháp lưu trữ mở rộng phức tạp như Thanos hoặc Cortex, làm tăng thêm gánh nặng quản trị cho hệ thống.

2.9 So sánh Grafana với các công cụ khác

- Để làm rõ cơ sở lựa chọn công nghệ cho đồ án, chúng ta cần xem xét sự khác biệt về tư duy kiến trúc và phương pháp xử lý dữ liệu. Bảng dưới đây sẽ đối chiếu chi tiết các đặc tính kỹ thuật giữa mô hình Grafana kết hợp Prometheus so với các giải pháp giám sát nguyên khối (Monolithic).

Tiêu chí kỹ thuật	Mô hình Grafana + Prometheus	Mô hình Zabbix / PRTG
Kiến trúc hệ thống	**Decoupled (Tách rời):** Lớp hiển thị (Grafana) và lớp dữ liệu (Prometheus) là hai phần mềm riêng biệt, giao tiếp qua API.	**Monolithic (Nguyên khối):** Thu thập, lưu trữ, xử lý và hiển thị đều nằm trong một phần mềm duy nhất cài đặt trên máy chủ.
Khả năng mở rộng	**Cao:** Có thể dễ dàng thay thế hoặc nâng cấp từng thành phần mà không ảnh hưởng đến toàn bộ hệ thống. Có thể dùng 1 Grafana để soi dữ liệu từ 10 Prometheus khác nhau.	**Trung bình:** Khó mở rộng. Khi hệ thống lớn, database (thường là SQL) dễ bị nghẽn, buộc phải nâng cấp toàn bộ máy chủ hoặc thiết lập phân tán phức tạp.
Mô hình dữ liệu	**Time-series (Chuỗi thời gian):** Tối ưu hóa cho việc lưu trữ hàng triệu điểm dữ liệu theo thời gian thực.	**Relational DB (Quan hệ):** Thường dùng MySQL/PostgreSQL để lưu trữ, tốt cho dữ liệu có cấu trúc nhưng nặng nề khi xử lý lượng lớn metrics.
Cơ chế thu thập	**Pull Model (Kéo):** Server chủ động đi lấy dữ liệu từ các máy con. Giúp kiểm soát tải tốt hơn.	**Push Model (Đẩy) / Agent:** Máy con liên tục đẩy dữ liệu về Server. Dễ gây "ngập lụt" (DDOS) server nếu không cấu hình kỹ.
Kết luận áp dụng	Phù hợp với môi trường Cloud, Container (Docker), Microservices và các hệ thống cần tính linh hoạt cao như tại HPU.	Phù hợp với môi trường mạng truyền thống, hệ thống máy chủ vật lý cố định, ít thay đổi.

Bảng 4: So sánh Grafana với các công cụ khác

Kết luận: Qua phân tích so sánh, mô hình Grafana kết hợp Prometheus đã chứng minh sự vượt trội về kiến trúc linh hoạt và khả năng phân tích dữ liệu chuyên sâu, đáp ứng đúng nhu cầu chuyển dịch từ giám sát trạng thái sang quản trị hiệu năng tại HPU. Đây là cơ sở then chốt để đề tài tiến hành xây dựng và áp dụng giải pháp vào môi trường thực tế, với quy trình triển khai chi tiết sẽ được trình bày trong chương sau.

CHƯƠNG 3: TRIỂN KHAI HỆ THỐNG GIÁM SÁT GRAFANA TẠI NHÀ TRƯỜNG.

3.1. Mục tiêu triển khai

- Việc triển khai hệ thống giám sát dựa trên Grafana và Prometheus tại Trường Đại học Quản lý và Công nghệ Hải Phòng hướng tới các mục tiêu cụ thể sau:

- **Xây dựng hệ thống giám sát tập trung:** Thiết lập một nền tảng quản trị thống nhất, cho phép theo dõi đồng thời trạng thái của cả hạ tầng máy chủ và các dịch vụ website trên cùng một giao diện Dashboard trực quan.
- **Giám sát hiệu năng tài nguyên (Resource Monitoring):** Thu thập và phân tích các chỉ số phần cứng quan trọng như tỷ lệ sử dụng CPU, dung lượng RAM và băng thông mạng theo thời gian thực để đánh giá độ ổn định và phát hiện sớm nguy cơ quá tải.
- **Đảm bảo tính sẵn sàng của dịch vụ (Availability Monitoring):** Kiểm tra liên tục trạng thái hoạt động (Uptime) của website nhà trường nhằm phát hiện ngay lập tức các sự cố gián đoạn dịch vụ.
- **Tự động hóa quy trình cảnh báo:** Thiết lập cơ chế gửi thông báo sự cố tức thời qua kênh Discord, giúp đội ngũ kỹ thuật phản ứng nhanh chóng, giảm thiểu tối đa thời gian ngừng hoạt động của hệ thống.

3.2. Dịch vụ giám sát trong nhà trường

- Căn cứ vào nhu cầu quản trị thực tế và phạm vi của đồ án, hệ thống được cấu hình để giám sát hai nhóm đối tượng cốt lõi tại trường Đại học Quản lý và Công nghệ Hải Phòng:

a) Nhóm hạ tầng máy chủ

- **Đối tượng giám sát:** Máy chủ ảo (VPS) chạy hệ điều hành Linux, nơi vận hành các dịch vụ nội bộ và nền tảng Docker.

- **Mục tiêu:** Theo dõi sát sao tài nguyên phần cứng để đảm bảo máy chủ không bị quá tải.
- **Các chỉ số giám sát:**
    - CPU: Tỷ lệ phần trăm xử lý của vi xử lý.
    - RAM: Dung lượng bộ nhớ đang được sử dụng.
    - Băng thông mạng (Network Traffic): Lưu lượng dữ liệu truyền và nhận qua card mạng.
- **Tần suất thu thập dữ liệu:** 15 giây/lần.
- **Kênh cảnh báo:** Discord (khi CPU > 90%).

b) Nhóm dịch vụ ứng dụng

- **Đối tượng giám sát:** Website chính thức của nhà trường.
- **Địa chỉ: https://hpu.edu.vn/**
- **Mục tiêu:** Đảm bảo tính sẵn sàng phục vụ của cổng thông tin đối với sinh viên và giảng viên.
- **Kiểu giám sát:** Trạng thái
- **Chỉ số giám sát:** Uptime (Trạng thái hoạt động)
- **Tần suất kiểm tra:** 60 giây/lần.
- **Kênh cảnh báo:** Discord (khi Website không phản hồi).

3.3. Mô hình triển khai thực tế

- Mô hình triển khai thực tế của hệ thống giám sát được xây dựng tập trung trên một máy chủ ảo (VPS) duy nhất, sử dụng nền tảng **Container hóa (Docker)** để tối ưu tài nguyên và quản lý.

- Kiến trúc vận hành dựa trên sự phối hợp chặt chẽ giữa **Prometheus** (đảm nhiệm thu thập dữ liệu tập trung) và **Grafana** (đảm nhiệm hiển thị Dashboard và cảnh báo), qua đó tạo nên một quy trình giám sát khép kín và tự động từ lớp hạ tầng máy chủ đến người quản trị.

Hình 9: Mô hình triển khai thực tế

- Kiến trúc hệ thống bao gồm 4 tầng chính và các thông số thực tế:

- **Tầng hạ tầng và nguồn dữ liệu (Targets):**
  - **Máy chủ VPS:** Nền tảng vật lý chạy hệ thống.
  - **Website HPU:** Mục tiêu giám sát.
  - **Node Exporter (Cổng 9101):** Thu thập các thông số máy chủ (CPU, RAM, tốc độ băng thông)
  - **Blackbox Exporter (Cổng 9115):** Thu thập trạng thái và độ trễ phản hồi của Website HPU.
- **Tầng thu thập và lưu trữ (Collector & Storage):**
  - **Prometheus (Cổng 9091):** Đóng vai trò trung tâm, thực hiện Pull dữ liệu từ **Node Exporter** (để lấy thông số phần cứng) và **Blackbox Exporter** (để lấy trạng thái website) và lưu trữ vào cơ sở dữ liệu chuỗi thời gian (Time Series DataBase).

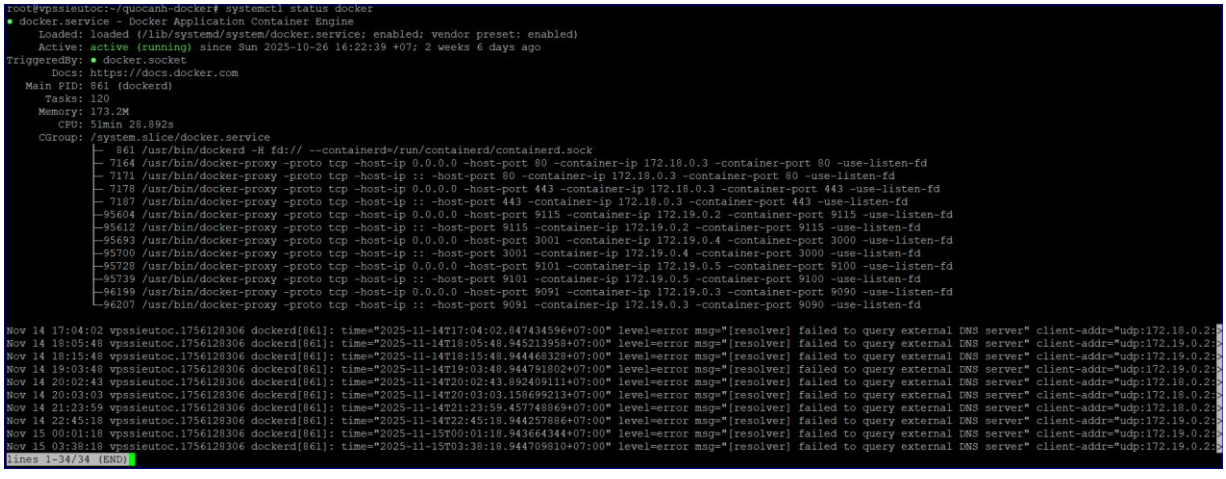
- **Tầng hiển thị và cảnh báo (Presentation & Alerting):**
  - **Grafana (Cổng 3001):** Giao diện quản trị, hiển thị Dashboard.
- **Kênh thông báo (Notification Channel):**
  - **Discord Webhook:** Điểm tiếp nhận cảnh báo.

3.4 Triển khai môi trường và hạ tầng

- Tập trung vào việc chuẩn bị hạ tầng cơ sở và thiết lập các container Docker. Đây là bước tiền đề để khởi chạy đồng thời toàn bộ dịch vụ giám sát cốt lõi (Prometheus, Grafana, Exporters), chuyển đổi bản thiết kế kiến trúc sang môi trường hoạt động thực tế.

a, Chuẩn bị môi trường: Cài đặt Docker và Docker Compose trên VPS Linux

- Hệ thống được triển khai trên máy chủ VPS chạy Ubuntu Server. Docker và Docker Compose đã được cài đặt và cấu hình từ trước để phục vụ việc chạy các dịch vụ giám sát như Prometheus, Grafana, Node Exporter và Blackbox Exporter. Để đảm bảo môi trường hoạt động ổn định và không ảnh hưởng đến dữ liệu đang chạy, nhóm chỉ thực hiện các thao tác **kiểm tra** thay vì cài đặt lại.
- Đầu tiên, trạng thái của Docker Engine được kiểm tra bằng lệnh: **systemctl status docker**



Hình 10: Trạng thái Docker Engine (đã được active)

- Kết quả cho thấy dịch vụ Docker đang ở trạng thái **active (running)**, đảm bảo đủ điều kiện khởi chạy các container giám sát. Sau đó, phiên bản Docker và Docker Compose đang sử dụng được xác thực bằng:

  docker --version

  docker compose version

```
root@vpssieutoc:~/quocanh-docker# docker --version
docker compose version
Docker version 28.5.1, build e180ab8
Docker Compose version v2.40.2
root@vpssieutoc:~/quocanh-docker#
```

Hình 11: Phiên bản của docker và docker compose

b, Thiết lập kiến trúc dịch vụ và ánh xạ ports

- Lệnh kiểm tra container đang hoạt động được thực hiện như sau:

  docker ps

```
root@vpssieutoc:~/quocanh-docker# docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED       STATUS       PORTS
 NAMES
074304bd982f   prom/prometheus        "/bin/prometheus --c…"   2 weeks ago   Up 2 weeks   0.0.0.0:9091->9090/tcp, [::]:9091->9090/tcp
 prometheus-quocanh
b15ff86143ad   prom/blackbox-exporter "/bin/blackbox_expor…"   2 weeks ago   Up 2 weeks   0.0.0.0:9115->9115/tcp, [::]:9115->9115/tcp
 blackbox-quocanh
3ee48ba04f9b   grafana/grafana        "/run.sh"                2 weeks ago   Up 2 weeks   0.0.0.0:3001->3000/tcp, [::]:3001->3000/tcp
 grafana-quocanh
9c177b59d2e1   prom/node-exporter     "/bin/node_exporter …"   2 weeks ago   Up 2 weeks   0.0.0.0:9101->9100/tcp, [::]:9101->9100/tcp
 node-exporter-quocanh
b2a9e4345c86   n8nio/n8n              "tini -- /docker-ent…"   2 weeks ago   Up 5 days    5678/tcp
 n8n-server_n8n_1
06d518798c67   caddy:latest           "caddy run --config …"   2 weeks ago   Up 2 weeks   0.0.0.0:80->80/tcp, [::]:80->80/tcp, 0.0.0.0:443->443/tcp, [::]:443->443/tcp, 443/udp, 2019/tcp
 n8n-server_caddy_1
5138306aee22   n8nio/n8n:latest       "tini -- /docker-ent…"   2 weeks ago   Up 2 weeks   5678/tcp
 n8n-an
root@vpssieutoc:~/quocanh-docker#
```

Hình 12: Hình ảnh các container đang hoạt động

- Kết quả hiển thị đầy đủ 4 dịch vụ chính gồm Prometheus, Grafana, Node Exporter và Blackbox Exporter, tất cả đều ở trạng thái **Up** liên tục từ thời điểm triển khai ban đầu.

