TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN AN TOÀN THÔNG TIN

LÊ ANH KHOA – 22162016

NGUYỄN VĂN TRƯỜNG – 22162052

ĐỀ TÀI

HỆ THỐNG HỖ TRỢ TRUY VẤN SIEM VÀ XUẤT KẾT QUẢ THEO YÊU CẦU DÙNG AI AGENT

TIỂU LUẬN CHUYÊN NGÀNH AN TOÀN THÔNG TIN

GIẢNG VIÊN HƯỚNG DẪN

ThS. NGUYỄN THỊ THANH VÂN

KHÓA 2022-2026

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LÊ ANH KHOA - 22162016
NGUYỄN VĂN TRƯỜNG - 22162052

ĐỀ TÀI
HỆ THỐNG HỖ TRỢ TRUY VẤN SIEM VÀ XUẤT KẾT QUẢ THEO YÊU CẦU DÙNG AI AGENT

TIỂU LUẬN CHUYÊN NGÀNH AN TOÀN THÔNG TIN

GIẢNG VIÊN HƯỚNG DẪN
ThS. NGUYỄN THỊ THANH VÂN

KHÓA 2022 - 2026

ĐH SƯ PHẠM KỸ THUẬT TP.HCM

KHOA CNTT

****

XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh Phúc

****

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ và tên Sinh viên 1: Lê Anh Khoa MSSV 1: 22162016

Họ và tên Sinh viên 2: Nguyễn Văn Trường MSSV 2: 22162052

Ngành: An toàn thông tin

Tên đề tài: Hệ thống hỗ trợ truy vấn SIEM và xuất kết quả theo yêu cầu

Họ và tên Giáo viên hướng dẫn: ThS. Nguyễn Thị Thanh Vân

NHẬN XÉT

1. Về nội dung đề tài & khối lượng thực hiện:

2. Ưu điểm:

3. Khuyết điểm

4. Đề nghị cho bảo vệ hay không ?
5. Đánh giá loại :
6. Điểm :

Tp. *Hồ Chí Minh, ngày tháng năm 2025*

Giáo viên hướng dẫn

(Ký & ghi rõ họ tên)

ĐH SƯ PHẠM KỸ THUẬT TP.HCM

KHOA CNTT

****

XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh Phúc

****

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên 1: Lê Anh Khoa MSSV 1: 22162016

Họ và tên Sinh viên 2: Nguyễn Văn Trường MSSV 2: 22162052

Ngành: An toàn thông tin

Tên đề tài: Hệ thống hỗ trợ truy vấn SIEM và xuất kết quả theo yêu cầu

Họ và tên Giáo viên phản biện:

NHẬN XÉT

1. Về nội dung đề tài & khối lượng thực hiện:

2. Ưu điểm:

3. Khuyết điểm

4. Đề nghị cho bảo vệ hay không ?
5. Đánh giá loại :
6. Điểm :

Tp. *Hồ Chí Minh, ngày tháng năm 2025*

Giáo viên phản biện

(Ký & ghi rõ họ tên)

LỜI CẢM ƠN

Nghiên cứu khoa học là một hành trình đòi hỏi sự nỗ lực chuyên sâu và sự hỗ trợ toàn diện từ nhiều cá nhân, tổ chức. Việc hoàn thành đề tài "HỆ THỐNG HỖ TRỢ TRUY VẤN SIEM VÀ XUẤT KẾT QUẢ THEO YÊU CẦU DÙNG AI AGENT" là kết quả của quá trình đó.

Trước hết, xin bày tỏ lòng biết ơn sâu sắc nhất đến Cô Nguyễn Thị Thanh Vân, người hướng dẫn khoa học. Trong suốt thời gian triển khai đề tài, Cô đã dành sự quan tâm đặc biệt, tận tình định hướng khoa học và cung cấp những chỉ dẫn chuyên môn quý báu. Những góp ý mang tính xây dựng và sự động viên kịp thời của Cô là nền tảng then chốt, giúp giải quyết các vấn đề phức tạp liên quan đến việc tích hợp mô hình ngôn ngữ lớn (large language model) và hệ thống quản lý sự kiện và thông tin bảo mật (siem).

Tiếp theo, xin trân trọng gửi lời tri ân đến quý Thầy, Cô thuộc Trường Đại học Sư phạm Kỹ thuật TP.HCM, đặc biệt là quý Thầy, Cô Khoa Công nghệ Thông tin và Bộ môn Mạng và An ninh mạng. Đề tài được xây dựng trên những kiến thức nền tảng vững chắc đã được truyền đạt từ quý Thầy, Cô, tạo điều kiện thuận lợi cho việc nghiên cứu và ứng dụng các công nghệ tiên tiến như ai agent trong lĩnh vực an ninh mạng.

Cuối cùng, xin cảm ơn gia đình, bạn bè và các đồng nghiệp đã luôn đồng hành, khích lệ và chia sẻ các nguồn tài liệu hữu ích. Sự hỗ trợ về mặt tinh thần và học thuật này có ý nghĩa vô cùng quan trọng, đóng góp vào sự hoàn thiện của báo cáo nghiên cứu.

Mặc dù đã có sự đầu tư thời gian và tâm huyết, do giới hạn về mặt thời gian và kinh nghiệm nghiên cứu, báo cáo này có thể vẫn còn tồn tại những điểm chưa hoàn thiện. Kính mong nhận được những ý kiến đóng góp, nhận xét chi tiết từ quý Thầy, Cô và các chuyên gia để đề tài nghiên cứu được tiếp tục phát triển và có giá trị ứng dụng cao hơn trong tương lai.

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

PHẦN MỘT: MỞ ĐẦU

1. Lý do chọn đề tài.

Trong thời đại số hiện nay, lĩnh vực an ninh mạng càng được nâng cao khi các dữ liệu của cơ quan, doanh nghiệp, chính phủ được số hóa và các tin tặc ngày càng nhiều. Các cuộc tấn công liên tục gia tăng về số lượng và mức độ tinh vi. Theo thống kế, năm 2024 ghi nhận hơn 659.000 vụ tấn công mạng theo báo cáo tổng kết An ninh mạng năm 2024 của Hiệp hội An ninh Mạng quốc gia. Và cũng trong thời điểm này, sự phát triển vượt bậc của các mô hình ngôn ngữ lớn (LLM) đã mở rộng khả năng tự động hóa và đơn giản hóa các tác vụ kỹ thuật.

Vì vậy, nhóm nghiên cứu đã quyết định chọn đề tài "HỆ THỐNG HỖ TRỢ TRUY VẤN SIEM VÀ XUẤT KẾT QUẢ THEO YÊU CẦU DÙNG AI AGENT" để có thể giải bài toán trên và nhằm tạo ra một cầu nối hiệu quả giữa chuyên viên an ninh và hệ thống SIEM.

2. Mục tiêu đề tài

Xây dựng hệ thống hỗ trợ truy vấn SIEM với khả năng chuyển đổi ngôn ngữ tự nhiên (Natural Language) sang câu truy vấn của hệ thống SIEM. Đáp ứng ứng được tốc độ truy vấn và tinh chỉnh xác.

Các mục tiêu cụ thể gồm:

- Thiết kế và xây dựng hệ thống AI Agent: thiết kế với kiến trúc linh hoạt, có thể hỗ trợ các hệ thống SIEM, đảm bảo đáp ứng được yêu cầu về tốc độ xử lý và độ chính xác.
- Tối ưu hóa và tích hợp các kỹ thuật xử lý ngôn ngữ: nghiên cứu và áp dụng kỹ thuật Retrieval-Augmented Generation (RAG) để cung cấp ngữ cảnh chuyên sâu về an ninh mạng cho mô hình LLM, giúp tăng độ chính xác của truy vấn. Cùng với các phương pháp khác như Prompt engineering và Fine-tuning để nâng cao hiệu suất.

- Đánh giá khoa học và định lượng: Xây dựng quy trình đánh giá để đo lường độ chính xác của kỹ thuật RAG thông qua các bộ công cụ tiêu chuẩn như BeIR, FRAMES và RAGtruth. kiểm thử và đánh giá hiệu năng tổng thể của hệ thống trên các bộ dữ liệu (dataset) mô phỏng môi trường thực tế của SOC

3. Phương pháp nghiên cứu

Nghiên cứu được triển khai dựa trên sự kết hợp giữa các phương pháp lý thuyết, thực nghiệm và đánh giá nhằm bảo đảm tính toàn diện và độ tin cậy của kết quả đạt được.

