Hệ thống Phát Hiện Bất Thường cho Giám sát Hạ tầng Mạng

## Dùng dịch vụ AWS để tạo nên một chuỗi xử lí tự động phát hiện bất thường dữ liệu bằng Machine Learning với việc học không giám sát.

## 📄 1. Tóm tắt điều hành (Executive Summary)

Dự án này trình bày giải pháp xây dựng một hệ thống phát hiện bất thường (anomaly detection) cho mục tiêu giám sát mạng và bảo mật hạ tầng. Giải pháp tận dụng dữ liệu log dạng JSON từ hệ thống, phân tích bằng thuật toán học máy không giám sát (Random Cut Forest) để xác định các hành vi lạ có thể là dấu hiệu của tấn công hoặc cấu hình sai.

**Vấn đề thực tiễn:**

* Doanh nghiệp đang gặp khó khăn trong việc xử lý và giám sát lượng lớn log dữ liệu hệ thống.
* Các giải pháp thương mại như SIEM (Mắt thần an ninh) quá đắt đỏ với các nhóm nhỏ hoặc sinh viên.
* Hệ thống rule-based không đủ khả năng phát hiện các mẫu bất thường mới.

**Giải pháp đề xuất:**

* Sử dụng Amazon S3 để lưu trữ log JSON.
* Sử dụng Amazon SageMaker để huấn luyện và chạy mô hình Random Cut Forest nhằm phát hiện bất thường.
* Gửi cảnh báo tự động đến Discord thông qua webhook từ một bot tùy chỉnh.
* Thiết kế theo hướng đơn giản, sẽ hướng đến mở rộng lên chuỗi xử lý tự động (pipeline) thời gian thực trong tương lai.

**Lợi ích nổi bật:**

* Giảm chi phí giám sát hệ thống.
* Giảm rủi ro bảo mật và thời gian phát hiện sự cố.
* Dễ triển khai, phù hợp cho thực tập sinh hoặc tổ chức nhỏ.

**Thông tin đầu tư và lộ trình:**

* Hạ tầng sử dụng Free Tier của AWS.
* Thời gian triển khai từ 2 đến 3 tuần.
* Chi phí vận hành duy trì dưới 5 USD/tháng.

**Chỉ số mong đợi:**

* Thời gian xử lý log: đạt ≤ 5 phút/lần.
* Tỷ lệ cảnh báo giả (false positive): < 5%.
* Số lượng log xử lý/ngày: ~10.000 sự kiện.
* Chi phí AWS duy trì ở mức thấp hoặc miễn phí.

## 🎯 2. Định nghĩa vấn đề (Problem Statement)

### Tình trạng hiện tại

Các tổ chức phụ thuộc rất nhiều vào dữ liệu log để phát hiện sớm sự cố bảo mật hoặc lỗi hệ thống. Tuy nhiên, lượng dữ liệu lớn khiến việc giám sát trở nên khó khăn. Nhiều tổ chức nhỏ không có khả năng chi trả cho các công cụ SIEM (Mắt thần an ninh) thương mại.

### Những vấn đề gặp phải

* Quá tải khi xử lý log thủ công hoặc dùng các công cụ phân tích cơ bản.
* Hệ thống phát hiện dựa trên quy tắc không đủ linh hoạt để phát hiện mối đe dọa mới.
* Chi phí cao nếu thuê nền tảng SIEM chuyên nghiệp.

### Những bộ phận liên quan đến khía cạnh này

* Nhóm bảo mật (Security Operation): cần cảnh báo sớm, tránh bị động.
* Nhóm hạ tầng và vận hành hệ thống: cần phát hiện lỗi cấu hình hoặc sai lệch hoạt động.
* Bộ phận quản trị doanh nghiệp: mong muốn giảm thiểu rủi ro bảo mật và chi phí dài hạn.