Kết luận: Kết quả kiểm tra cho thấy Docker và Docker Compose đã được cài đặt và hoạt động ổn định trên VPS. Các dịch vụ hỗ trợ chạy container đều ở trạng thái **active (running)**, đảm bảo môi trường sẵn sàng cho quá trình triển khai hệ thống giám sát. Việc xác thực này giúp khẳng định rằng nền tảng hạ tầng đã được chuẩn

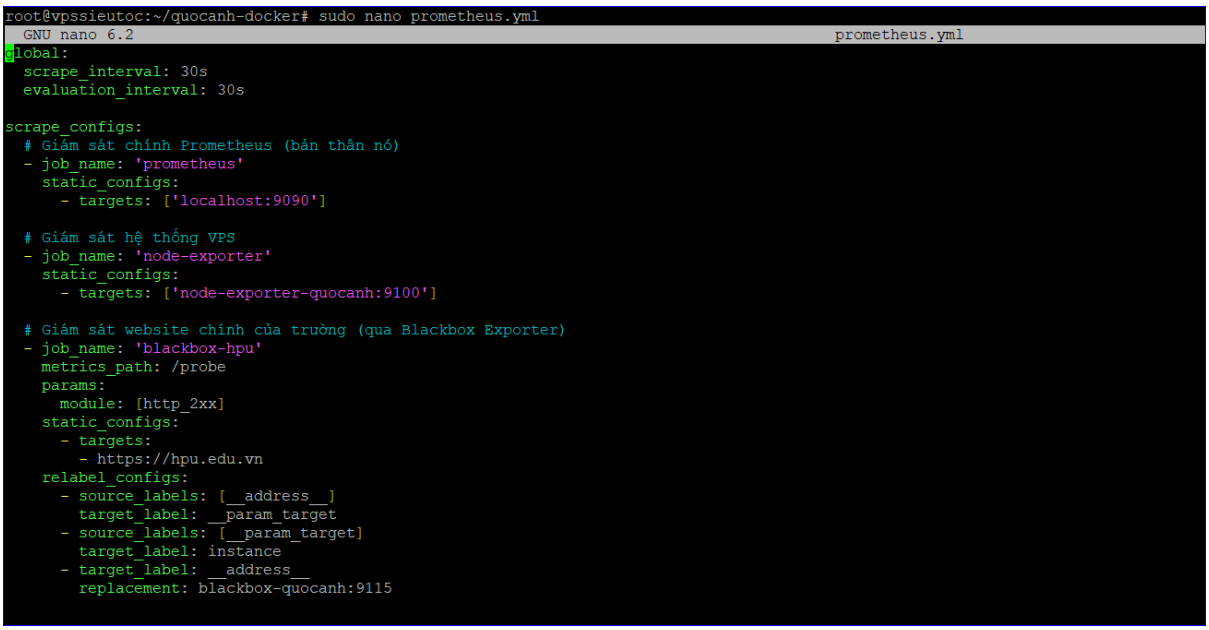
bị đầy đủ, phục vụ cho các bước cấu hình và triển khai các dịch vụ Prometheus, Grafana, Node Exporter và Blackbox Exporter ở các mục tiếp theo.

3.5 Cấu hình thu thập dữ liệu

- Mục này tập trung vào việc thiết lập các file cấu hình YAML cốt lõi để chỉ thị cho Prometheus biết tần suất thu thập dữ liệu (Scrape Interval) và định danh chính xác các mục tiêu giám sát (Exporters và Website)

a, Cấu hình Prometheus Server

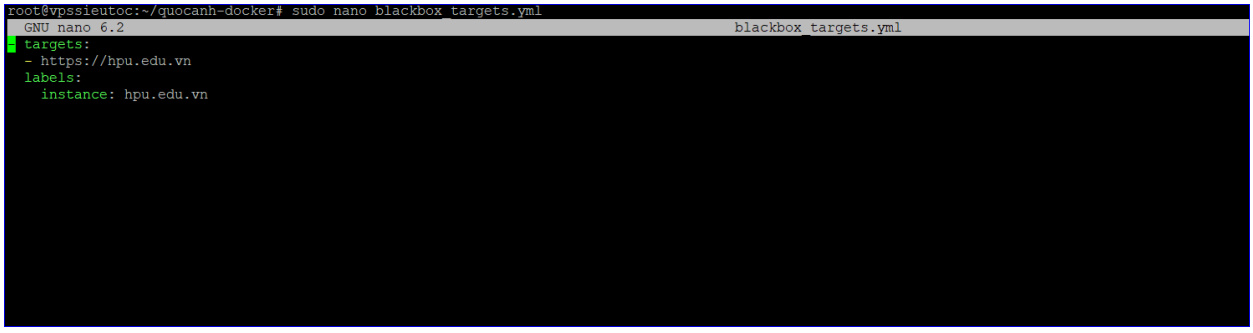
- File cấu hình chính định nghĩa các công việc giám sát (Jobs) và tích hợp các Exporter cần thiết. Cấu hình này chỉ thị cho Prometheus tần suất thu thập dữ liệu và định tuyến các truy vấn kiểm tra Website. Gõ lệnh **sudo nano prometheus.yml**
- **Cài đặt tần suất Scrape:** Thiết lập **scrape_interval: 30s** cho các công việc chung
- **Job node-exporter:** Cấu hình để Prometheus kết nối tới Node Exporter (tại cổng nội bộ 9100) để thu thập các chỉ số phần cứng (CPU, RAM và băng thông mạng).
- **Job blackbox-hpu:** Thiết lập mục tiêu Website một cách **tĩnh (static)** ngay trong file prometheus.yml
- **Định tuyến dữ liệu:** Sử dụng relabel_configs để đảm bảo các truy vấn kiểm tra Website được định tuyến chính xác đến Blackbox Exporter (cổng 9115).



Hình 13: Cấu hình Prometheus Server

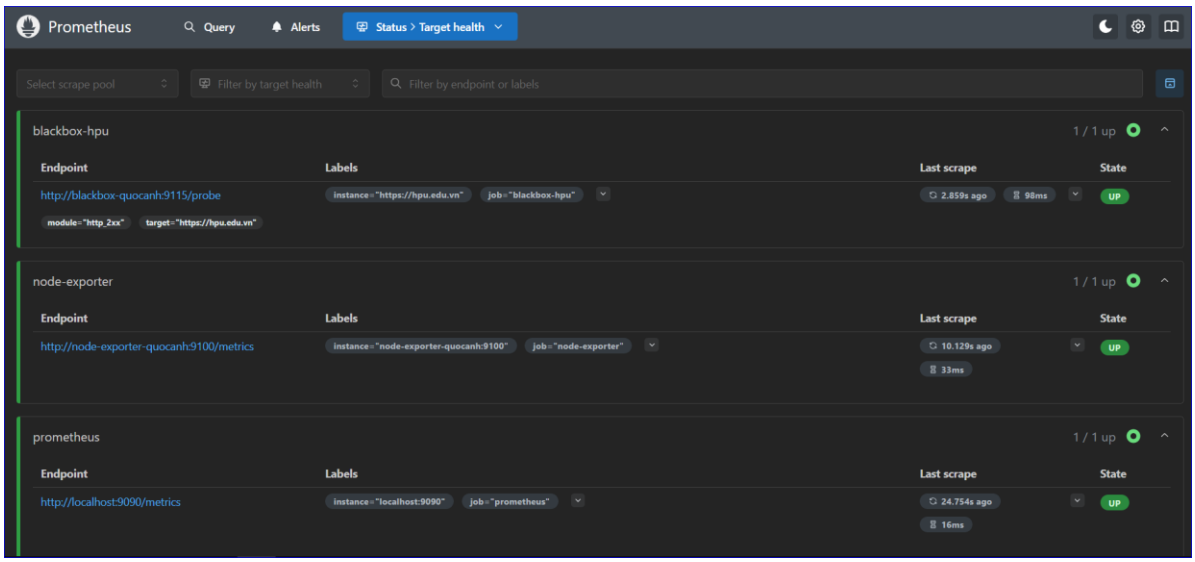
b, Cấu hình mục tiêu giám sát

- File này cung cấp danh sách URL cụ thể mà Blackbox Exporter cần thăm dò. Đây là bước quan trọng để đảm bảo hệ thống kiểm tra đúng Website của trường, đồng thời giúp Prometheus biết cách phân loại dữ liệu thu thập được bằng các nhãn (labels) đính kèm. Cấu hình dưới đây xác định Website HPU là mục tiêu duy nhất. Gõ lệnh **sudo nano blackbox_targets.yml**
- **Khai báo Target:** Khai báo địa chỉ chính xác của Website HPU (https://hpu.edu.vn) làm mục tiêu kiểm tra.
- **Đính kèm Labels:** Đính kèm nhãn **instance: hpu.edu.vn** để Prometheus dễ dàng phân loại và truy vấn dữ liệu này.



Hình 14: Cấu hình mục tiêu giám sát

- Việc kiểm tra trạng thái mục tiêu (Target Health) trên giao diện Prometheus xác nhận toàn bộ quá trình cấu hình đã thành công. Tất cả các Job (bao gồm Node Exporter, Blackbox Exporter và Prometheus tự giám sát) đều hiển thị trạng thái **UP** (hoạt động), chứng minh hệ thống đang sẵn sàng cho bước trực quan hóa dữ liệu trên Grafana.



Hình 15: Giao diện Prometheus Target Health

Kết luận: Quá trình cấu hình Prometheus đã hoàn tất. Với các thiết lập về Job và Target, hệ thống đã được chỉ thị cụ thể để thu thập dữ liệu từ các Exporter theo đúng tần suất định kỳ, đảm bảo toàn bộ dữ liệu hiệu năng và trạng thái sẵn sàng cho bước tiếp theo là trực quan hóa trên Grafana

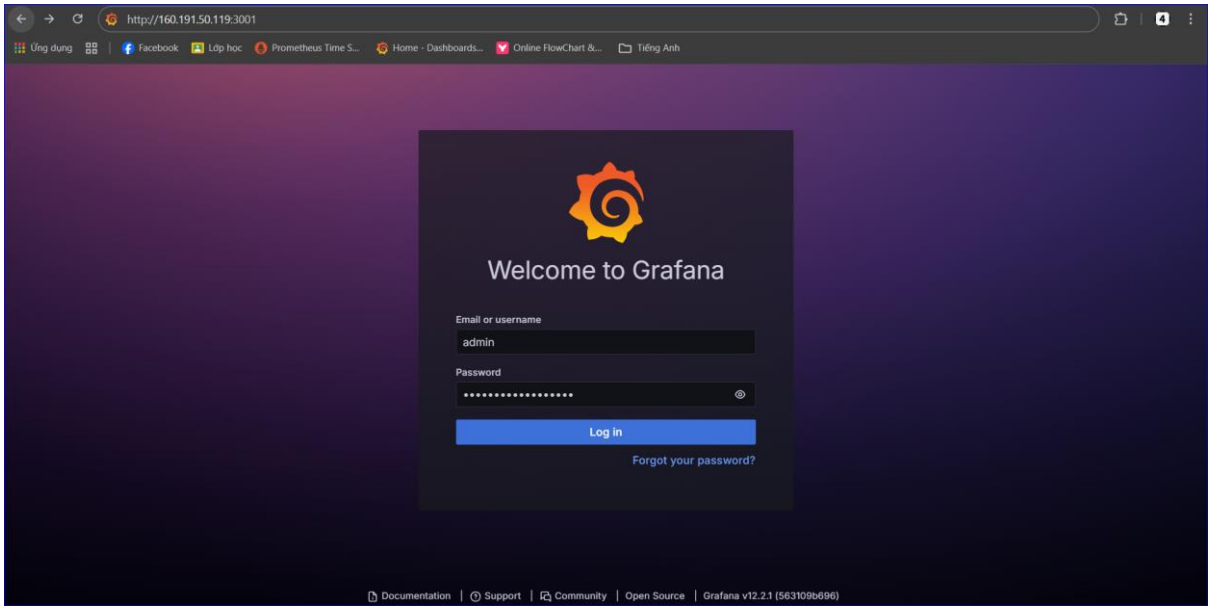
3.6 Cấu hình trực quan hóa trên Grafana

- Sau khi dữ liệu đã được thu thập và lưu trữ thành công trong Prometheus, mục này sẽ tập trung vào việc sử dụng Grafana (truy cập qua cổng 3001) để truy vấn dữ liệu, xây dựng Dashboard trực quan nhằm chuyển đổi các chỉ số thô thành thông tin quản trị có giá trị

3.6.1. Kết nối Data Source: Thiết lập kết nối tới Prometheus

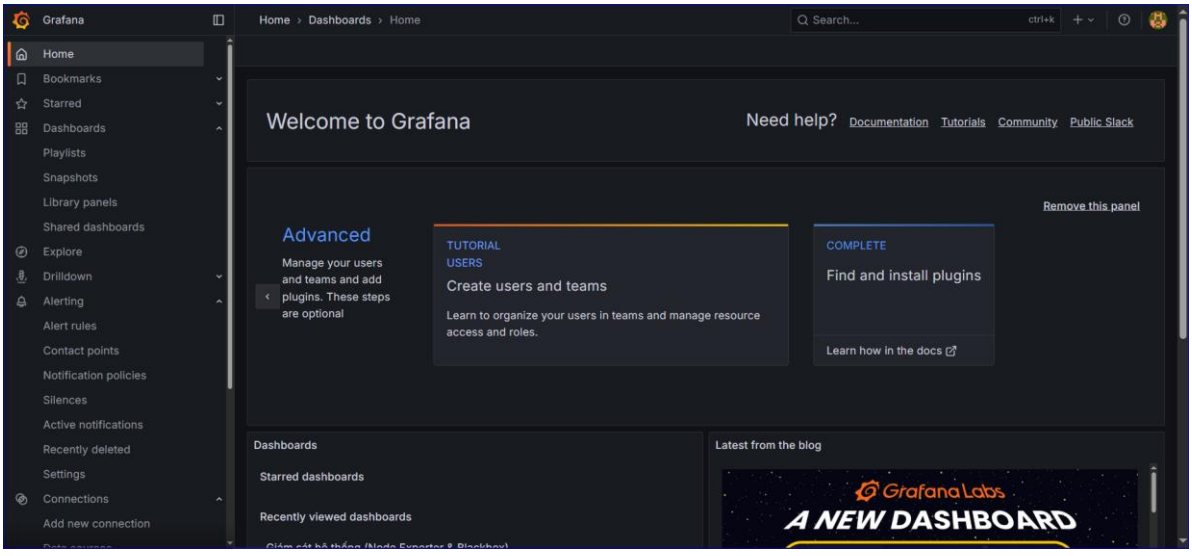
Việc này đảm bảo Grafana có thể giao tiếp với Prometheus để lấy dữ liệu.