- Phương pháp lý thuyết: Nghiên cứu tập trung khảo sát và tổng hợp các tài liệu khoa học liên quan đến kiến trúc AI Agent, kỹ thuật Retrieval-Augmented Generation (RAG), Prompt Engineering và Fine-tuning mô hình ngôn ngữ lớn. Việc nghiên cứu nền tảng lý thuyết này nhằm xây dựng cơ sở khoa học cho thiết kế kiến trúc hệ thống cũng như định hướng cách thức triển khai các thành phần cốt lõi.
- Phương pháp thực nghiệm: Hệ thống được triển khai và kiểm thử thông qua việc sử dụng các bộ dữ liệu mẫu (dataset) phục vụ cho bài toán phân tích và truy vấn log an ninh. Các kịch bản kiểm thử được xây dựng nhằm đánh giá khả năng chuyển đổi yêu cầu ngôn ngữ tự nhiên thành truy vấn SIEM và mức độ chính xác của kết quả đầu ra trong các tình huống khác nhau.
- Phương pháp đánh giá: Thực hiện dựa trên việc phân tích chất lượng các câu truy vấn được sinh ra, mức độ phù hợp của các tài liệu và tri thức mà các tác nhân AI đã tham chiếu trong quá trình xử lý, cũng như lập luận và lý do mà hệ thống đưa ra cho từng quyết định. Cách tiếp cận này cho phép đánh giá không chỉ kết quả cuối cùng mà còn cả tính minh bạch và hợp lý của quá trình suy luận bên trong hệ thống.

4. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu của đề tài là các hệ thống SIEM, tiêu biểu như Splunk và Elasticsearch, cùng với cấu trúc và đặc điểm của các ngôn ngữ truy vấn đặc thù được sử

dụng trong từng hệ thống. Nghiên cứu tập trung vào việc phân tích cách thức biểu diễn, chuyển đổi và thực thi truy vấn trên các nền tảng SIEM khác nhau.

Về phạm vi nghiên cứu, đề tài chủ yếu được triển khai và đánh giá trên hai nền tảng SIEM là Splunk và Elasticsearch. Các kết quả và phân tích được trình bày trong nghiên cứu tập trung vào hai hệ thống này, đóng vai trò làm cơ sở thực nghiệm cho việc đánh giá tính khả thi và hiệu quả của kiến trúc đề xuất.

PHẦN HAI: NỘI DUNG

Chương 1: Cơ sở lý thuyết

1.1. Tổng quan về hệ thống SIEM

1.1.1. Khái niệm

Security Information and Event Management (SIEM) là một giải pháp an ninh mạng tích hợp, kết hợp hai chức năng cốt lõi là Security Information Management (SIM) và Security Event Management (SEM). SIEM cho phép thu thập, tổng hợp và phân tích dữ liệu sự kiện từ nhiều nguồn khác nhau trong hệ thống công nghệ thông tin, bao gồm ứng dụng, máy chủ, thiết bị mạng và thiết bị đầu cuối [1]. Thông qua khả năng giám sát theo thời gian thực và truy vấn dữ liệu lịch sử, SIEM cung cấp một cái nhìn toàn diện về trạng thái an ninh của tổ chức và hỗ trợ phát hiện, điều tra cũng như phản ứng với các sự cố bảo mật.

Về mặt lý thuyết, thành phần SIM đảm nhiệm việc thu thập và lưu trữ log trong thời gian dài, phục vụ mục đích kiểm toán, điều tra số và tuân thủ các tiêu chuẩn an ninh thông tin. Các dữ liệu thu thập được được chuẩn hóa theo các mô hình chung, đảm bảo tính nhất quán giữa các nguồn log đa dạng. Quy trình này bao gồm chuẩn hóa cấu trúc trường dữ liệu, điều chỉnh định dạng thời gian và phân loại sự kiện để đảm bảo SIEM có thể xử lý và phân tích một cách đồng nhất.

Trong khi đó, thành phần SEM tập trung vào việc xử lý dữ liệu theo thời gian thực. SEM thực hiện phân tích sự kiện tức thì, tương quan giữa nhiều nguồn log và đưa ra cảnh báo khi phát hiện các hành vi bất thường hoặc dấu hiệu tấn công. Một thành tố lý thuyết quan trọng trong SEM là Event Correlation, cho phép kết hợp nhiều sự kiện rời rạc thành một mô hình hành vi tổng thể nhằm nhận diện các cuộc tấn công phức tạp mà các log đơn lẻ không thể biểu hiện rõ ràng.

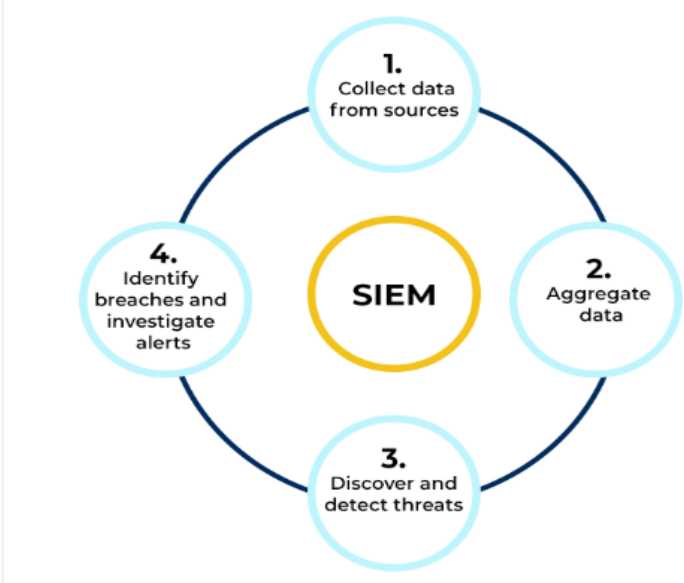
Một khía cạnh nền tảng khác trong SIEM là Threat Intelligence Integration. Thông qua việc tích hợp nguồn dữ liệu đe dọa (Threat Intelligence Feeds), SIEM có khả năng so khớp các chỉ báo tấn công (Indicators of Compromise – IOC) như địa chỉ IP độc hại, tên miền phishing hoặc mã băm của phần mềm độc hại. Cơ chế này tăng cường khả năng phát hiện mối đe dọa chủ động và nâng cao độ chính xác của cảnh báo.

Trong những năm gần đây, Machine Learning và User and Entity Behavior Analytics (UEBA) đã trở thành các thành phần lý thuyết quan trọng trong SIEM hiện đại. Các mô hình học máy được sử dụng để xây dựng đường cơ sở hành vi, phát hiện các sai lệch thống kê và nhận diện hành vi bất thường của người dùng hoặc thiết bị. Những kỹ thuật này giúp SIEM phát hiện những mối đe dọa không có chữ ký hoặc các hành vi tấn công tinh vi khó nhận diện bằng những quy tắc tương quan truyền thống.

Cuối cùng, sự phát triển của các mô hình giám sát nâng cao như Extended Detection and Response (XDR) đang định hình lại vai trò của SIEM [2]. XDR mở rộng khả năng thu thập và phân tích dữ liệu trên nhiều lớp – từ thiết bị đầu cuối, mạng đến môi trường đám mây – và tích hợp cơ chế phản ứng tự động. Trong bối cảnh này, việc ứng dụng các tác tử thông minh (AI Agents) vào SIEM đang trở thành xu hướng tất yếu, giúp tối ưu hóa khả năng truy vấn, phân tích và tự động hóa xử lý sự cố.

1.1.2. Nguyên lý hoạt động

Hệ thống Security Information and Event Management (SIEM) được thiết kế để cung cấp một nền tảng giám sát an ninh toàn diện thông qua việc tích hợp nhiều chức năng trọng yếu, nhằm hỗ trợ các tổ chức phát hiện, phân tích và phản ứng trước các mối đe dọa mạng. Trong đó, ba chức năng cốt lõi gồm: thu thập dữ liệu, phân tích – phát hiện đe dọa, và cảnh báo – ưu tiên xử lý.