### Hậu quả nếu không có một “mắt thần an ninh”

* Gia tăng nguy cơ bị tấn công mạng.
* Chi phí khắc phục hậu quả cao hơn nhiều so với chi phí phòng ngừa.
* Mất uy tín thương hiệu và nguy cơ vi phạm quy định pháp lý.

### Cơ hội thị trường

Theo báo cáo Gartner năm 2024, 85% doanh nghiệp sẽ ưu tiên các giải pháp bảo mật dựa trên đám mây vào năm 2025. Việc phát triển một hệ thống đơn giản, chi phí thấp là hướng đi phù hợp với nhu cầu hiện tại.

## 🏗️ 3. Thiết kế kiến trúc giải pháp (Solution Architecture)

### Kiến trúc tổng quan ban đầu

Giải pháp thiết kế dựa trên các dịch vụ serverless, dùng hạ tầng miễn phí của AWS. Log được lưu trữ trên Amazon S3, sau đó được phân tích theo từng đợt (batch 5 phút) bằng SageMaker. Mô hình học máy (Random Cut Forest) phát hiện điểm bất thường và gửi cảnh báo qua Discord Webhook.

### Dịch vụ AWS sử dụng

* **Amazon S3**: lưu trữ log JSON đầu vào.
* **Amazon SageMaker (Notebook Instance)**: nơi chạy code huấn luyện và dự đoán.
* **IAM**: phân quyền truy cập giữa các dịch vụ.
* Tùy chọn mở rộng: Lambda để tự động hóa chuỗi xử lý trong tương lai.

### Cấu trúc thành phần

1. Người dùng (hình thức ban đầu) hoặc hệ thống (hình thức tự động hóa) upload log dạng JSON lên S3.
2. SageMaker Notebook thực hiện phân tích theo lịch hoặc theo đợt (batch).
3. Mô hình Random Cut Forest tính điểm bất thường (anomaly score).
4. Nếu vượt ngưỡng cảnh báo, bot Discord sẽ gửi tin nhắn qua webhook vào kênh chỉ định.

### Kiến trúc bảo mật

* Sử dụng quyền IAM tối thiểu cần thiết.
* S3 được mã hóa dữ liệu tại chỗ (SSE-S3).
* Sử dụng HTTPS và kiểm tra quyền gọi API giữa các dịch vụ.

### Về giao diện

* Có thể tích hợp với bảng điều khiển (dashboard) sau này bằng Amazon QuickSight hoặc Grafana.

## 🔧 4. Kế hoạch triển khai kỹ thuật (Technical Implementation)

### Các giai đoạn triển khai

1. Thiết lập S3, tạo bucket và cấu hình IAM.
2. Phát triển SageMaker Notebook (huấn luyện + inference).
3. Tích hợp webhook Discord để gửi cảnh báo.
4. Tinh chỉnh tham số mô hình để giảm cảnh báo giả.
5. Triển khai Lambda để tự động hóa pipeline.

### Yêu cầu kỹ thuật

* Dữ liệu log dạng JSON, có chứa timestamp và thông tin hoạt động.
* Mô hình Random Cut Forest dùng để phát hiện bất thường.
* Bot Discord webhook để gửi cảnh báo.

### Phương pháp phát triển

* Tận dụng đoạn code RCF đã có để phát triển tiếp.
* Kiểm thử mô hình bằng tập cpu\_utilization\_asg\_misconfiguration.csv.
* Viết lại pipeline theo cách modular để dễ tích hợp thêm.

### Chiến lược kiểm thử

* Unit test: kiểm thử đọc log, xử lý dữ liệu, tính điểm bất thường.
* Kiểm thử tích hợp: đảm bảo hệ thống gửi đúng cảnh báo tới Discord.
* Kiểm thử hiệu năng: xử lý log 10.000 sự kiện/ngày.

### Kế hoạch triển khai ban đầu

* Mã nguồn lưu tại máy tính cá nhân.
* Môi trường phát triển: SageMaker Studio.
* Môi trường chạy: SageMaker Notebook + S3.