Bước 1: Truy cập Grafana: Mở trình duyệt web và truy cập giao diện quản trị Grafana bằng địa chỉ IP của VPS và cổng đã ánh xạ: **http://160.191.50.119:3001**



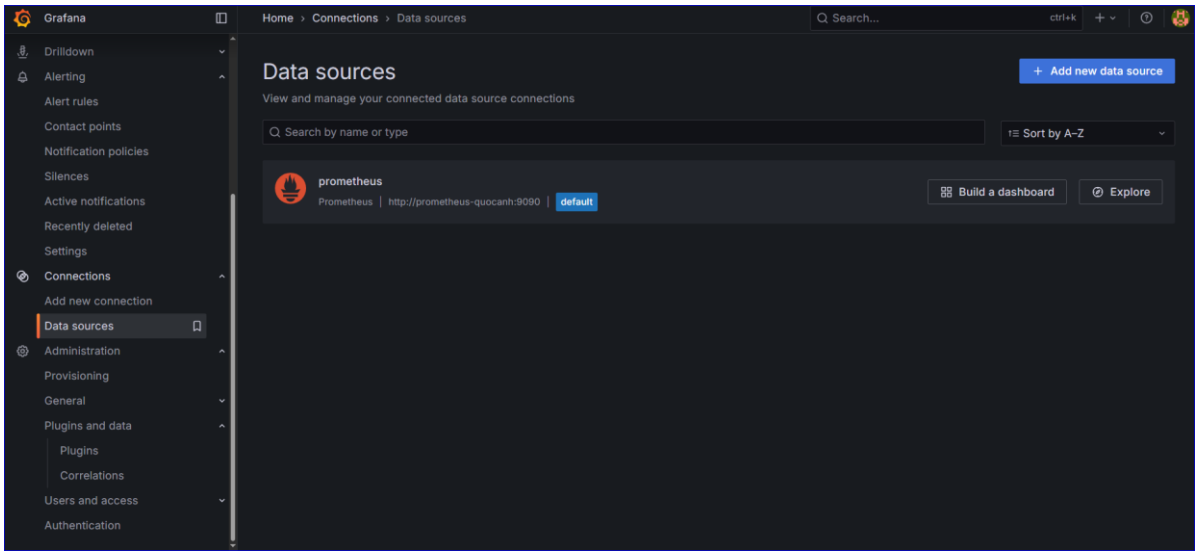
Hình 16: Giao diện đăng nhập Grafana

Bước 2: Đăng nhập: Tại màn hình đăng nhập, nhập Username và Password quản trị đã thiết lập để truy cập hệ thống. Vào màn hình giao diện chính Grafana.



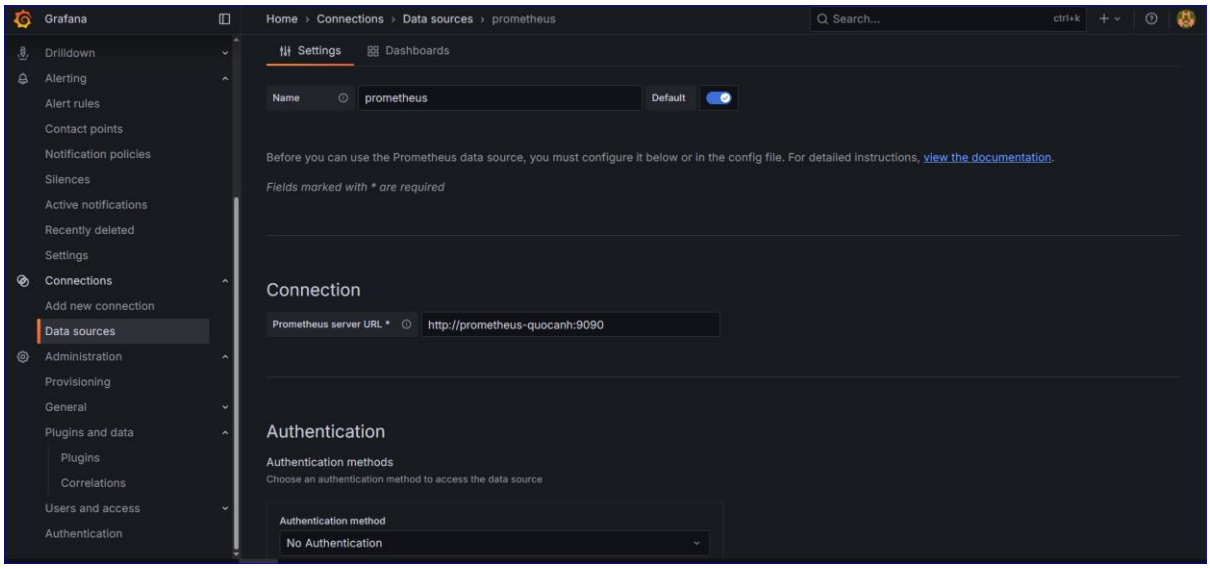
Hình 17: Giao diện chính Grafana

Bước 3: Thêm Data Source: Trong giao diện quản trị, chọn **Connections → Add new connection →** Chọn loại nguồn dữ liệu là **Prometheus**.



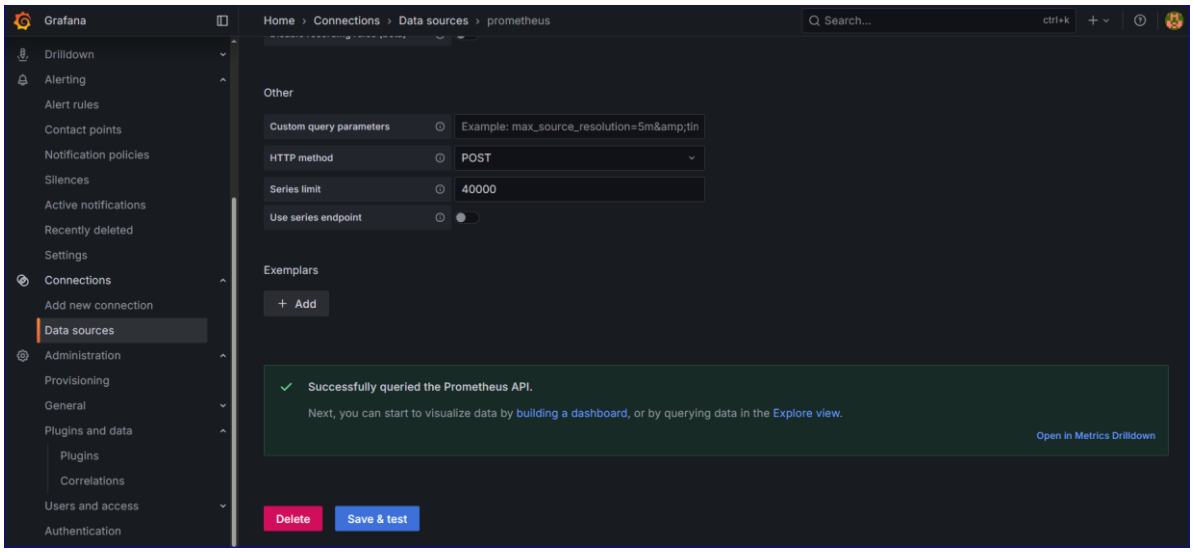
Hình 18: Thêm Data Source

Bước 4: Cấu hình URL: Trong mục **HTTP**, tại trường **URL** (địa chỉ nội bộ Docker), nhập địa chỉ: **http://prometheus-quocanh:9090** (sử dụng tên service và cổng nội bộ) để thiết lập kết nối giữa Grafana và Prometheus.



Hình 19: Cấu hình URL

Bước 5: Lưu: Nhấn **Save & Test**.



Hình 20: Nguồn dữ liệu đang hoạt động

3.6.2. Xây dựng dashboard giám sát hiệu năng server và giám sát dịch vụ website (dashboard tổng)

- Dashboard này được thiết kế để cung cấp cái nhìn toàn diện hiển thị cả hiệu năng server và trạng thái dịch vụ website trên cùng một màn hình. Quy trình thực hiện:

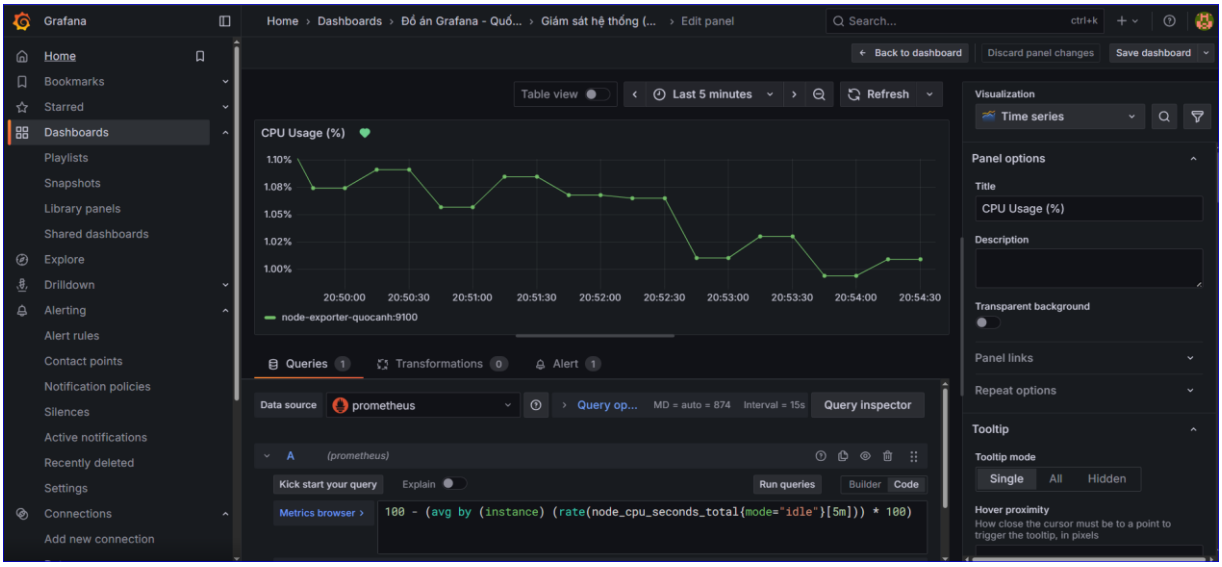
- **Tạo Dashboard Mới:** Trong giao diện Grafana, chọn **Dashboards → New Dashboard**. Đặt tên là: **Giám sát hệ thống (Node Exporter & Blackbox)**
- **Thêm Panel:** Chọn **Add visualization**. Chọn Data Source là **Prometheus**.