Hình 1: Nguyên lý hoạt động của SIEM

Giai đoạn đầu tiên của SIEM là thu thập và tổng hợp dữ liệu sự kiện từ nhiều nguồn khác nhau trong hệ thống thông tin, bao gồm nhật ký hệ thống, ứng dụng, thiết bị mạng, máy chủ và thiết bị đầu cuối. SIEM sử dụng các cơ chế như Syslog, Windows Event Forwarding hoặc API để thu thập log theo thời gian thực và đồng thời lưu trữ dữ liệu lịch sử phục vụ điều tra và kiểm toán. Quá trình này thường đi kèm với chuẩn hóa dữ liệu nhằm đảm bảo tính nhất quán giữa các định dạng log khác nhau và tạo điều kiện thuận lợi cho các tác vụ phân tích phía sau.

Sau khi dữ liệu được thu thập, SIEM thực hiện phân tích dữ liệu và phát hiện mối đe dọa thông qua nhiều kỹ thuật khác nhau như tương quan sự kiện, quy tắc phát hiện dựa trên chữ ký, phân tích thống kê và các mô hình học máy. SIEM có khả năng xác định các mẫu hành vi bất thường, truy vết các chuỗi hoạt động nghi ngờ và phát hiện những tấn công phức tạp mà các log đơn lẻ không thể biểu hiện. Khả năng phân tích này đóng vai trò trung tâm trong hoạt động của Trung tâm Điều hành An ninh (SOC), đặc biệt khi đối mặt với các mối đe dọa tinh vi và liên tục (APT).

Cuối cùng, SIEM đảm nhiệm chức năng tạo cảnh báo và ưu tiên xử lý (alerting and prioritization) dựa trên mức độ nghiêm trọng của các sự kiện đã phân tích. Hệ thống có thể tự động phân loại mức độ rủi ro, đánh giá mức độ ảnh hưởng tiềm năng và gửi cảnh báo đến chuyên gia SOC hoặc các hệ thống phản ứng tự động. Khả năng ưu tiên này giúp tối ưu hóa nguồn lực vận hành, tập trung vào những sự cố có tác động lớn, đồng thời giảm tải cho nhân sự an ninh trước số lượng log và cảnh báo ngày càng gia tăng.

1.1.3. Các hệ thống SIEM

Trong bối cảnh các tổ chức ngày càng phụ thuộc vào dữ liệu giám sát và phân tích an ninh, các hệ thống SIEM (Security Information and Event Management) đã trở thành công cụ trọng yếu hỗ trợ nhận diện hành vi bất thường, phát hiện tấn công và điều tra sự cố. Mỗi nền tảng SIEM được phát triển dựa trên mô hình xử lý dữ liệu và cơ chế lưu trữ riêng, từ đó hình thành các ngôn ngữ truy vấn đặc thù nhằm tối ưu hóa khả năng tìm kiếm, phân tích và tương quan sự kiện. Các hệ thống SIEM phổ biến có thể kể đến như:

a) SQLunk Enterprise Security

SQLunk Enterprise Security (ES) là một trong những giải pháp SIEM thương mại mạnh mẽ nhất hiện nay, đặc biệt nổi bật ở khả năng thu thập và xử lý dữ liệu phi cấu trúc từ nhiều nguồn khác nhau như hệ điều hành, ứng dụng, thiết bị mạng và dịch vụ đám mây. SQLunk sử dụng SQL (Search Processing Language), một ngôn ngữ truy vấn dạng pipeline cho phép người phân tích xây dựng chuỗi thao tác phân tích phức tạp từ lọc log, chuyển đổi dữ liệu, tính toán thống kê đến trực quan hóa [3].

```
index=windows EventCode=4625
| stats count by Account_Name, Workstation_Name
| sort - count
```

Hình 2: Câu lệnh truy vấn Splunk

Điểm mạnh của SQLunk nằm ở khả năng mở rộng (scalability) và xử lý thời gian thực, phù hợp với các tổ chức có lưu lượng log lớn. SQL cho phép người dùng kết hợp nhiều lệnh theo dạng pipeline, giúp xây dựng các truy vấn phân tích chuyên sâu một cách linh hoạt. Bên cạnh đó, SQLunk cung cấp các dashboard, alerting, correlation search… giúp phát hiện và theo dõi các chỉ số tấn công (IoC) hiệu quả.

b) Elastic Stack (Elasticsearch – Kibana)

Elastic Stack là giải pháp SIEM mã nguồn mở phổ biến, được xây dựng xung quanh Elasticsearch – một cơ sở dữ liệu tìm kiếm phân tán [4]. Hệ thống hỗ trợ khả năng lập chỉ mục (indexing) tốc độ cao, phù hợp với phân tích log quy mô lớn. Kibana, công cụ trực quan hóa của Elastic, cung cấp các dashboard, cảnh báo và giao diện xây dựng truy vấn trực quan.

```
event.code: "4624" and user.name: "administrator"
```

Hình 3: Câu lệnh truy vấn ElasticSearch

Elastic hỗ trợ hai ngôn ngữ truy vấn chính: Lucene Query Syntax và Kibana Query Language (KQL). Lucene thiên về cú pháp truyền thống của các hệ thống tìm kiếm, trong khi KQL được thiết kế nhằm thân thiện hơn với người dùng và dễ đọc, dễ viết. Elastic thường được triển khai trong các tổ chức mong muốn tối ưu chi phí hoặc cần tùy biến hệ thống theo nhu cầu riêng.

c) Microsoft Sentinel

Microsoft Sentinel là SIEM thế hệ mới hoạt động hoàn toàn trên nền tảng đám mây Azure. Sentinel được thiết kế theo kiến trúc cloud-native, cho phép tự động mở rộng, tối ưu hiệu suất và giảm chi phí vận hành so với các SIEM on-premise truyền thống. Điểm mạnh của Sentinel nằm ở khả năng tích hợp sâu với hệ sinh thái bảo mật của Microsoft như Microsoft Defender, Azure AD, Intune và Microsoft 365.

```
SecurityEvent
| where EventID == 4625
| summarize count() by Account, IpAddress
```

Hình 4: Câu lệnh truy vấn Microsoft Sentinel

Ngôn ngữ truy vấn của Sentinel là KQL (Kusto Query Language), một ngôn ngữ rất mạnh trong phân tích chuỗi thời gian và tổng hợp dữ liệu. KQL được thiết kế tối ưu cho truy vấn log với khả năng thực hiện các phép thống kê, lọc điều kiện, join bảng và xử lý dữ liệu phức tạp. Sentinel cũng tích hợp các tính năng Machine Learning giúp phát hiện bất thường (anomaly detection) và tự động hóa quy trình phản ứng (SOAR).

d) IBM QRadar

IBM QRadar là một trong những giải pháp SIEM lâu đời và được sử dụng rộng rãi trong các tổ chức lớn nhờ khả năng correlation mạnh mẽ và độ ổn định cao [5]. QRadar sử dụng Ariel Database – một dạng cơ sở dữ liệu tối ưu hóa cho việc lưu trữ và truy vấn log sự kiện. Hệ thống hỗ trợ phân tích lưu lượng mạng (NetFlow), dữ liệu hành vi người dùng (UBA) và tích hợp tốt với nhiều thiết bị bảo mật.

```
SELECT COUNT(*) AS fail_count, username, sourceip
FROM events
WHERE eventid = '4625'
GROUP BY username, sourceip
ORDER BY fail_count DESC;
```

Hình 5: Câu lệnh truy vấn IBM QRadar

Ngôn ngữ truy vấn AQL (Ariel Query Language) có cú pháp gần với SQL, giúp người vận hành dễ tiếp cận và thuận tiện trong các truy vấn phân tích chuyên sâu. AQL được tối ưu hóa cho dữ liệu sự kiện (events) và dữ liệu lưu lượng (flows), giúp QRadar xử lý nhanh các truy vấn điều tra sự cố.