Trong quá trình phát triển sản phẩm, vì phát sinh chi phí dịch vụ ngoài mong đợi. Kế hoạch triển khai sẽ được điều chỉnh lại cho phù hợp với thực tế.

## 📅 5. Lộ trình và các mốc quan trọng (Timeline & Milestones)

### Timeline

* Tuần 1: Thiết lập hạ tầng, chuẩn bị dữ liệu.
* Tuần 2: Phát triển mô hình, kết nối Discord.
* Tuần 3: Tối ưu, viết tài liệu, thử nghiệm.

### Mốc chính

* M1: S3 bucket hoạt động, có log mẫu.
* M2: SageMaker chạy mô hình RCF.
* M3: Discord webhook hoạt động ổn định.
* M4: Kết quả đầu ra đáng tin cậy.

### Phân bổ nguồn lực

* 1 người thực hiện toàn bộ (có kiến thức AWS cơ bản).
* Sử dụng dịch vụ AWS Free Tier để tiết kiệm chi phí.

## 💰 6. Ước tính chi phí (Budget Estimation)

| **Thành phần** | **Dịch vụ** | **Ước tính chi phí** |
| --- | --- | --- |
| Lưu trữ | Amazon S3 | Miễn phí ≤ 5GB |
| Tính toán | SageMaker Notebook | Free Tier (250 giờ/tháng) |
| Gửi cảnh báo | Discord Webhook | Miễn phí |
| Tùy chọn (sau) | AWS Lambda | Miễn phí ≤ 1 triệu requests |
| Tổng cộng |  | ≈ 0–5 USD/tháng |

Chi phí hạ tầng là cực kỳ thấp, chỉ trả phí nếu vượt quá Free Tier hoặc mở rộng thêm.

## ⚠️ 7. Đánh giá rủi ro (Risk Assessment)

| **Rủi ro chính** | **Ảnh hưởng** | **Xác suất** | **Giảm thiểu** |
| --- | --- | --- | --- |
| Tăng chi phí do vượt Free Tier | Cao | Trung | Đặt báo động billing, giám sát tài nguyên |
| Mô hình cảnh báo sai (false positive) | Trung | Cao | Tinh chỉnh threshold, tăng dữ liệu huấn luyện |
| Lỗi khi gửi webhook Discord | Thấp | Trung | Viết lại logic retry, log lỗi cụ thể |
| Quyền IAM cấu hình sai | Cao | Thấp | Soát xét quyền định kỳ, test trước |

## ✅ 8. Kết quả kỳ vọng (Expected Outcomes)

### Chỉ số thành công

* Phát hiện bất thường trong thời gian ≤ 5 phút
* Giảm tỷ lệ cảnh báo giả dưới 5%
* Hệ thống hoạt động ổn định với 10.000 log/ngày
* Chi phí duy trì dưới 5 USD/tháng

### Lợi ích ngắn hạn

* Tự động hóa giám sát bất thường từ log
* Thay thế phần lớn thao tác thủ công
* Gửi cảnh báo kịp thời cho người vận hành

### Lợi ích trung và dài hạn

* Tích hợp thêm dashboard, biểu đồ phân tích
* Mở rộng thành chuỗi xử lý tự động thời gian thực (real-time pipeline)
* Ứng dụng mô hình cho dữ liệu khác ngoài CPU logs

#### 🔄 Điều chỉnh hướng đi dự án

Trong quá trình thực tập, em đã phát triển hình thức ban đầu cho hệ thống phát hiện bất thường, sử dụng Amazon SageMaker và Random Cut Forest (RCF). Hướng đi cũ đã chứng minh được khả năng phát hiện bất thường hiệu quả, trực quan hóa kết quả, và cho phép tinh chỉnh ngưỡng cảnh báo.

Tuy nhiên, quá trình thử nghiệm thực tế phát sinh **chi phí AWS ngoài dự kiến**, lên đến ~40 USD trong vòng 2 ngày do các sự cố về việc chưa nắm rõ dịch vụ, những dịch vụ hoạt động liên tục mà không tắt kịp. Điều này **vượt quá ngân sách sinh viên** và khó có thể phát triển về lâu dài.