- **Xây dựng Panel:** Thực hiện các bước sau để xây dựng từng Panel giám sát:

a, Panel: CPU Usage (%):

- **Mục đích:** Đo lường phần trăm CPU đang hoạt động.
- **Công thức (PromQL):** 100 - (avg by (instance) (rate(node_cpu_seconds_total{mode="idle"}[5m])) * 100)

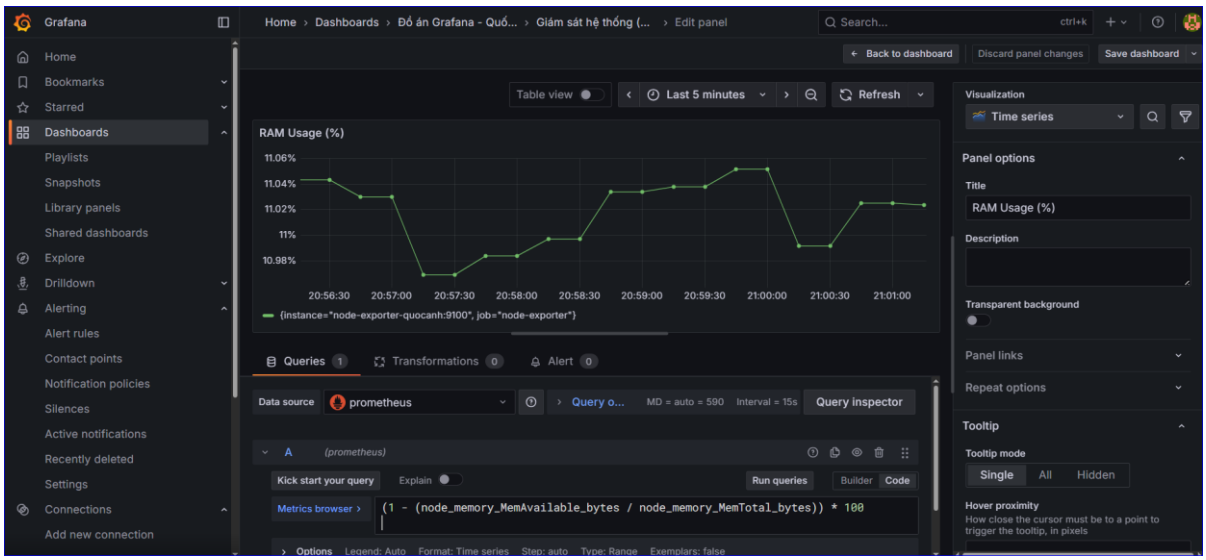
- **Thao tác:** Điền công thức vào trường **Query**. Đặt tiêu đề Panel là **CPU Usage**. Chọn loại **Visualization** là **Time Series**. Cấu hình đơn vị hiển thị là **percent (0-100)**.



Hình 21: Panel CPU Usage (%)

b, Panel: RAM Usage (%)

- **Mục đích:** Đo lường phần trăm RAM đã được sử dụng so với tổng dung lượng RAM.
- **Công thức (PromQL):** (1 - (node_memory_MemAvailable_bytes / node_memory_MemTotal_bytes)) * 100
- **Thao tác:** Điền công thức vào trường **Query**. Đặt tiêu đề Panel là **RAM Usage**. Chọn loại **Visualization** là **Time Series**. Cấu hình đơn vị hiển thị là **percent (0-100)**.

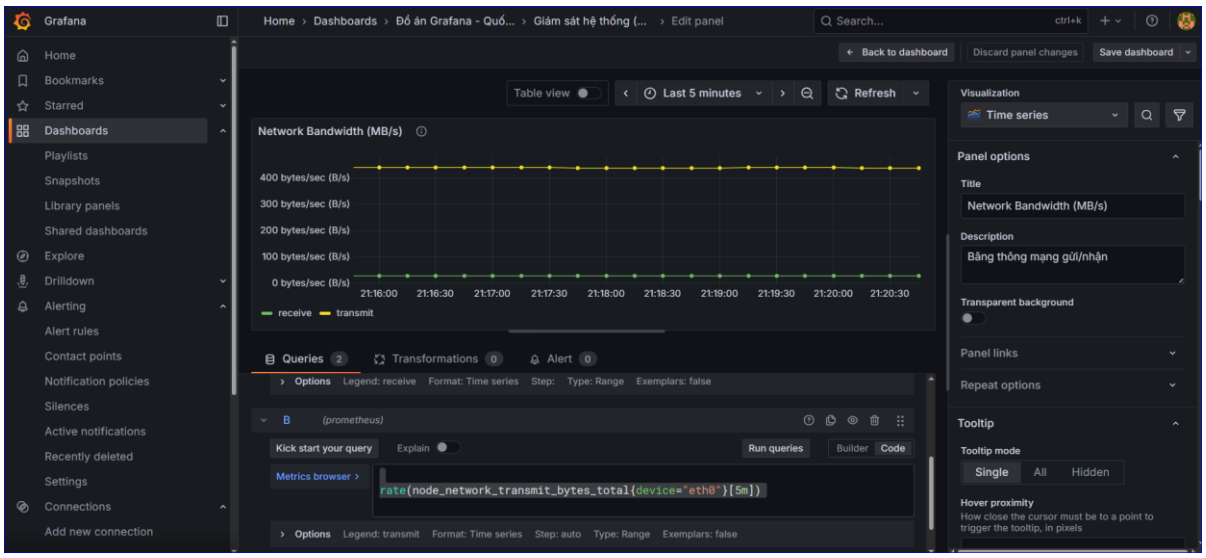


Hình 22: Panel RAM Usage (%)

c, Panel: Network Traffic (Băng thông mạng)

- **Mục đích:** Đo lường tốc độ truyền và nhận dữ liệu qua giao diện mạng của VPS.
- **Công thức (PromQL):**
  - Tốc độ Nhận (RX - Receive):
    rate(node_network_receive_bytes_total{device="eth0"}[5m])
  - Tốc độ Truyền (TX - Transmit):
    rate(node_network_transmit_bytes_total{device="eth0"}[5m])

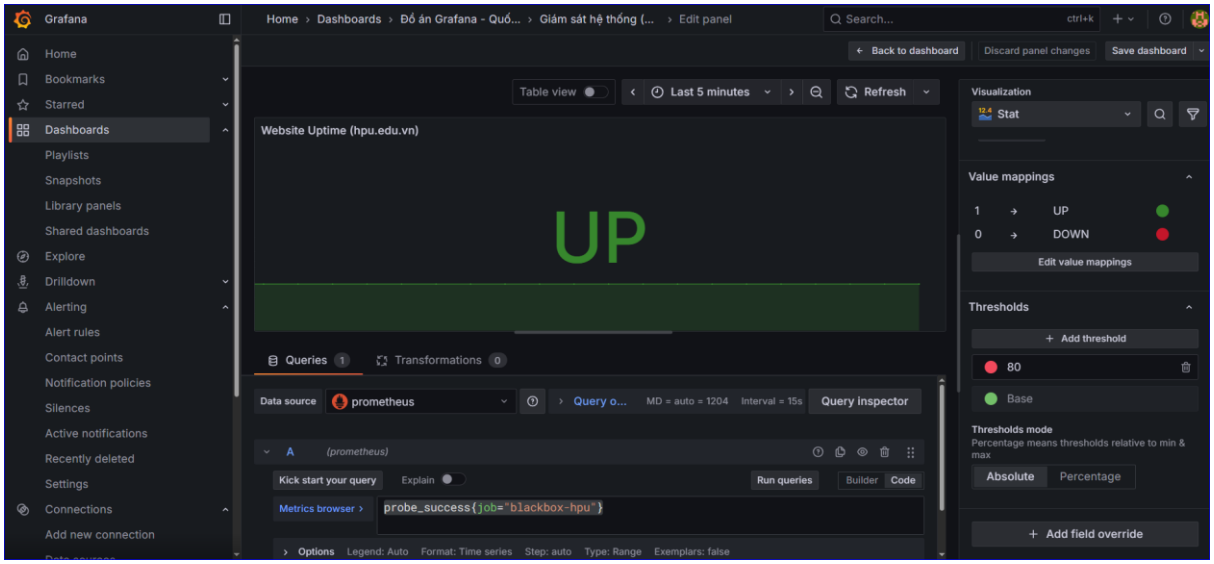
- **Thao tác:** Điền từng công thức vào Panel, sử dụng tính năng **Legend** để đặt tên hiển thị. **Chọn loại Visualization là Time Series**. Cấu hình đơn vị hiển thị là **bytes/sec (B/s)**.



Hình 23: Panel Network Traffic

d, Panel: Website Uptime (Trạng thái hoạt động của website)

- **Mục đích:** Xác định trạng thái hoạt động của website, sống/chết (UP = 1, DOWN = 0).
- **Công thức (PromQL):** probe_success{job="blackbox-hpu"}
- **Thao tác:** Điền công thức vào trường **Query**. Chọn loại **Visualization** là **Stat**. Cấu hình **Thresholds** (Ngưỡng) để hiển thị màu sắc: 1 (UP) sẽ có nền màu xanh lá, 0 (DOWN) sẽ có nền màu đỏ.



Hình 24: Panel Website Uptime

- Sau khi đã xây dựng thành công 4 Panels cốt lõi (CPU, RAM, Network Traffic, Website Uptime), em tiến hành lưu Dashboard này lại. Dashboard được đặt tên **'Giám sát hệ thống (Node Exporter & Blackbox)'** và được lưu trữ trong thư mục **'Đồ án Grafana - Quốc Anh'** để đảm bảo tính quản lý tập trung.

Hình 25: Dashboard tổng

3.7Thiết lập hệ thống cảnh báo

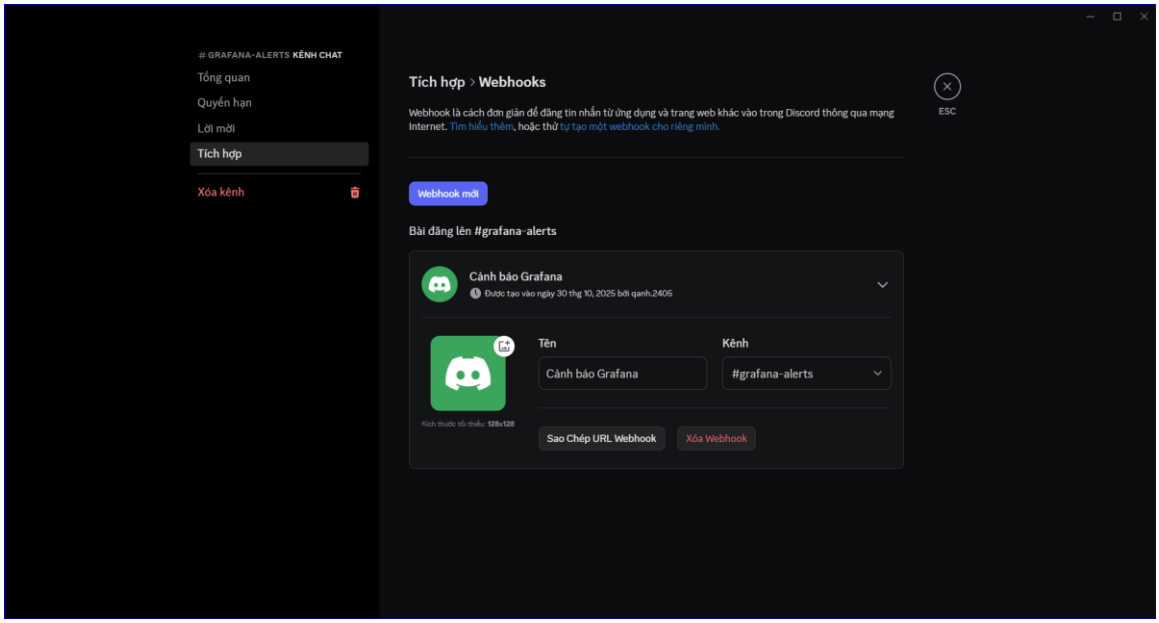
- Sau khi các Dashboard đã được xây dựng thành công, mục này sẽ thiết lập cơ chế phản ứng tự động. Việc thiết lập cảnh báo giúp hệ thống chuyển từ chế độ giám sát thụ động sang chế độ **chủ động (Active Monitoring)**, đảm bảo thông báo sự cố được gửi ngay lập tức tới đội ngũ kỹ thuật thông qua kênh Discord.

3.7.1 Cấu hình kênh thông báo (Contact Point)

- Bước này thiết lập cầu nối giữa Grafana và kênh nhận cảnh báo Discord.

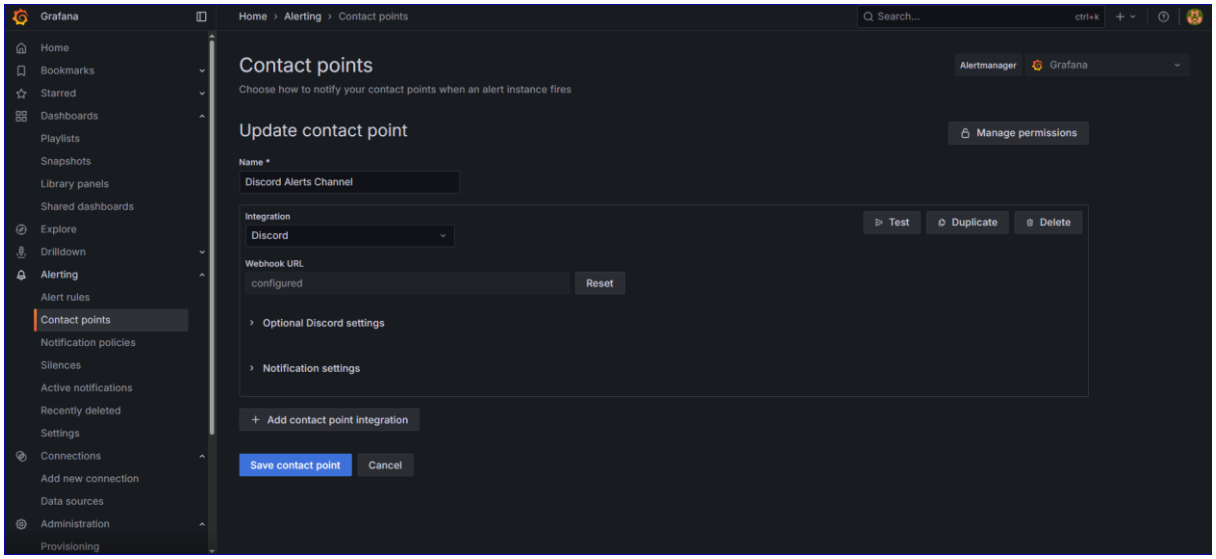
- **Truy cập Cấu hình:** Trong menu bên trái Grafana, chọn **Alerting → Contact points**.
- **Tạo Contact Point mới:** Nhấn nút **+ New contact point**.
- **Thiết lập chi tiết:**
  - **Name:** Đặt tên là Discord Alerts Channel.
  - **Integration Type:** Chọn **Discord**

- **URL:** Điền Webhook URL của kênh Discord đã tạo: https://discord.com/api/webhooks/1433240477490217010/Z9XorQu6fLT1ClklRIHHsOL_ivHNtLybJJz86-sLdOFTEe0rd4O6_uHtzUw87hkaBZXC



Hình 26: Nơi chưa Webhook URL của kênh Discord

- **Kiểm tra và Lưu:** Nhấn nút **Test** ở cuối trang để xác minh Discord nhận được thông báo kiểm tra, sau đó nhấn **Save contact point**.