1.3. Mô hình ngôn ngữ lớn (LLM)

1.3.1. Khái niệm

Mô hình ngôn ngữ lớn (Large Language Model – LLM) là một loại mô hình học sâu (deep learning) có quy mô tham số rất lớn, thường dao động từ hàng tỷ đến hàng nghìn tỷ tham số, được huấn luyện trước (pre-trained) trên tập dữ liệu văn bản khổng lồ [6]. Các mô hình này được thiết kế nhằm học biểu diễn ngôn ngữ tự nhiên, từ đó có khả năng sinh văn bản, trả lời câu hỏi, tóm tắt nội dung, phân loại thông tin và thực hiện nhiều tác vụ liên quan đến xử lý ngôn ngữ tự nhiên (NLP).

Cốt lõi của LLM hiện đại là kiến trúc Transformer, được giới thiệu năm 2017. Kiến trúc này bao gồm hai thành phần chính: bộ mã hóa (encoder) và bộ giải mã (decoder), vận hành dựa trên cơ chế tự tập trung (self-attention) [6]. Cơ chế tự tập trung cho phép mô hình hiểu được mối quan hệ giữa các từ trong toàn bộ ngữ cảnh của câu hoặc đoạn văn, thay vì xử lý tuần tự như các mô hình truyền thống trước đây. Nhờ vậy, LLM có khả năng nắm bắt ngữ nghĩa, cấu trúc và sự phụ thuộc giữa các cụm từ một cách hiệu quả .

- Bộ mã hóa (Encoder): trích xuất đặc trưng ngữ nghĩa của chuỗi văn bản đầu vào, biểu diễn chúng dưới dạng vector.
- Bộ giải mã (Decoder): sử dụng thông tin đã mã hóa để sinh ra văn bản mới hoặc dự đoán token tiếp theo.

Việc huấn luyện LLM dựa trên dữ liệu đa lĩnh vực (general-domain) giúp mô hình có khả năng tổng quát hóa cao, xử lý nhiều ngữ cảnh và công việc khác nhau. Tuy nhiên, đặc điểm này cũng dẫn đến một số hạn chế. Mặc dù LLM được huấn luyện trên lượng dữ liệu khổng lồ, kiến thức trong mô hình như không quá chuyên sâu đối với các lĩnh

vực đặc thù (như y sinh, mật mã học, pháp lý chuyên sâu), có thể bị lỗi thời do mô hình phản ánh tri thức tại thời điểm huấn luyện và không tự động cập nhật theo thời gian và khó đảm bảo tinh chỉnh xác tuyệt đối nhất là đối với nhiệm vụ yêu cầu dữ liệu thời gian thực hoặc thông tin chuyên ngành mới.

Chính vì vậy, các tổ chức ngày nay thường kết hợp LLM với cơ chế bổ sung dữ liệu ngoài (như Retrieval-Augmented Generation – RAG) hoặc tinh chỉnh mô hình (fine-tuning) để tăng mức độ chính xác, cập nhật và phù hợp với từng lĩnh vực cụ thể.

1.3.2. Các mô hình ngôn ngữ lớn

Trong những năm gần đây, lĩnh vực Xử lý Ngôn ngữ Tự nhiên (NLP) đã chứng kiến sự chuyển dịch mô hình (paradigm shift) mạnh mẽ với sự ra đời của các Mô hình Ngôn ngữ Lớn (Large Language Models - LLMs). Dựa trên kiến trúc Transformer với cơ chế sự chú ý (Self-Attention mechanism), các LLM đã vượt qua giới hạn của các phương pháp thống kê truyền thống. Quy mô tham số của các mô hình này đã tăng từ hàng triệu lên hàng trăm tỷ, thậm chí hàng nghìn tỷ tham số, cho phép chúng nắm bắt các sắc thái ngữ nghĩa phức tạp, suy luận logic và tổng hợp tri thức đa lĩnh vực [6]. Hệ sinh thái LLM hiện nay rất đa dạng, được phân loại dựa trên kiến trúc (Encoder, Decoder, hoặc lai ghép) và phương pháp huấn luyện. Các mô hình LLM phổ biến hiện nay có thể kể đến như:

a) GPT (Generative Pre-trained Transformer)

GPT là dòng mô hình được phát triển bởi OpenAI, đại diện tiêu biểu cho kiến trúc Transformer chỉ sử dụng bộ giải mã (Decoder-oNLy) [7]. Cơ chế cốt lõi của GPT là dự đoán từ tiếp theo (next-token prediction) trong một chuỗi, giúp mô hình này đặc biệt xuất sắc trong các tác vụ sinh văn bản tự nhiên và sáng tạo.

Hình 6: Logo GPT

Từ phiên bản GPT-3 đến GPT-4, mô hình đã chứng minh khả năng học tập ngữ cảnh (in-context learning) và học với ít dữ liệu mẫu (few-shot learning) vượt trội. GPT-4 không chỉ xử lý văn bản mà còn mở rộng sang đa phương thức, hỗ trợ mạnh mẽ cho các tác vụ phức tạp như lập trình, suy luận logic và tóm tắt thông tin, trở thành chuẩn mực so sánh cho các mô hình sinh ngôn ngữ hiện đại [7].

b) BERT (Bidirectional Encoder Representations from Transformers)

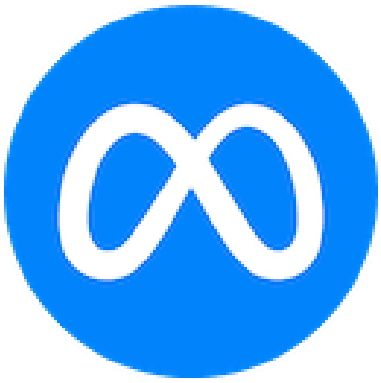
Khác với cơ chế sinh văn bản của GPT, BERT do Google phát triển tập trung vào kiến trúc Transformer chỉ sử dụng bộ mã hóa (Encoder-oNLy) và cơ chế sự chú ý hai chiều (bidirectional attention). Điều này cho phép BERT xem xét ngữ cảnh của một từ dựa trên cả các từ đứng trước và sau nó cùng một lúc.

Nhờ khả năng biểu diễn ngữ nghĩa sâu sắc, BERT trở thành nền tảng tối ưu cho các bài toán hiểu ngôn ngữ tự nhiên (Natural Language Understanding - NLU) như phân loại văn bản, nhận diện thực thể tên riêng (NER) và phân tích cảm xúc [8]. Trong lĩnh vực an toàn thông tin, các biến thể của BERT thường được ứng dụng để phân tích log hệ thống và phát hiện bất thường nhờ khả năng nắm bắt cấu trúc cú pháp chặt chẽ.

c) LLaMA (Large Language Model Meta AI)

LLaMA là dòng mô hình ngôn ngữ do Meta AI phát triển, đánh dấu bước ngoặt trong việc phổ cập mã nguồn mở các mô hình ngôn ngữ lớn. Thay vì tập trung tăng quy mô

tham số lên mức khổng lồ, LLaMA chú trọng tối ưu hóa hiệu suất hoạt động bằng cách huấn luyện trên tập dữ liệu token chất lượng cao và đa dạng hơn.



Hình 7: Logo LLaMA

Các phiên bản LLaMA (như LLaMA 2 và LLaMA 3) cung cấp hiệu năng tương đương với các mô hình thương mại đóng kín nhưng yêu cầu tài nguyên tính toán thấp hơn. Điều này cho phép cộng đồng nghiên cứu và doanh nghiệp có thể tự triển khai (self-hosted) và tinh chỉnh (fine-tune) mô hình trên hạ tầng nội bộ, giải quyết hiệu quả bài toán về bảo mật dữ liệu riêng tư.

d) Gemini

Gemini là thế hệ mô hình AI tiên tiến nhất của Google DeepMind, được thiết kế với kiến trúc đa phương thức nguyên bản (native multimodal). Khác với các mô hình trước đó thường ghép nối các thành phần xử lý hình ảnh và văn bản riêng biệt, Gemini được huấn luyện đồng thời trên dữ liệu văn bản, hình ảnh, âm thanh, video và mã nguồn ngay từ đầu.