Do đó, em dự kiến **chuyển hướng phát triển** sang thiết kế giải pháp:

* Tận dụng tối đa AWS Free Tier nếu có thể.
* Đơn giản, dễ triển khai, phù hợp thực tập sinh
* Vẫn giữ được năng lực phát hiện bất thường bằng Machine Learning.
* Hạn chế sử dụng tài nguyên trả phí ngoài Free Tier

## Minh họa kết quả và trực quan hóa từ bản thử nghiệm ban đầu

Trong giai đoạn phát triển ban đầu của workshop, nhóm đã thực hiện một **Proof of Concept (PoC)** với dữ liệu mẫu để kiểm chứng tính khả thi của ý tưởng.

✅ **Dataset sử dụng:**

* cpu\_utilization\_asg\_misconfiguration.csv
* Dạng dữ liệu: chuỗi thời gian (time series) với cột chính là CPU Utilization (%)

✅ **Tiền xử lý:**

* Chuyển đổi dữ liệu thành chuỗi thời gian đều đặn.
* Loại bỏ giá trị thiếu, làm sạch dữ liệu.

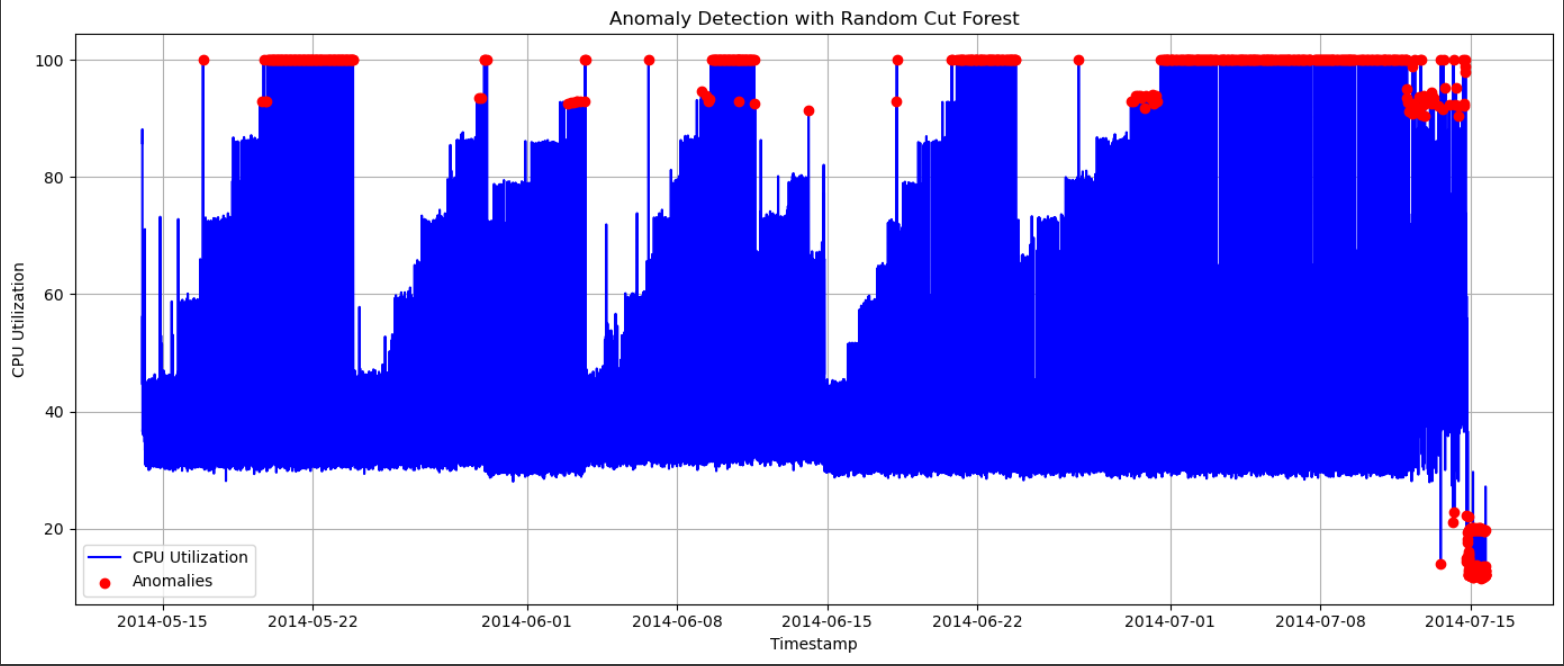
✅ **Thuật toán:**

* Random Cut Forest (RCF) – unsupervised anomaly detection.
* Đã triển khai trên Jupyter Notebook (SageMaker).

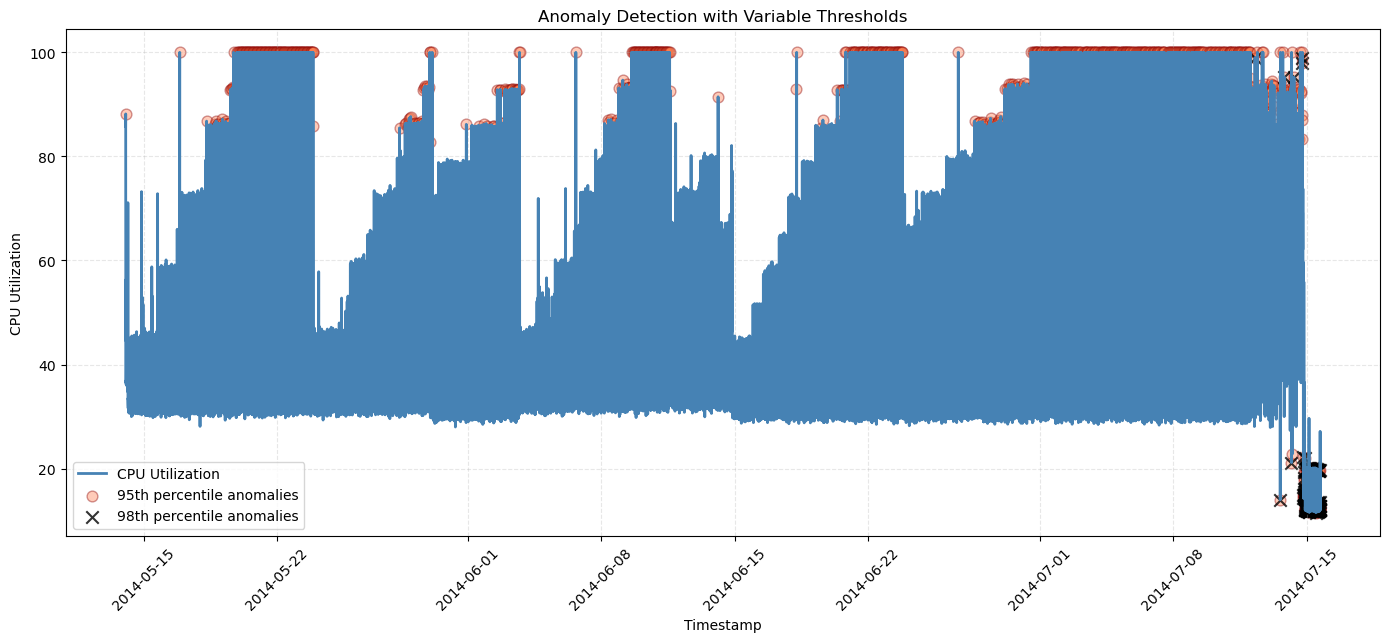
✅ **Kết quả trực quan hóa:**

* Biểu đồ thể hiện giá trị CPU Utilization theo thời gian.
* Overlay các điểm được gắn nhãn “bất thường” trên biểu đồ.
* Cho phép trực quan phát hiện các spike hoặc bất thường rõ rệt.