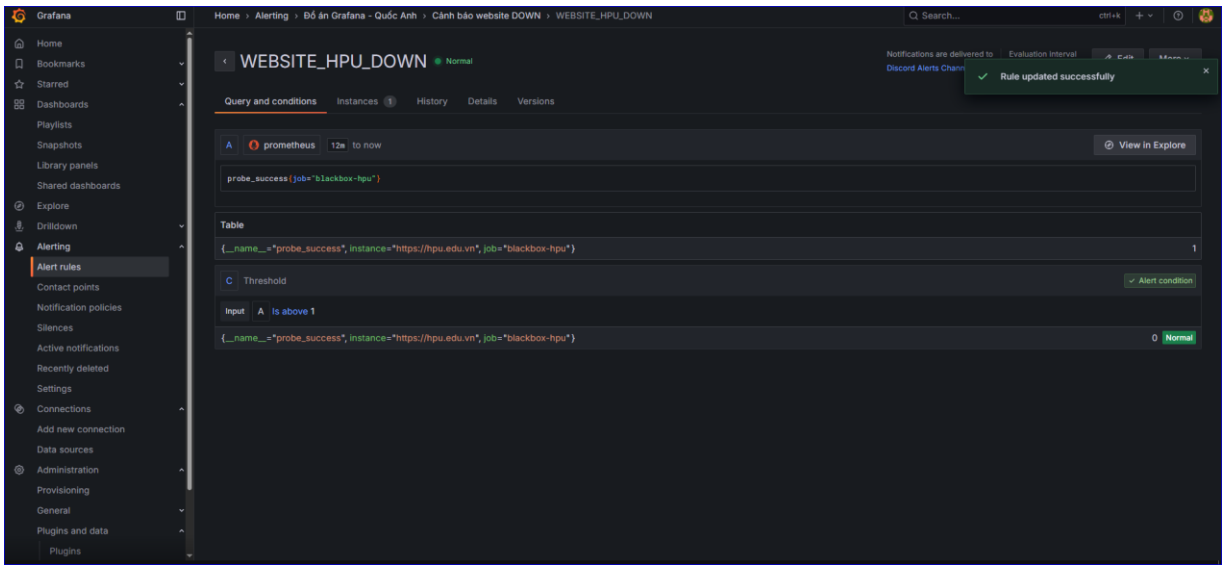


Hình 27: Xác nhận kết nối tới Discord Webhook đã được thiết lập thành công

3.7.2 Thiết lập Quy tắc cảnh báo (Alert Rules)

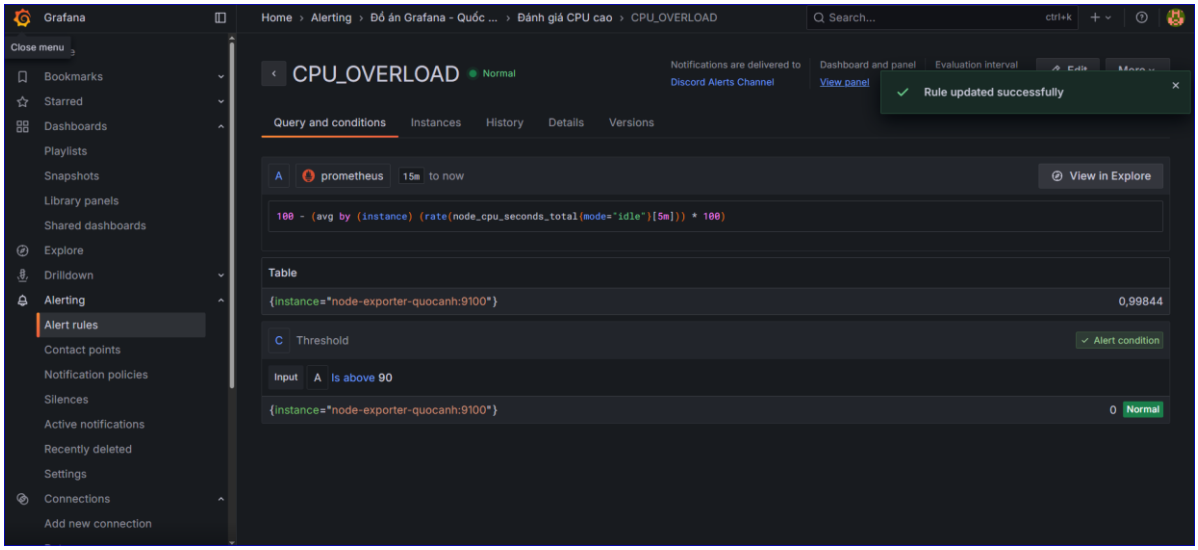
- Quy trình thiết lập quy tắc cảnh báo được thực hiện trực tiếp trên giao diện Grafana, chuyển đổi các công thức PromQL thành các ngưỡng kích hoạt cảnh báo tự động.

- **Quy tắc 1: Cảnh báo Website DOWN**: Quy tắc này nhằm đảm bảo tính **sẵn sàng** của Website chính thức của trường.
  - **Bước 1:** Truy cập **Alerting →Alert rules** và nhấn + **New alert rule.**
  - **Bước 2:** Đặt tên quy tắc là **WEBSITE_HPU_DOWN**.
  - **Bước 3:** Tại mục **Define query and alert condition**, điền công thức PromQL kiểm tra trạng thái Website: **probe_success{job="blackbox-hpu"}**
  - **Bước 4:** Thiết lập điều kiện kích hoạt: **WHEN QUERY A IS BELOW 1.**
  - **Bước 5:** Chọn thư mục đồ án: Đồ án Grafana - Quốc Anh
  - **Bước 6:** Đặt thời gian duy trì **for 1m** (1 phút) để cảnh báo được kích hoạt nhanh chóng.
  - **Bước 7:** Trong mục **Configure notifications**, chọn **Discord Alerts Channel** làm kênh nhận thông báo.
  - **Bước 8:** Nhấn **Save.**



Hình 28: Thiết lập cảnh báo Website Down thành công

- **Quy tắc 2: Cảnh báo CPU quá tải**
  - **Bước 1:** Lặp lại quy trình tạo Rule mới.
  - **Bước 2:** Đặt tên Quy tắc là **CPU_OVERLOAD**.
  - **Bước 3:** Điền công thức PromQL tính phần trăm CPU đang hoạt động**: 100 - (avg by (instance) (rate(node_cpu_seconds_total{mode="idle"}[5m])) * 100)**
  - **Bước 4:** Thiết lập điều kiện kích hoạt: **WHEN QUERY A IS ABOVE 90**.
  - **Bước 5:** Chọn thư mục đồ án: Đồ án Grafana - Quốc Anh.
  - **Bước 6:** Đặt thời gian duy trì **for 5m** (5 phút) để tránh các cảnh báo giả mạo.
  - **Bước 7:** Liên kết thông báo tới **Discord Alerts Channel**.
  - **Bước 8:** Nhấn **Save.**



Hình 29: Thiết lập cảnh báo CPU quá tải thành công

3.8 Kịch bản kiểm thử và kết quả

- Đóng vai trò là bước xác thực cuối cùng, chuyển đổi hệ thống từ trạng thái đã được cấu hình sang trạng thái **đã được kiểm chứng** (End-to-End validation).
- Phần này sẽ trình bày chi tiết các kịch bản giả lập sự cố (Website Down và CPU Overload) để kích hoạt Alert Rules, qua đó chứng minh toàn bộ quy trình từ thu thập dữ liệu đến gửi cảnh báo Discord hoạt động chính xác và hiệu quả.

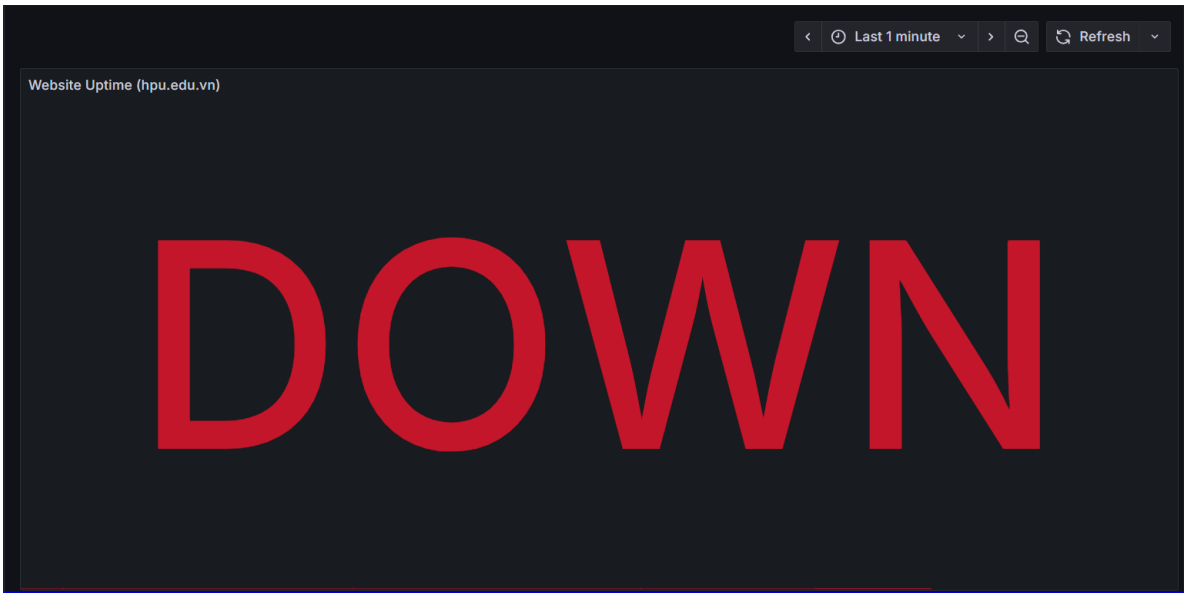
3.8.1. Kịch bản giả lập sự cố mất kết nối với website HPU và kết quả

- **Kịch bản** được thực hiện bằng cách giả lập tình trạng mất kết nối với Website HPU để kích hoạt Rule **WEBSITE_HPU_DOWN** trong môi trường Docker. Để kích hoạt cảnh báo, em thực hiện thay đổi URL mục tiêu trong file **prometheus.yml** thành một địa chỉ không tồn tại (sẽ trả về lỗi 404 hoặc timeout).
    - **Thao tác làm lỗi:** Mở file cấu hình mục tiêu trên VPS (trong thư mục quocanh-docker): Chạy lệnh **sudo nano prometheus.yml**
    - **Sửa URL:** Sửa địa chỉ Website HPU thành một địa chỉ giả:
        - **Trước khi sửa:** - https://hpu.edu.vn
        - **Sau khi sửa (Làm lỗi):** - https://hpu.edu.vn/url_fake
    - **Khởi động lại Prometheus:** Chạy lệnh **docker restart prometheus-quocanh** để Prometheus tải cấu hình lỗi mới.
- **Kết quả cảnh báo thực tế:** Sau khi khởi động lại Prometheus, Rule WEBSITE_HPU_DOWN sẽ bắt đầu đánh giá URL lỗi.

a, Trạng thái ALERTING (Mất kết nối):

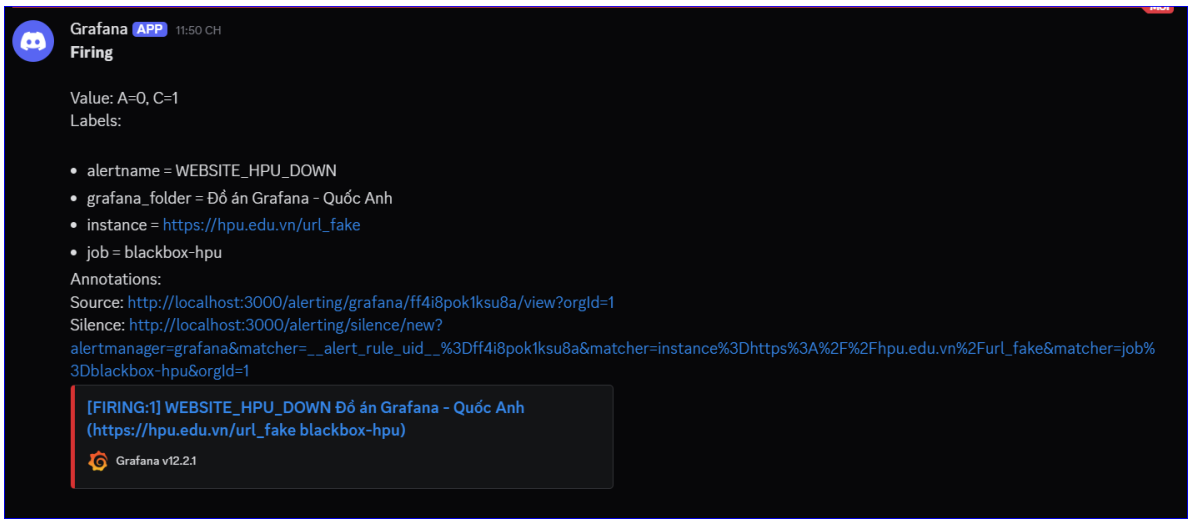
- Sau khoảng **1 phút** (theo thời gian for 1m đã thiết lập), Grafana xác nhận sự cố và chuyển Rule sang trạng thái **Firing** (Đang báo động).