Hình 8: Logo Gemini

Kiến trúc này mang lại cho Gemini khả năng suy luận linh hoạt và hiểu ngữ cảnh phức tạp vượt trội. Đặc biệt, Gemini thể hiện hiệu suất cao trong việc viết mã lập trình (coding) và giải quyết các bài toán logic đa bước, đóng vai trò quan trọng trong việc tự động hóa các quy trình phân tích kỹ thuật.

e) Claude

Claude là dòng mô hình ngôn ngữ được phát triển bởi Anthropic, nổi bật với triết lý "AI Hợp hiến" (Constitutional AI). Mục tiêu thiết kế của Claude là tạo ra một hệ thống AI hữu ích, trung thực và vô hại (helpful, honest, and harmless) thông qua việc giám sát quá trình huấn luyện bằng các nguyên tắc an toàn nghiêm ngặt.

Hình 9: Logo Claude

Về mặt kỹ thuật, Claude gây ấn tượng với khả năng xử lý cửa sổ ngữ cảnh (context window) cực lớn, cho phép mô hình đọc hiểu và phân tích toàn bộ các tài liệu kỹ thuật

dài, sách, hoặc các tệp log khổng lồ mà không bị mất thông tin. Điều này làm cho Claude trở thành công cụ đắc lực trong việc tra cứu và tổng hợp tri thức từ các nguồn dữ liệu lớn.

1.4. Các kỹ thuật trong mô hình ngôn ngữ lớn

1.4.1. Kỹ thuật tương tác (Prompt Engineering)

Kỹ thuật tương tác (Prompt Engineering) được định nghĩa là một phương pháp luận hệ thống nhằm thiết kế, tinh chỉnh, và tối ưu hóa đầu vào văn bản (prompt) cho các Mô hình Ngôn ngữ Lớn (LLM) [6]. Cơ sở lý thuyết của kỹ thuật này xoay quanh khái niệm học tập trong ngữ cảnh (In-context Learning - ICL), một khả năng nổi bật của các kiến trúc Transformer quy mô lớn. ICL cho phép mô hình học hỏi từ các ví dụ hoặc chỉ dẫn được nhúng trực tiếp trong ngữ cảnh đầu vào mà không cần cập nhật các tham số (trọng số) của mô hình. Do đó, Prompt Engineering là quá trình điều khiển không gian ngữ nghĩa đầu vào để kích hoạt các hành vi và khả năng lý luận đã tiềm ẩn trong mô hình thông qua quá trình huấn luyện trước (pre-training). Các loại prompt quan trọng có thể kể đến như:

- Zero-shot Prompting: Là phương thức cơ bản nhất, trong đó mô hình được yêu cầu thực hiện một nhiệm vụ chỉ dựa trên chỉ dẫn trực tiếp và kiến thức đã được huấn luyện mà không có bất kỳ ví dụ minh họa nào [9]. Zero-shot Prompting phụ thuộc trực tiếp vào mức độ phổ biến của nhiệm vụ trong dữ liệu huấn luyện và khả năng khái quát hóa của mô hình. Một prompt điển hình là: "Dịch câu này sang tiếng Anh: 'Tôi đang học kỹ thuật tương tác'."
- Few-shot Prompting: Few-shot Prompting hoạt động như một cơ chế tinh chỉnh ngữ cảnh (contextual fine-tuning) tạm thời, giúp mô hình suy luận ra quy tắc, định dạng hoặc phong cách cụ thể mà người dùng yêu cầu, từ đó cải thiện tính nhất quán và độ chính xác của kết quả. Ví dụ: "Hãy làm theo mẫu sau: Hà Nội -> Việt Nam, Tokyo -> Nhật Bản. Giờ hãy làm Berlin -> ?"
- Chain-of-Thought (CoT) Prompting: Là một kỹ thuật tiên tiến được thiết kế để giải quyết các vấn đề phức tạp đòi hỏi lý luận đa bước (multi-step reasoning).

CoT yêu cầu mô hình tạo ra một chuỗi các suy luận trung gian trước khi đưa ra câu trả lời cuối cùng. Việc này giúp mở rộng năng lực lý luận của mô hình, biến một bài toán phức tạp thành một chuỗi các thao tác đơn giản, tuần tự. Về mặt nhận thức, việc hiển thị các bước suy luận giúp mô hình duy trì "bộ nhớ làm việc" và giảm thiểu lỗi tích lũy, đặc biệt quan trọng trong các bài toán logic hoặc toán học. Khi được yêu cầu giải một bài toán về tỉ lệ hoặc phân loại logic, việc bổ sung chỉ dẫn "Hãy suy nghĩ từng bước để giải quyết vấn đề này" đã được chứng minh là có thể chuyển đổi hiệu suất của mô hình từ mức trung bình sang mức tiên tiến, nhờ vào việc khai thác khả năng lập luận từng bước vốn đã tồn tại trong kiến trúc của nó.

Kỹ thuật tương tác (Prompt Engineering) không chỉ là một thủ thuật giao tiếp mà còn là một công cụ phương pháp trong việc điều khiển LLM [9]. Khả năng tùy biến hành vi đầu ra mà không cần tái huấn luyện hoặc tinh chỉnh tham số nặng nề đã biến nó thành một thành phần không thể thiếu trong quy trình làm việc của Trí tuệ Nhân tạo hiện đại. Việc tiếp tục nghiên cứu và phát triển các phương pháp tương tác phức tạp hơn sẽ tiếp tục mở rộng giới hạn ứng dụng của các mô hình ngôn ngữ lớn.

1.4.2. Kỹ thuật mở rộng tri thức (Context Extension)

Kỹ thuật mở rộng tri thức (Context Extension) là một phương pháp luận nâng cao nhằm vượt qua giới hạn về tri thức tĩnh của các Mô hình Ngôn ngữ Lớn (LLM). Về mặt lý thuyết, tri thức của LLM bị giới hạn bởi tập dữ liệu huấn luyện và ngày cắt dữ liệu (cut-off date), dẫn đến vấn đề huyễn tưởng (hallucination) và thiếu khả năng truy cập thông tin thời gian thực [10]. Kỹ thuật mở rộng tri thức giải quyết vấn đề này bằng cách tích hợp các nguồn dữ liệu bên ngoài vào quy trình tạo sinh, trong khi vẫn duy trì tính bất biến của cấu trúc mô hình gốc. Sự mở rộng ngữ cảnh này cung cấp thông tin đã được kiểm chứng, qua đó tăng cường độ tin cậy và tính cập nhật của đầu ra. Trong kỹ thuật mở rộng tri thức có các phương pháp như:

a) RAG

Retrieval-Augmented Generation (RAG) là một phương pháp then chốt trong kỹ thuật mở rộng tri thức, kết hợp hai thành phần chính: một mô hình truy xuất thông tin (retriever) và mô hình tạo sinh (generator - LLM). Quy trình RAG bắt đầu bằng việc mô hình truy xuất sử dụng truy vấn của người dùng để tìm kiếm các đoạn thông tin liên quan từ một kho tri thức bên ngoài (knowledge base) [10]. Các đoạn văn bản này sau đó được thêm vào dưới dạng ngữ cảnh đầu vào (context) cho LLM. LLM sử dụng ngữ cảnh mở rộng này để tạo ra câu trả lời cuối cùng. Về cơ bản, RAG chuyển đổi nhiệm vụ tạo sinh thuần túy thành một nhiệm vụ tạo sinh dựa trên chứng cứ (evidence-based generation).

Khi một LLM được hỏi về một văn bản pháp luật mới được ban hành sau ngày huấn luyện của nó, hệ thống RAG sẽ truy xuất các điều khoản liên quan từ kho dữ liệu văn bản pháp luật nội bộ (ví dụ: các file PDF, cơ sở dữ liệu) và sử dụng chính xác thông tin đó để tạo ra câu trả lời, loại bỏ rủi ro huyễn tưởng về nội dung luật.

b) Function Call / Tool Use

Function Calling (Gọi hàm) hoặc Tool Use (Sử dụng Công cụ) là một phương pháp mở rộng tri thức mang tính hành động, cho phép LLM tự động gọi và thực thi các công cụ phần mềm hoặc API bên ngoài để hoàn thành một tác vụ phức tạp. Trong phương pháp này, LLM được huấn luyện để nhận diện ý định của người dùng và quyết định liệu có cần sử dụng công cụ bên ngoài hay không. Nếu có, mô hình sẽ tạo ra một cấu trúc dữ liệu chuẩn hóa (thường là định dạng JSON) chứa tên hàm và các đối số cần thiết. Sau khi công cụ thực thi và trả về kết quả, kết quả đó sẽ được đưa ngược lại vào LLM dưới dạng ngữ cảnh để mô hình tạo ra câu trả lời cuối cùng. Kỹ thuật này mở rộng khả năng của LLM từ một mô hình tạo sinh văn bản thành một tác nhân lập trình (programmable agent) có thể tương tác với thế giới thực.