Ví dụ minh họa từ notebook (ảnh chụp màn hình):

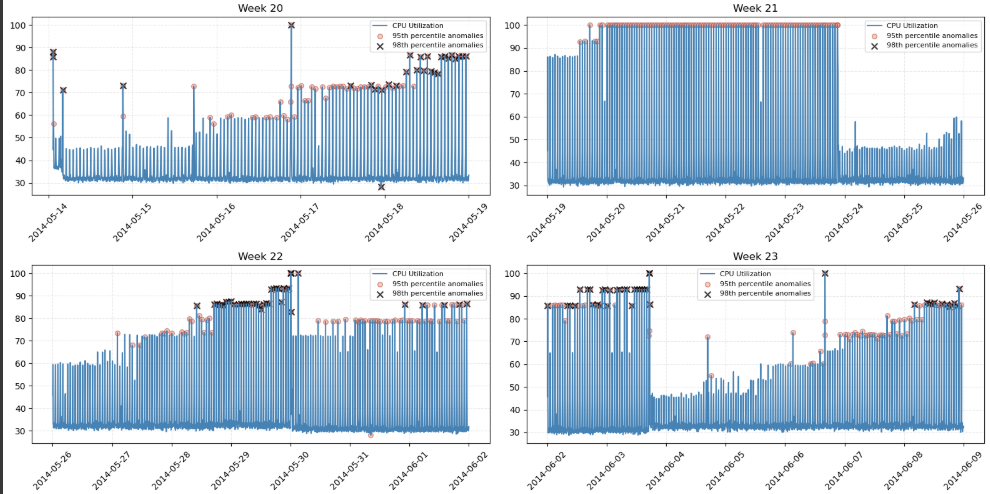


Trong đồ thị trên, các điểm màu đỏ là bất thường theo ngưỡng threshold đã chọn.



Ở đồ thị thứ hai, đặt ra hai ngưỡng là 5% các giá trị Anomaly(bất thường) cao nhất được đánh màu cam và top 2% giá trị có chỉ số Anomaly cao nhất sẽ là dấu X đen.

Ngoài ra, em còn visualize theo từng tuần để dữ liệu được rõ ràng hơn.



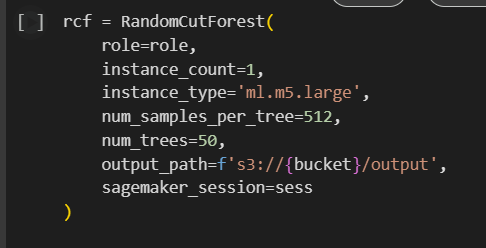
✅ **Ý nghĩa:**

* Đã chứng minh pipeline kỹ thuật hoạt động:
  + Load dữ liệu log → Huấn luyện mô hình → Phát hiện bất thường → Đánh dấu trực quan.
* Là cơ sở quan trọng để nhóm tự tin mở rộng sang dữ liệu log JSON hoặc nguồn thời gian thực (sau này).

## 📎 Phụ lục

**A. Thông số kỹ thuật mô hình**

* Random Cut Forest



* Threshold tùy chỉnh theo tập dữ liệu.

Threshold hiện tại đang được tinh chỉnh ở ngưỡng 2.0, 2.44, 2.5 và 3.0; Ngoài ra còn sẽ còn tham khảo đến những phương pháp lọc biến bất thường khác.

**B. Ước tính chi phí**

* Theo AWS Pricing Calculator và Free Tier

**C. Sơ đồ kiến trúc hệ thống**

* (Hiện chưa có)

**D. Tài liệu tham khảo**

* AWS Random Cut Forest
* Tài liệu về SageMaker Notebook
* Dữ liệu mẫu: cpu\_utilization\_asg\_misconfiguration.csv
* Discord Webhook API Docs

## Thực tập sinh – Người thực hiện dự án

## Nguyễn Huy Hùng