- **Bằng chứng 1:** Dashboard chuyển trạng thái sang **màu đỏ** (Alerting).



Hình 30: Dashboard chuyển trạng thái sang DOWN màu đỏ

- **Bằng chứng 2:** Kênh Discord nhận được tin nhắn **[FIRING]** (Cảnh báo Kích hoạt).



Hình 31: Kênh Discord nhận tin nhắn FIRING (WEBSITE_HPU_DOWN)

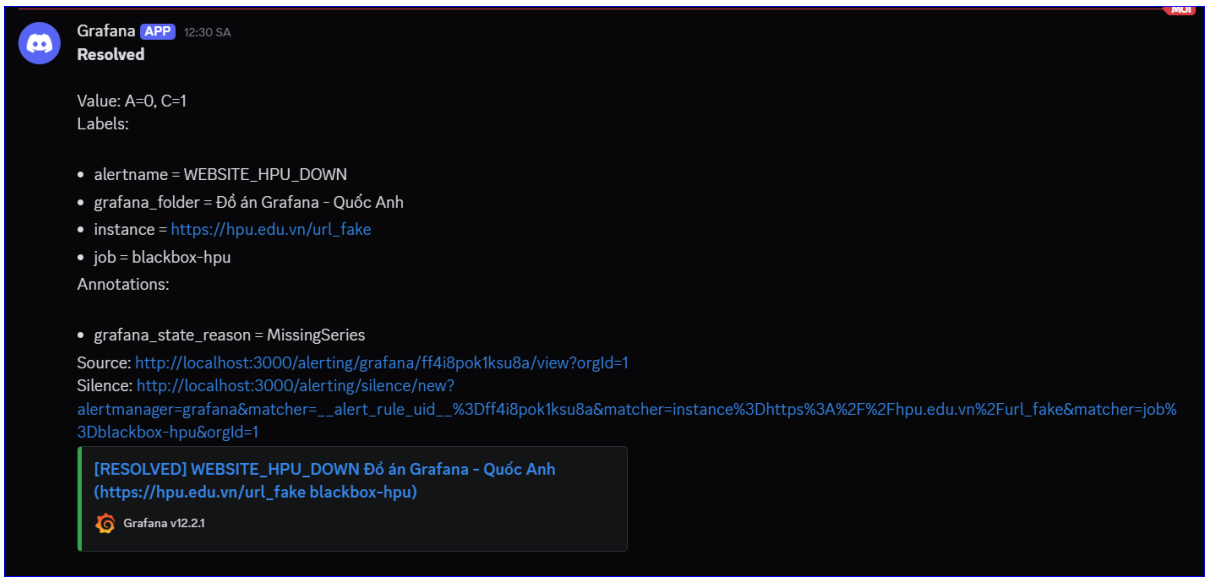
b, Trạng thái RESOLVED (Hệ thống khôi phục bình thường):

- Sau khi đã ghi nhận cảnh báo ở trạng thái Firing, em tiến hành hoàn tác cấu hình để mô phỏng việc hệ thống trở lại hoạt động bình thường.
- Hoàn tác URL lỗi: Mở lại file cấu hình mục tiêu và chỉnh URL về đúng địa chỉ ban đầu: https://hpu.edu.vn
- Quá trình đánh giá lại trạng thái (Recovery): Ngay sau khi Prometheus tải lại cấu hình hợp lệ, các phép đo probe_success bắt đầu trả về giá trị **1**, biểu thị việc website đã truy cập được.

- **Kết quả khôi phục:**
  - Sau khoảng 30–60 giây, Grafana tự động chuyển Rule từ trạng thái **Firing → Normal (OK)**.
  - Panel trên dashboard chuyển từ màu đỏ trở lại màu xanh.
  - Discord ghi nhận một tin nhắn mới ở trạng thái **[RESOLVED]**, xác nhận sự cố đã được giải quyết.
  - **Bằng chứng 3:** Trạng thái trên Dashboard chuyển từ Alerting (đỏ) về Normal (xanh).

Hình 32: Trạng thái trên Dashboard chuyển về UP màu xanh

- **Bằng chứng 4:** Tin nhắn **[RESOLVED]** được gửi tới kênh Discord (tự động).



Hình 33: Kênh Discord nhận tin nhắn RESOLVED (WEBSITE_HPU_DOWN)

3.8.2. Kịch bản giả lập sự cố CPU quá tải và kết quả

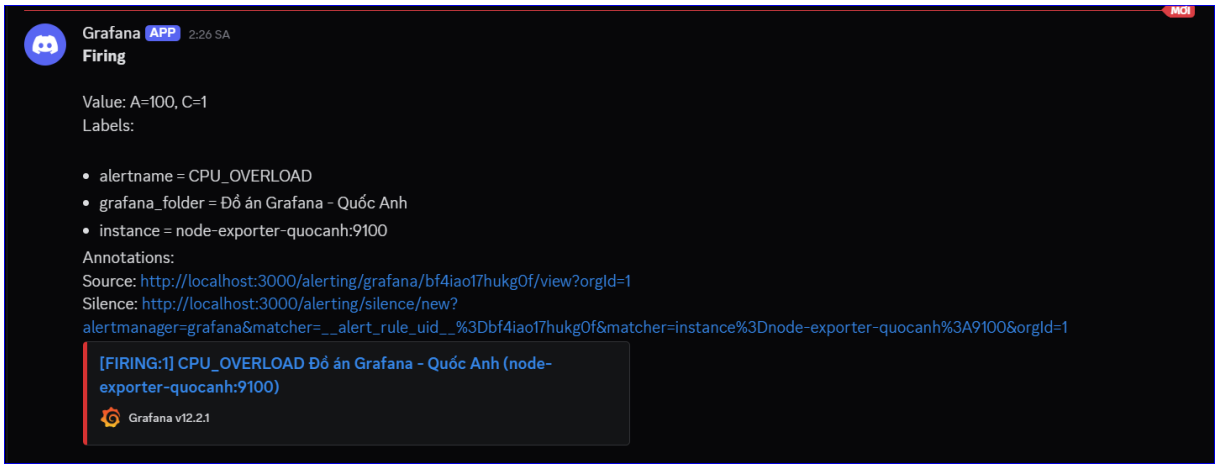
- **Kịch bản** được thực hiện bằng cách sử dụng các lệnh hệ thống để tạo tải giả lập trên VPS, đẩy mức sử dụng CPU vượt quá ngưỡng 90% để kích hoạt Rule **CPU_OVERLOAD**. Do môi trường VPS dùng chung, em sử dụng phương pháp tạo tải chủ động (dùng lệnh yes) để dễ dàng kiểm soát và dừng lại ngay lập tức.
    - **Thao tác làm lỗi (Tạo tải):** Chạy lệnh yes liên tục để ép CPU hoạt động tối đa công suất.
    - Chạy nhiều luồng song song trong Putty để đảm bảo chiếm dụng toàn bộ các nhân CPU: **for i in {1..n}; do yes > /dev/null & done**
- **Kết quả cảnh báo thực tế:** Ngay sau khi lệnh được thực thi, Node Exporter ghi nhận mức tiêu thụ CPU tăng đột biến vượt mức 90%. Grafana bắt đầu đánh giá Rule.

a, Trạng thái QUÁ TẢI & ALERTING (Phát hiện sự cố):

- Sau khoảng 1 phút (theo thời gian for 1m đã thiết lập cho Demo), Grafana xác nhận tình trạng quá tải kéo dài và chuyển Rule sang trạng thái **Firing** (Đang báo động).
- **Bằng chứng 1:** Trên Dashboard, biểu đồ **CPU Usage (%)** vượt mức hơn 90%.

Hình 34: Biểu đồ CPU Usage vượt mức 90%

- **Bằng chứng 2:** Kênh Discord nhận được tin nhắn **[FIRING]** với nội dung cảnh báo máy chủ đang quá tải.



Hình 35: Kênh Discord nhận tin nhắn FIRING (CPU_OVERLOAD)

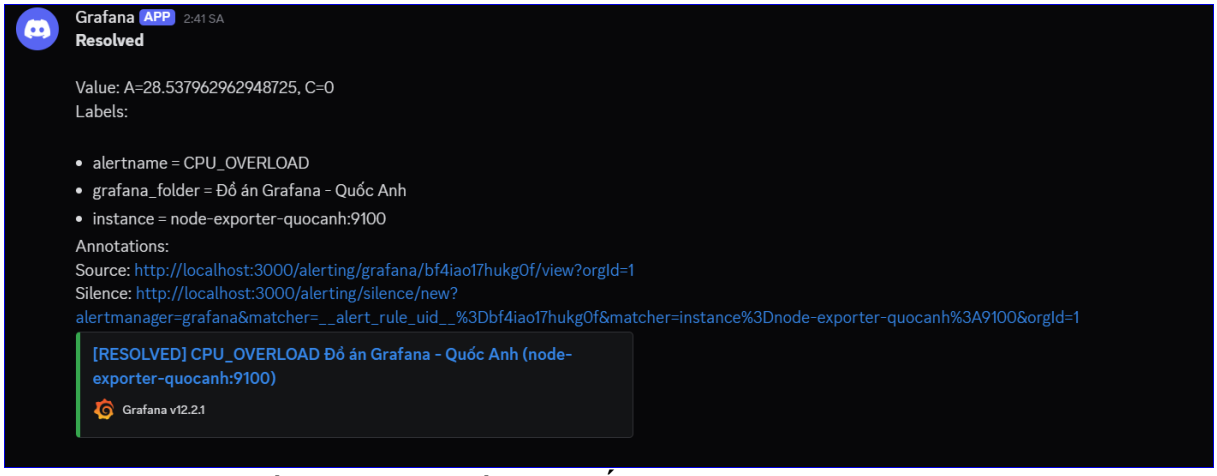
b, Trạng thái RESOLVED (Hệ thống khôi phục bình thường):

- Sau khoảng 1 phút (theo thời gian for 1m đã thiết lập cho Demo), Grafana xác nhận tình trạng quá tải kéo dài và chuyển Rule sang trạng thái **Firing** (Đang báo động).
- **Thao tác Khôi phục:** Chạy lệnh tiêu diệt toàn bộ các tiến trình yes đang chạy ngầm: **sudo killall yes**
- **Quá trình đánh giá lại trạng thái (Recovery):** Ngay lập tức, mức tiêu thụ CPU giảm sâu về mức bình thường (< 5%). Prometheus ghi nhận sự thay đổi này và gửi dữ liệu mới về Grafana.

- **Kết quả khôi phục:**
  - Sau khoảng 30 giây (theo chu kỳ đánh giá), Grafana nhận thấy chỉ số CPU đã nằm dưới ngưỡng 90% an toàn. Rule tự động chuyển từ trạng thái **Firing →Normal (OK).**
  - Panel trên dashboard đường biểu đồ đi ngang ổn định.
  - Discord ghi nhận một tin nhắn mới ở trạng thái **[RESOLVED]**, xác nhận sự cố quá tải đã chấm dứt.
  - **Bằng chứng 3:** Trạng thái trên Dashboard mức tiêu thụ CPU giảm dần xuống dưới mức 90%.

Hình 36: Biểu đồ CPU Usage giảm dần dưới mức 90%

- **Bằng chứng 4:** Tin nhắn **[RESOLVED]** được gửi tới kênh Discord (tự động).



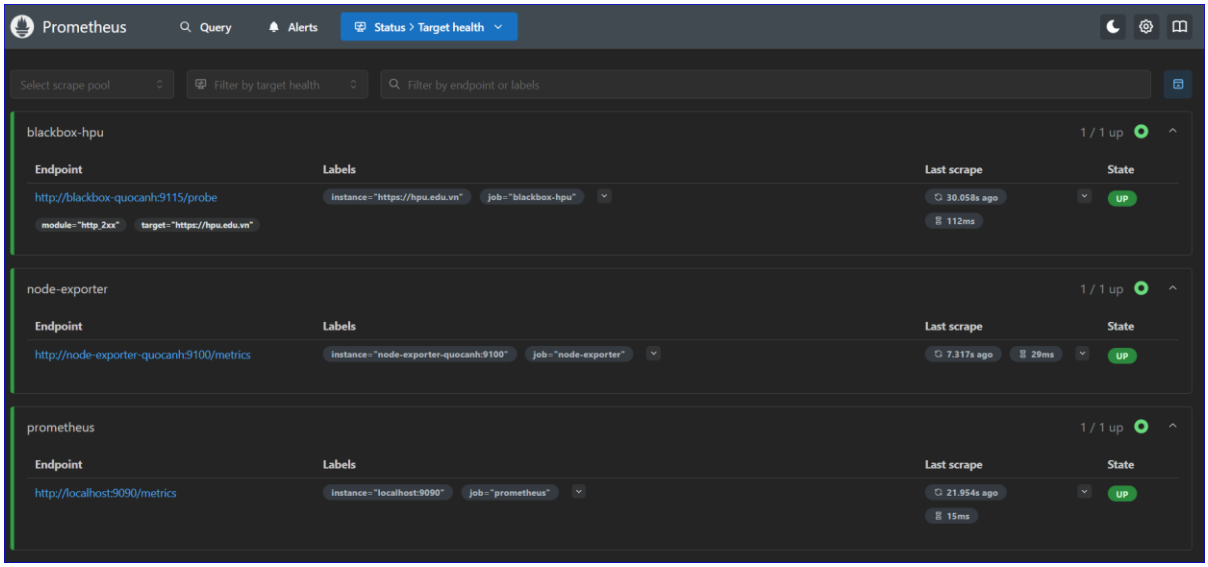
Hình 37: Kênh Discord nhận tin nhắn RESOLVED (CPU_OVERLOAD

Chương 4: KẾT QUẢ ĐẠT ĐƯỢC VÀ ĐÁNH GIÁ HIỆU QUẢ CỦA GRAFANA

4.1. Kết quả triển khai hệ thống giám sát

4.1.1 Kết quả thu được từ Prometheus

- Sau khi triển khai thành công Prometheus trong môi trường Docker, hệ thống đã thu thập dữ liệu ổn định từ các nguồn chính: Node Exporter, Blackbox Exporter, và các dịch vụ cài đặt trên VPS. Prometheus thực hiện việc scrape dữ liệu theo chu kỳ 5–15 giây tùy từng target.
- Toàn bộ các metric như CPU, RAM, Disk, Network… đều được thu thập chính xác và sẵn sàng cho truy vấn bằng PromQL.

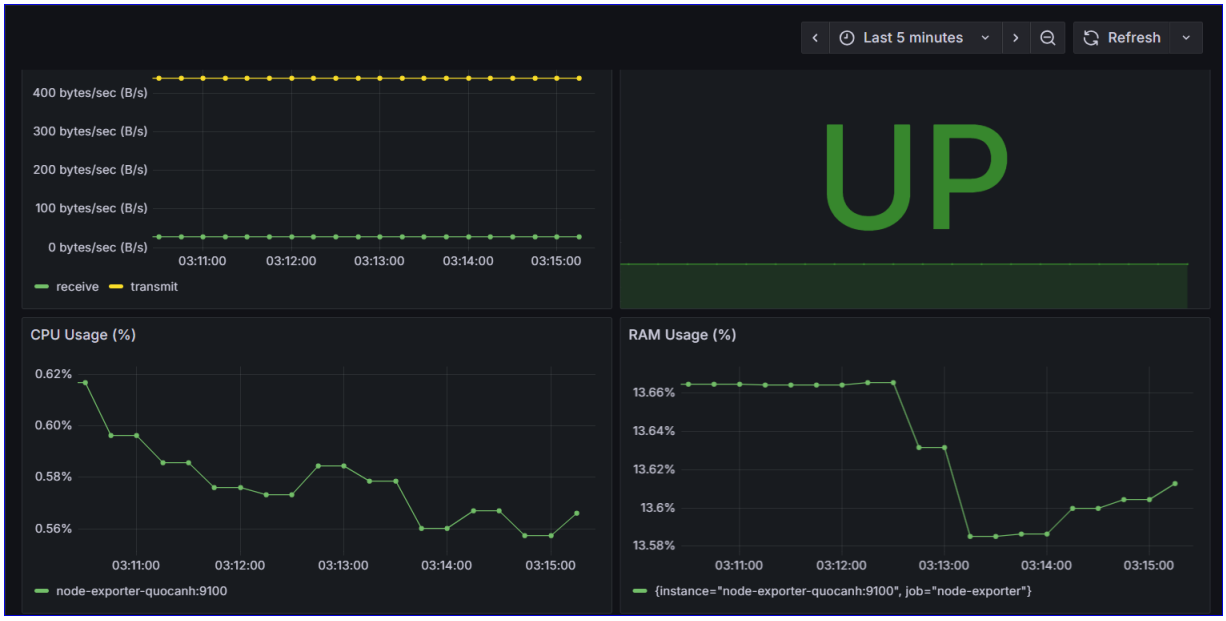


Hình 38: Chứng minh Prometheus sẵn sàng trực quan hóa dữ liệu trên Grafana.

- Bên cạnh đó, Prometheus cũng phản hồi nhanh với các thay đổi cấu hình (reload/restart), giúp việc thử nghiệm Rule và Blackbox Target diễn ra thuận lợi, phù hợp với mục tiêu xây dựng hệ thống giám sát linh hoạt.

4.1.2. Kết quả từ Grafana Dashboard

- Grafana đã được triển khai thành công và kết nối trực tiếp với Prometheus làm Datasource. Sau đó, hệ thống đã xây dựng và vận hành nhiều Dashboard trực quan gồm:
    - **Dashboard tài nguyên hệ thống**: hiển thị CPU load, memory usage, traffic mạng theo thời gian thực.
    - **Dashboard Website giám sát bằng Blackbox Exporter**: theo dõi tình trạng uptime.
- Tất cả biểu đồ đều hoạt động ổn định, tự động cập nhật theo thời gian thực và cho phép thao tác phóng to, lọc, xem chi tiết metric.



Hình 39: Chứng minh Dashboard giám sát tổng thể hoạt động ổn định

Kết luận: Điều này chứng minh hệ thống có khả năng trực quan hóa dữ liệu hiệu quả và phù hợp cho nhu cầu giám sát chuyên nghiệp.

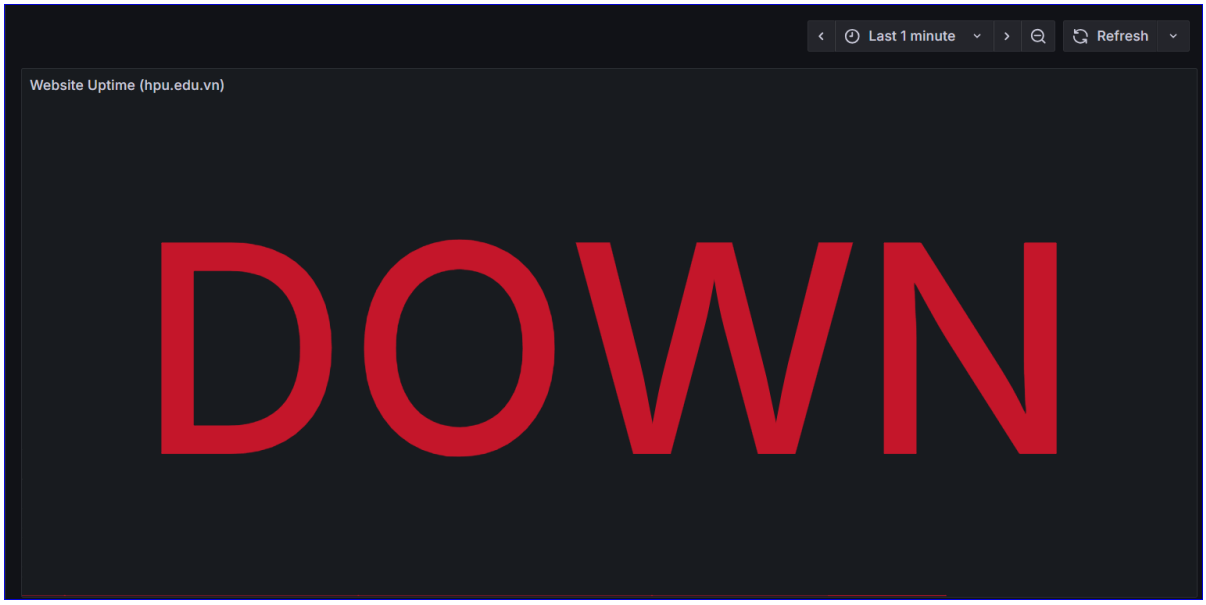
Chỉ số giám sát	Trạng thái ghi nhận	Đánh giá
CPU Usage	Ổn định (< 5%)	Hệ thống hoạt động dưới tải, tài nguyên dư thừa.
RAM Usage	Ổn định (~15%)	Không có hiện tượng rò rỉ bộ nhớ
Website Uptime	100% (UP)	Dịch vụ web hoạt động liên tục trong thời gian thử nghiệm.
Network Traffic	Ổn định	Không phát hiện tấn công mạng hay nghẽn băng thông.

Bảng 5: Kết quả giám sát từ Grafana

4.1.3. Kết quả về hệ thống cảnh báo (Alerting System)

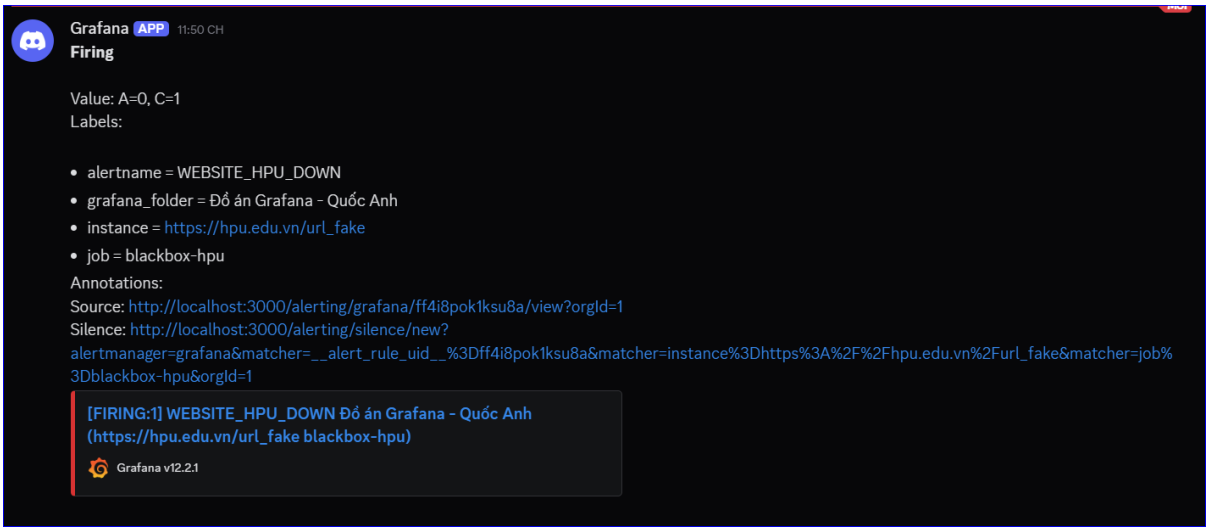
- **Hệ thống cảnh báo** đã được xây dựng dựa trên bộ Rule PromQL dành cho CPU, RAM, băng thông và đặc biệt là kiểm tra trạng thái hoạt động của Website HPU thông qua Blackbox Exporter.
- **Các điểm nổi bật:**
    - Alert hoạt động chính xác theo ngưỡng thiết lập.
    - Thời gian Pending và Firing phù hợp với cấu hình thời gian (for 30s -1m).
    - Tất cả cảnh báo được gửi thành công đến Discord Webhook.
- **Ví dụ tiêu biểu:** Trong quá trình thử nghiệm lỗi Website HPU, sau khi chỉnh URL trong **prometheus.yml** thành URL không tồn tại, Prometheus đã phát hiện tình trạng bất thường và kích hoạt Rule **WEBSITE_HPU_DOWN**.

- Hệ thống ngay lập tức chuyển sang trạng thái **Pending → Firing**, đồng thời Dashboard Grafana đổi màu cảnh báo sang đỏ.



Hình 40: Chứng minh Dashboard phát hiện và cảnh báo sự cố Website DOWN

- Tin nhắn cảnh báo cũng xuất hiện đầy đủ tại **Discord.**

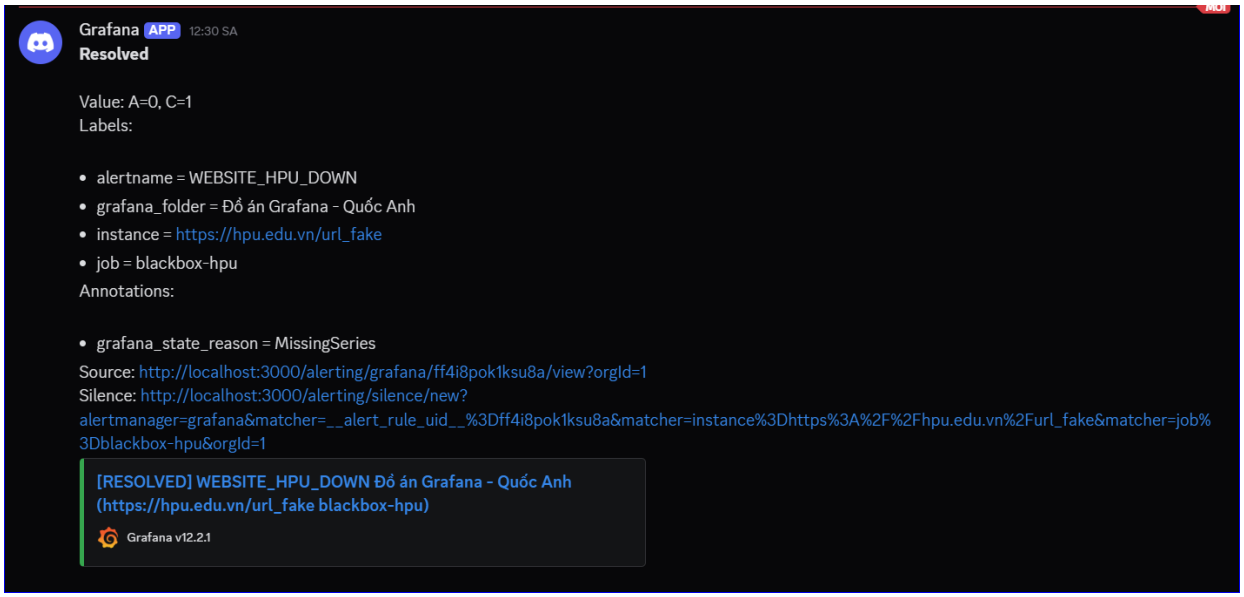


Hình 41: Chứng minh tin nhắn cảnh báo FIRING được gửi về kênh Discord

- Sau khi nhận được cảnh báo, quản trị viên đã kịp phát hiện và tiến hành khắc phục sự cố. Ngay khi Prometheus ghi nhận các chỉ số đã trở về ngưỡng an toàn, Grafana tự động chuyển trạng thái quy tắc từ **Firing** sang **Normal. Website UP trở lại.**

Hình 42: Chứng minh Dashboard khôi phục trở lại Website UP

- Đồng thời gửi tin nhắn xác nhận **RESOLVED** tới kênh Discord, đánh dấu quy trình xử lý sự cố đã hoàn tất.



Hình 43: Chứng minh tin nhắn RESOLVED được gửi về Discord

Kết luận: Điều này cho thấy hệ thống cảnh báo hoạt động hiệu quả và có tính ứng dụng thực tế cao.

4.2. Đánh giá hiệu quả giám sát bằng Grafana

4.2.1 Tính trực quan và dễ quan sát

- Giao diện của Grafana mang tính trực quan cao, dễ sử dụng và thân thiện với người quản trị.
- Các biểu đồ sử dụng màu sắc rõ ràng, khả năng tương tác mạnh mẽ và hỗ trợ đa dạng loại biểu diễn như Graph, Gauge, Bar Gauge, Stat Panel…
- Điều này hỗ trợ đắc lực trong việc theo dõi trạng thái hệ thống một cách nhanh chóng và chính xác.

4.2.2 Độ chính xác của dữ liệu

- Với cơ chế thu thập metric ổn định của Prometheus, dữ liệu được đưa vào Grafana luôn có độ trễ rất thấp (< 1 giây).
- Phản ánh chính xác trạng thái hiện tại của tài nguyên máy chủ và dịch vụ mạng, giúp loại bỏ các sai số do độ trễ đường truyền hoặc lỗi đồng bộ hóa."
- Nhờ cơ chế thu thập dữ liệu chủ động (Pull Model) từ Prometheus kết hợp với các Exporter chuyên dụng, dữ liệu hiển thị trên Grafana đảm bảo tính trung thực và độ tin cậy cao.

4.2.3 Khả năng phát hiện và phản ứng với sự cố

- Hệ thống cảnh báo giúp phát hiện nhanh các vấn đề như:
    - CPU và RAM sử dụng vượt mức cho phép.
    - Tốc độ truyền và nhận dữ liệu trên băng thông.
    - Website có đang hoạt động hay không.
- Nhờ vậy, thời gian phát hiện sự cố giảm đáng kể so với việc kiểm tra thủ công hoặc kiểm tra theo lịch.
- Việc gửi cảnh báo đến Discord giúp người quản trị có thể nhận thông tin ngay lập tức và xử lý kịp thời.

4.2.4 Lợi ích mang lại cho hệ thống

- Việc triển khaiPrometheus – Grafana đem lại nhiều lợi ích rõ rệt:
    - **Tăng độ tin cậy của dịch vụ** nhờ phát hiện lỗi sớm.

- **Hạn chế downtime**, góp phần nâng cao chất lượng vận hành.
- **Tiết kiệm thời gian** cho đội kỹ thuật khi phân tích sự cố.
- **Theo dõi toàn diện** tình trạng VPS và Website theo thời gian thực.
- **Tăng tính tự động hóa** trong quy trình giám sát.

4.3 Hạn chế của hệ thống giám sát

- Mặc dù hệ thống hoạt động tốt, nhưng vẫn còn một số hạn chế:
  - Chưa triển khai **High Availability** cho Prometheus (mới chạy single instance).
  - Dữ liệu Prometheus chủ yếu lưu trong container, dễ mất khi update nếu không gắn volume.
  - Blackbox Exporter mới kiểm tra HTTP/HTTPS, chưa mở rộng sang TCP, DNS, SMTP…
  - Khi số lượng node giám sát tăng cao, việc scrape có thể bị chậm nếu chưa tối ưu.
- Các hạn chế này sẽ được giải quyết trong chương tiếp theo với các đề xuất cải thiện.

CHƯƠNG 5: ĐỀ XUẤT CÁC GIẢI PHÁP TỐI ƯU HÓA VÀ NÂNG CAO HIỆU QUẢ GIÁM SÁT.

5.1. Tối ưu hóa hệ thống Prometheus – Grafana

5.1.1 Tối ưu Prometheus

- Sử dụng Volume Persist để lưu dữ liệu metric lâu dài.
- Tối ưu chu kỳ scrape để giảm tải hệ thống nhưng vẫn đảm bảo độ chính xác.
- Chia nhỏ các file rule để dễ quản lý hơn.
- Áp dụng Service Discovery nếu muốn mở rộng thêm nhiều server.
- Giảm số lượng metric không cần thiết để tiết kiệm tài nguyên.

5.1.2 Tối ưu Grafana

- Tổ chức Dashboard theo nhóm (System, Network, Website…).
- Sử dụng biến Dashboard (Variables) giúp tái sử dụng.
- Cấu hình Alerting theo Unified Alerting mới của Grafana.
- Tự động sao lưu Dashboard định kỳ.
- Sử dụng folder để phân loại rõ ràng.

5.2 Mở rộng hệ thống giám sát

5.2.1. Bổ sung các Exporter cần thiết

- Để hệ thống giám sát toàn diện, có thể bổ sung:
    - **MySQL Exporter** → Giám sát Database.
    - **Nginx Exporter** → Giám sát Webserver.
    - **Docker Exporter** → Giám sát container.
    - **Process Exporter** → Giám sát ứng dụng quan trọng.
- Việc mở rộng Exporter giúp hệ thống giám sát sâu hơn và phát hiện lỗi đa dạng hơn.

5.2.2. Giám sát nhiều server (Multi-node Monitoring)

- Khi hệ thống phát triển, có thể triển khai:
    - Node Exporter tại tất cả server.
    - Prometheus Federation để gom dữ liệu từ nhiều Prometheus con.

- Kết nối qua VPN/TLS để bảo mật khi thu thập dữ liệu.

- Điều này phù hợp cho doanh nghiệp hoặc trường học có nhiều hệ thống cần giám sát.

5.3. Tự động hóa cảnh báo

5.3.1. Tích hợp thêm kênh cảnh báo

- Hệ thống có thể mở rộng gửi cảnh báo sang:
    - Telegram Bot.
    - Slack API.
    - Zalo Official API.
    - Email SMTP.
    - SMS Gateway.
- Việc đa dạng hóa kênh cảnh báo giúp hệ thống linh hoạt và phù hợp với từng môi trường.

5.3.2. Cảnh báo nâng cao

- Ngoài ngưỡng cảnh báo cố định, có thể áp dụng:
    - Ngưỡng động dựa trên lịch sử hoạt động.
    - Cảnh báo khi có hành vi bất thường (anomaly detection).
    - Cảnh báo theo xu hướng (ví dụ RAM tăng dần 30 phút liên tục).
- Những kỹ thuật này giúp hệ thống cảnh báo thông minh và chính xác hơn.

5.4. Nâng cao độ tin cậy và tính ổn định

5.4.1. Prometheus HA (High Availability)

- Để tránh mất dữ liệu hoặc downtime:
    - Chạy 2 Prometheus song song.
    - Sử dụng Thanos để lưu trữ dài hạn và hợp nhất dữ liệu từ nhiều Prometheus.
    - Dùng Hardening và nén dữ liệu để tối ưu dung lượng.

5.4.2. Tối ưu hạ tầng phần cứng

- Dùng Hardening và nén dữ liệu để tối ưu dung lượng.
- Tách Grafana và Prometheus thành 2 VPS để giảm tải.

- Sử dụng SSD để tăng tốc độ đọc ghi metric.

5.4.3. Tăng cường bảo mật

- Đặt mật khẩu mạnh cho Grafana.
- Đặt sau Nginx Reverse Proxy.
- Bật HTTPS (Let’s Encrypt).
- Giới hạn IP truy cập.
- Kiểm tra log để ngăn truy cập trái phép.

5.5. Hướng phát triển trong tương lai

- Phát triển hệ thống cảnh báo thông minh sử dụng AI/ML.
- Tích hợp với CICD để tự động deploy dashboard và rule.
- Tự động scale Prometheus dựa trên tải.
- Phát triển thêm dashboard dành cho bảo mật hoặc log (kết hợp Loki).
- Ứng dụng thực tế tại doanh nghiệp, trường đại học hoặc dự án cá nhân.

KẾT LUẬN

Trong quá trình thực hiện đồ án tốt nghiệp với đề tài **"Nghiên cứu tìm hiểu và ứng dụng Grafana– Công cụ giám sát máy chủ và dịch vụ mạng tại trường Đại Học Quản lý và Công nghệ Hải Phòng"**, em đã nỗ lực tìm hiểu, nghiên cứu và triển khai hệ thống vào thực tế. Mặc dù đã hoàn thành các mục tiêu cơ bản đề ra, nhưng do năng lực chuyên môn của em còn nhiều hạn chế, nên đồ án chắc chắn không tránh khỏi những thiếu sót. Em rất mong nhận được sự đóng góp ý kiến, chỉ bảo của quý thầy/cô để đồ án được hoàn thiện hơn và em có thêm kinh nghiệm cho công việc sau này.

1. Kết quả đạt được

- Sau thời gian nghiên cứu và triển khai, đồ án đã đạt được những kết quả cụ thể sau:
  - **Về mặt lý thuyết:** Đã tìm hiểu và nắm vững kiến trúc hoạt động của các công cụ giám sát mã nguồn mở hiện đại là Prometheus (Thu thập dữ liệu), Grafana (Trực quan hóa) và các Exporter. Hiểu rõ sự khác biệt giữa giám sát trạng thái và giám sát hiệu năng.
  - **Về mặt thực tiễn:**
    - Xây dựng thành công hạ tầng giám sát trên nền tảng **Docker/VPS**.
    - Cấu hình thành công việc thu thập dữ liệu đa chiều: Từ tài nguyên phần cứng (Node Exporter) đến trạng thái dịch vụ Website (Blackbox Exporter).
    - Thiết kế hoàn thiện **Dashboard trực quan** hiển thị thời gian thực các chỉ số quan trọng (CPU, RAM, Network, Uptime).
    - Thiết lập thành công hệ thống **Cảnh báo tự động (Alerting)** qua kênh Discord, giúp phát hiện và thông báo sự cố (Website Down, CPU Overload) ngay lập tức.

2. Hướng phát triển cho tương lai

- Đồ án mang lại những giá trị thiết thực cho công tác quản trị mạng tại Trường Đại học Quản lý và Công nghệ Hải Phòng:

- **Nâng cao hiệu quả quản trị:** Chuyển đổi phương thức giám sát từ thụ động (đợi người dùng báo lỗi) sang chủ động (hệ thống tự phát hiện và cảnh báo), giúp giảm thiểu thời gian gián đoạn dịch vụ.
- **Tiết kiệm chi phí:** Việc ứng dụng các công cụ mã nguồn mở (Open-Source) giúp nhà trường xây dựng được hệ thống giám sát chuyên nghiệp, mạnh mẽ mà không tốn chi phí bản quyền phần mềm.
- **Tính ứng dụng cao:** Mô hình này dễ dàng nhân rộng để giám sát thêm nhiều máy chủ và dịch vụ khác trong trường mà không cần thay đổi kiến trúc hệ thống.

3. Hướng phát triển trong tương lai

- Để hệ thống trở nên hoàn thiện và mạnh mẽ hơn, em đề xuất các hướng phát triển tiếp theo:

- **Mở rộng phạm vi giám sát:** Tích hợp thêm các Exporter để giám sát Cơ sở dữ liệu (MySQL/PostgreSQL), giám sát Logs tập trung (sử dụng Loki) và giám sát thiết bị mạng (Switch/Router qua SNMP).
- **Tăng cường bảo mật:** Triển khai giao thức HTTPS và cơ chế xác thực người dùng chặt chẽ hơn cho Dashboard Grafana.
- **Tối ưu hóa cảnh báo:** Sử dụng Prometheus Alertmanager để quản lý, gộp nhóm các cảnh báo và định tuyến thông báo thông minh hơn (ví dụ: gửi SMS hoặc Email cho các lỗi nghiêm trọng).

TÀI LIỆU THAM KHẢO

1. Tài liệu chính thức của Grafana: https://grafana.com.docs/
2. Sách “Monitoring with Grafana” – Alan Hohn.
3. Các bài báo khoa học về giám sát hệ thống.