Khi người dùng yêu cầu "Thời tiết ở Hồ Chí Minh ngày mai là bao nhiêu?", LLM sẽ không tự trả lời mà sẽ sinh ra lệnh gọi API: {'function_name': 'get_weather', 'parameters': {'city': 'Ho Chi Minh', 'date': 'tomorrow'}}. Kết quả trả về từ API ("30°C, nắng") được sử dụng làm cơ sở để LLM tạo ra câu trả lời bằng ngôn ngữ tự nhiên.

Kỹ thuật mở rộng tri thức, thông qua RAG và Function Calling, là một cải tiến kiến trúc ở cấp độ ứng dụng nhằm khắc phục những hạn chế cố hữu của LLM về tri thức và tính tương tác. Những phương pháp này không chỉ nâng cao đáng kể độ tin cậy (reliability) và tính xác thực (factuality) của thông tin mà mô hình cung cấp, mà còn mở rộng phạm vi tác vụ mà AI có thể thực hiện, chuyển đổi LLM thành các hệ thống AI lai (hybrid AI systems) có khả năng truy cập thông tin và thực thi hành động trong môi trường thực.

1.4.3. Kỹ thuật tinh chỉnh (Fine-tuning)

Kỹ thuật tinh chỉnh (Fine-tuning) là giai đoạn quan trọng trong vòng đời phát triển Mô hình Ngôn ngữ Lớn (LLM), tập trung vào việc cập nhật có chọn lọc các tham số (trọng số) của một mô hình đã được huấn luyện trước (pre-trained model). Khác biệt căn bản so với Kỹ thuật Tương tác (Prompt Engineering), Fine-tuning tạo ra sự thay đổi vĩnh viễn trong kiến trúc nội tại của mô hình [11]. Mục đích là để mô hình tiếp thu kiến thức chuyên sâu mới (ví dụ: dữ liệu y khoa, pháp luật) hoặc học các kỹ năng mới (ví dụ: tuân thủ định dạng phản hồi nghiêm ngặt, chuyển đổi giọng văn) mà chưa được phản ánh đầy đủ trong bộ dữ liệu huấn luyện ban đầu. Về mặt toán học, quá trình này bao gồm việc tiếp tục tối ưu hóa hàm mất mát (loss function) trên một tập dữ liệu chuyên biệt, với tốc độ học (learning rate) nhỏ hơn so với giai đoạn huấn luyện trước. Các phương pháp tinh chỉnh được sử dụng phổ biến như:

- Supervised Fine-Tuning (SFT): Là phương pháp cơ bản nhất, trong đó mô hình được tinh chỉnh trên một tập dữ liệu có giám sát bao gồm các cặp (input, target_output) chất lượng cao. SFT tận dụng tính chất của học có giám sát để dạy mô hình mối quan hệ giữa câu hỏi và câu trả lời lý tưởng trong một lĩnh vực cụ thể. Mục tiêu là để mô hình bắt chước các ví dụ này, từ đó thích nghi với một nhiệm vụ mới hoặc một bộ quy tắc giao tiếp nhất định. Trong lĩnh vực y tế, một LLM có thể được tinh chỉnh bằng các cặp dữ liệu nơi đầu vào là "triệu chứng lâm sàng" và đầu ra là "chẩn đoán bệnh tiềm năng." Quá trình này giúp mô hình không chỉ nắm bắt được từ vựng chuyên ngành mà còn học được mẫu hình lý luận

chuyên môn, cho phép nó thực hiện các tác vụ chẩn đoán sơ bộ với độ chính xác cao hơn so với mô hình cơ sở.

- Parameter-Efficient Fine-Tuning (PEFT): Là một nhóm các kỹ thuật ra đời để giải quyết các hạn chế về tài nguyên tính toán và bộ nhớ của SFT toàn bộ (Full Fine-tuning). Thay vì cập nhật hàng tỷ tham số của mô hình gốc, PEFT chỉ huấn luyện một tập hợp con nhỏ (sub-set) các tham số hoặc bổ sung các ma trận học được (learnable matrices) vào kiến trúc mô hình. Điều này giúp giảm đáng kể chi phí tính toán, thời gian huấn luyện và nhu cầu lưu trữ.
- Reinforcement Learning from Human Feedback (RLHF): Đây là giai đoạn tinh chỉnh cuối cùng, đóng vai trò then chốt trong việc căn chỉnh (align) hành vi của mô hình với các giá trị, sở thích và tiêu chuẩn an toàn của con người. Quá trình này bao gồm việc huấn luyện một mô hình phần thưởng (Reward Model) dựa trên dữ liệu phản hồi của con người (ví dụ: xếp hạng chất lượng hoặc mức độ hữu ích của các phản hồi do AI tạo ra). Mô hình phần thưởng sau đó được sử dụng để cung cấp tín hiệu phản hồi cho LLM thông qua thuật toán RL như Proximal Policy Optimization - PPO, nhằm tối đa hóa phần thưởng và tạo ra các câu trả lời được con người đánh giá cao hơn. RLHF là kỹ thuật đã tạo nên sự khác biệt của các mô hình như ChatGPT. Trong thực tế, RLHF được sử dụng để loại bỏ các câu trả lời độc hại, thiên vị hoặc không hữu ích, đồng thời khuyến khích mô hình duy trì giọng điệu an toàn, trung lập và tuân thủ các quy tắc đạo đức. Đây là cơ chế nền tảng cho căn chỉnh giá trị (value alignment) trong AI.

1.4.4. Kỹ thuật tối ưu hóa (Optimization)

Kỹ thuật Tối ưu hóa Mô hình (Model Optimization) là một tập hợp các phương pháp nhằm giảm thiểu tài nguyên tính toán (bộ nhớ và năng lượng) cần thiết để triển khai và chạy các Mô hình Ngôn ngữ Lớn (LLM), trong khi vẫn duy trì hoặc chỉ chịu sự suy giảm tối thiểu về hiệu suất (accuracy). Về mặt lý thuyết, các LLM thường chứa hàng tỷ tham số, dẫn đến yêu cầu phần cứng cao cấp (GPU chuyên dụng). Mục tiêu của tối ưu hóa là đạt được sự cân bằng giữa kích thước/tốc độ suy luận (inference speed) và chất lượng đầu ra (quality), cho phép triển khai mô hình trên các thiết bị cấu hình thấp hoặc môi

trường có độ trễ thấp (low-latency environment). Các phương pháp quan trọng trong kỹ thuật này bao gồm:

- Lượng tử hóa (Quantization): Đây là một phương pháp tối ưu hóa cốt lõi, hoạt động dựa trên việc giảm độ chính xác của các tham số và phép tính trong mạng nơ-ron. Ban đầu, các tham số của LLM thường được lưu trữ dưới dạng số dấu phẩy động 16-bit (FP16) hoặc 32-bit (FP32). Lượng tử hóa chuyển đổi các giá trị này xuống các định dạng có độ chính xác thấp hơn, phổ biến là 8-bit (INT8) hoặc 4-bit (INT4). Sự chuyển đổi này giúp giảm đáng kể dung lượng bộ nhớ (RAM/VRAM) cần thiết để lưu trữ mô hình, đồng thời tăng tốc độ tính toán do các phép toán trên số nguyên thường nhanh hơn.
- Cắt tỉa (Pruning): Là một phương pháp tối ưu hóa nhằm tạo ra một mô hình nhỏ hơn, gọn nhẹ hơn bằng cách loại bỏ các kết nối hoặc nơ-ron ít quan trọng trong mạng lưới. Dựa trên giả định rằng không phải tất cả các tham số đều đóng góp ngang nhau vào hiệu suất của mô hình. Cắt tỉa sẽ xác định và loại bỏ các trọng số có giá trị gần bằng 0 hoặc các đơn vị nơ-ron có mức độ kích hoạt thấp. Quá trình này có thể được thực hiện theo cấu trúc (cắt bỏ toàn bộ nơ-ron hoặc lớp) hoặc phi cấu trúc (cắt bỏ trọng số riêng lẻ). Mục tiêu là tạo ra một mạng lưới rối rậm (sparse) hơn, giảm số lượng phép tính mà không làm suy giảm hiệu suất đáng kể.
- Knowledge Distillation (Chưng cất Tri thức): Là một phương pháp dựa trên học tập, sử dụng mô hình lớn và phức tạp (mô hình Thầy - Teacher model) để huấn luyện một mô hình nhỏ hơn và nhanh hơn (mô hình Trò - Student model). Mô hình Thầy cung cấp "nhãn mềm" (soft labels)—là phân bố xác suất đầu ra (probability distribution) của nó—cho tập dữ liệu huấn luyện. Mô hình Trò sau đó được huấn luyện để bắt chước phân bố xác suất này, thay vì chỉ học các nhãn cứng (hard labels) truyền thống. Bằng cách học từ sự phân bố xác suất phong phú hơn của mô hình Thầy, mô hình Trò có thể hấp thụ phần lớn năng lực khái quát hóa của mô hình lớn hơn, mặc dù có kích thước tham số nhỏ hơn nhiều.

1.5. AI Agent

1.5.1. Khái niệm

Tác nhân trí tuệ nhân tạo (AI Agent) là một khuôn khổ phần mềm được thiết kế để nhận thức môi trường, đưa ra quyết định, và thực hiện hành động nhằm đạt được mục tiêu xác định [12]. Về mặt lý thuyết, AI Agent vượt ra khỏi vai trò của một mô hình tạo sinh văn bản thuần túy (như các Mô hình Ngôn ngữ Lớn - LLM) để trở thành một thực thể tự chủ (autonomous entity). Cấu trúc của một AI Agent được xây dựng dựa trên mô hình hoạt động của hệ thống nhận thức con người, tích hợp khả năng lý luận, tương tác với môi trường bên ngoài, và duy trì ngữ cảnh thông qua bộ nhớ. Sự tổng hợp này cho phép Agent thực hiện các nhiệm vụ phức tạp, đa bước một cách linh hoạt và hiệu quả hơn.

Cấu trúc chức năng của một AI Agent hiện đại thường được phân tách thành ba thành phần cốt lõi [12], mỗi thành phần đóng vai trò thiết yếu trong việc thực thi hành vi tự chủ:

- Bộ não (The Brain): Bộ não của Agent chính là các mô hình nền tảng như GPT-4 hay Llama 3. Thành phần này chịu trách nhiệm cho các chức năng nhận thức cấp cao, bao gồm lý luận (reasoning), lập kế hoạch (planning), hiểu ngữ cảnh (context comprehension) và tạo sinh ngôn ngữ (language generation). LLM tiếp nhận yêu cầu từ người dùng và các quan sát từ môi trường, sau đó xác định chuỗi hành động tối ưu cần thực hiện để đạt được mục tiêu. Khả năng Chain-of-Thought (CoT) của LLM là nền tảng cho việc lập kế hoạch đa bước của Agent.
- Công cụ (Tools): Công cụ đại diện cho khả năng hành động của Agent trong môi trường bên ngoài. Chúng được ví như "tay chân" của hệ thống, cho phép Agent vượt qua giới hạn nội tại của LLM. Các công cụ này bao gồm các chức năng kết nối bên ngoài như truy vấn API, tìm kiếm thông tin theo thời gian thực (ví dụ: Google Search), thực thi đoạn mã lập trình (ví dụ: Python Interpreter) hoặc tương tác với hệ thống file/email. Việc sử dụng công cụ cho phép Agent thu thập dữ liệu mới, cập nhật tri thức và thực hiện các tác vụ vật lý hoặc kỹ thuật số, từ đó mở rộng đáng kể phạm vi ứng dụng thực tiễn của nó.

- Bộ nhớ (Memory): Bộ nhớ là thành phần chịu trách nhiệm duy trì trạng thái (state preservation) và học hỏi từ kinh nghiệm (experiential learning). Bộ nhớ cho phép Agent lưu trữ ngữ cảnh của các tương tác trong quá khứ, các bước đã thực hiện, kết quả của các hành động, và các tri thức tạm thời cần thiết cho việc ra quyết định. Bộ nhớ thường được phân loại thành Bộ nhớ Ngắn hạn (Short-Term Memory) để lưu trữ ngữ cảnh hội thoại hiện tại, và Bộ nhớ Dài hạn (Long-Term Memory) (thường được triển khai bằng cơ sở dữ liệu vector) để lưu trữ các kinh nghiệm dưới dạng nhúng (embeddings), cho phép Agent truy xuất thông tin liên quan từ quá khứ để đưa ra các quyết định sáng suốt hơn trong tương lai.

1.5.2. CrewAI

a) Khái niệm

CrewAI là một khung công tác (framework) dựa trên ngôn ngữ lập trình Python, được thiết kế chuyên biệt để xây dựng và quản lý các Hệ thống Đa Tác nhân Trí tuệ Nhân tạo (Multi-Agent Systems - MAS). CrewAI vượt qua mô hình tương tác đơn lẻ giữa người dùng và một Mô hình Ngôn ngữ Lớn (LLM) bằng cách cho phép kiến tạo một tập hợp các tác nhân AI tự động hóa và cộng tác với nhau. Cơ sở lý thuyết của CrewAI nằm ở việc mô phỏng một cấu trúc tổ chức hoặc nhóm làm việc của con người, trong đó mỗi tác nhân được gán một vai trò và trách nhiệm chuyên biệt. Mục tiêu là phân tách một vấn đề phức tạp thành các nhiệm vụ nhỏ hơn, sau đó phân phối và điều phối việc thực thi giữa các tác nhân thông minh.

Hình 10: CrewAI

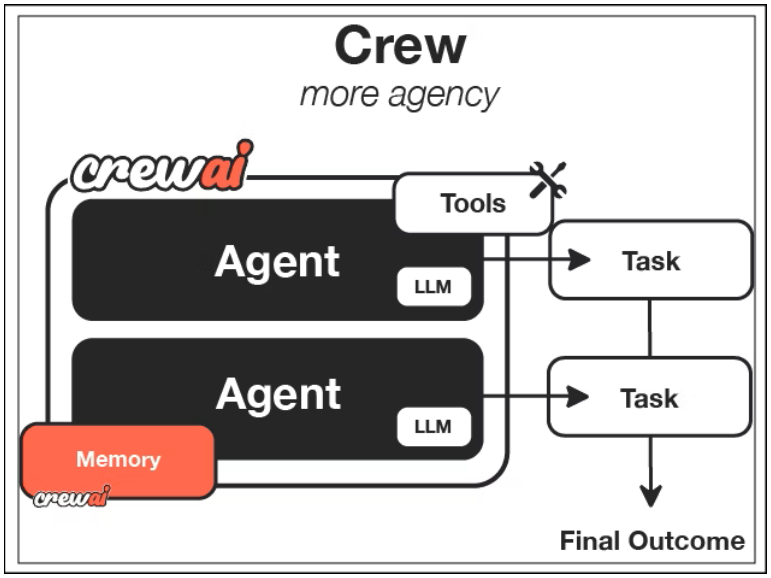
CrewAI được xây dựng trên ba khái niệm cấu trúc cốt lõi, đảm bảo sự linh hoạt và tính mô-đun hóa trong thiết kế hệ thống đa tác nhân:

- Agent (Tác nhân): Agent trong CrewAI được định nghĩa là một thực thể AI với một vai trò chuyên môn hóa (Role), một mục tiêu cụ thể (Goal) và một bản sắc (Backstory) rõ ràng. Vai trò này giúp định hình giọng điệu, phong cách lý luận và phạm vi chuyên môn của tác nhân. Quan trọng hơn, mỗi Agent có thể được trang bị Công cụ (Tools) bên ngoài (truy vấn API, tìm kiếm web,...) để thực hiện hành động trong môi trường thực.
- Task (Nhiệm vụ): Task là một đơn vị công việc cần được hoàn thành. Mỗi Task được gán cho một hoặc nhiều Agent. Task định nghĩa rõ ràng mô tả nhiệm vụ (description) và kết quả đầu ra mong muốn (expected output). Đây là đơn vị thực thi mà Agent sẽ sử dụng khả năng lý luận của mình (LLM) và các Công cụ được cấp để giải quyết.
- Crew (Đội ngũ): Crew là thực thể điều phối cấp cao, đóng vai trò là tập hợp các Agent và Task. Crew thiết lập luồng công việc (workflow), định nghĩa trình tự mà các Task sẽ được thực hiện và cách thức các Agent sẽ trao đổi thông tin và chuyển giao kết quả cho nhau. Cơ chế tự động điều phối (Automatic Orchestration) của CrewAI cho phép Crew quản lý việc giao tiếp liên tác nhân

(inter-agent communication) để đảm bảo đầu ra của Agent này trở thành đầu vào cho Agent khác một cách liền mạch.

b) Cơ chế hoạt động

Trong cơ chế hoạt động của CrewAI, tác nhân (Agent) là đơn vị thực thi cơ bản, được thiết kế để sở hữu tính chuyên biệt hóa cao. Về mặt lý thuyết, mỗi Agent được cấu hình với một tập hợp các thuộc tính định danh và khả năng chức năng. Các thuộc tính này bao gồm một vai trò (Role) rõ ràng, xác định chuyên môn của Agent (như "Nhà phân tích thị trường"); một mục tiêu (Goal), định hướng mục đích hoạt động; và một bối cảnh (Backstory), cung cấp thông tin ngữ cảnh để định hình giọng điệu và phong cách lý luận của Agent. Về mặt kỹ thuật, Agent tích hợp một mô hình ngôn ngữ lớn (LLM) làm "bộ não" cho việc lý luận và công cụ (Tools) làm khả năng tương tác ngoại vi, cho phép nó thực hiện các hành động cụ thể. Sự chuyên biệt hóa này là cần thiết để phân tách tính phức tạp của nhiệm vụ tổng thể.



Hình 11: Kiến trúc CrewAI

Nhiệm vụ (Task) đóng vai trò là đơn vị công việc có thể thực thi được trong hệ thống CrewAI. Mỗi Task được xác định bằng một mô tả nhiệm vụ chi tiết và một đầu ra mong muốn rõ ràng, nhằm thiết lập một tiêu chuẩn đánh giá thành công. Trong quy trình vận hành, mỗi Task sẽ được gán (assigned) cho một Agent cụ thể, hoặc trong một số trường

hợp, một nhóm Agent chuyên trách. Sự gán ghép chức năng này đảm bảo rằng mỗi phần của vấn đề được xử lý bởi Agent có chuyên môn phù hợp nhất, tối ưu hóa hiệu suất và chất lượng của đầu ra trung gian.

Đội ngũ (Crew) là thành phần điều phối cấp cao, chịu trách nhiệm tập hợp các Agent và Task thành một hệ thống chức năng hoàn chỉnh. Crew thiết lập và quản lý luồng công việc (workflow), quyết định trình tự và cách thức các Agent tương tác với nhau để hoàn thành nhiệm vụ chung. CrewAI hỗ trợ ba mô hình luồng công việc chính:

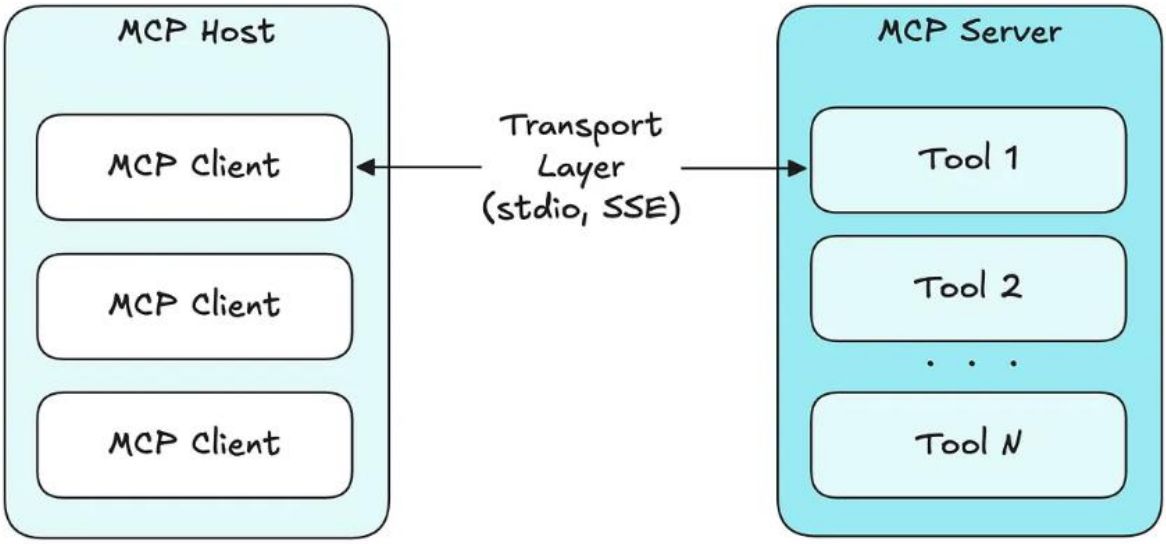
- Luồng công việc Tuần tự (Sequential Workflow): Trong mô hình này, các Task được thực thi theo một trình tự tuyến tính (linear sequence) đã được định trước. Đầu ra của Agent hoàn thành Task trước sẽ trở thành đầu vào hoặc ngữ cảnh làm việc cho Agent thực hiện Task tiếp theo. Mô hình này phù hợp với các quy trình có sự phụ thuộc rõ ràng giữa các bước, chẳng hạn như: (1) Thu thập dữ liệu, sau đó (2) Phân tích dữ liệu, và cuối cùng (3) Tổng hợp báo cáo.
- Luồng công việc Phân cấp (Hierarchical Workflow): Mô hình này giới thiệu một Tác nhân Giám sát (Supervisory Agent) đóng vai trò quản lý. Tác nhân giám sát chịu trách nhiệm tiếp nhận nhiệm vụ tổng thể, phân chia nó thành các Task nhỏ hơn, giao cho các Agent cấp dưới, và giám sát tiến độ. Tác nhân giám sát cũng đóng vai trò trọng tài khi xảy ra xung đột hoặc điều phối giao tiếp phức tạp giữa các Agent. Mô hình này hiệu quả cho các dự án lớn, phức tạp, nơi cần có một cơ chế kiểm soát và lập kế hoạch trung tâm.
- Luồng công việc Song song (Parallel Workflow): Mô hình này cho phép nhiều Agent thực hiện các Task không phụ thuộc lẫn nhau một cách đồng thời (concurrently). Luồng công việc song song được sử dụng để tối ưu hóa thời gian thực hiện bằng cách tận dụng khả năng xử lý song song của hệ thống. Sau khi các Task song song hoàn thành, kết quả thường được tổng hợp và xử lý bởi một Agent cuối cùng. Khả năng này có thể thay đổi tùy thuộc vào phiên bản và cấu hình của khung công tác CrewAI.

1.6. Model Context Protocol (MCP)

Giao thức giao tiếp mô hình khách hàng (Model-Client Protocol - MCP) là một tiêu chuẩn kỹ thuật được thiết kế để chuẩn hóa việc giao tiếp và tương tác giữa các Mô hình Ngôn ngữ Lớn (LLM) và các tài nguyên, công cụ hoặc nguồn dữ liệu bên ngoài. Về mặt lý thuyết, sự phát triển của LLM từ mô hình tạo sinh văn bản sang tác nhân hành động (AI Agent) đòi hỏi một cơ chế giao tiếp mạnh mẽ và nhất quán. MCP giải quyết nhu cầu này bằng cách thiết lập một ngôn ngữ và định dạng thông điệp chung, cho phép LLM, đóng vai trò là Client hoặc Bộ não lý luận, có thể gọi và sử dụng các khả năng của Server bên ngoài một cách hiệu quả, minh bạch và có thể mở rộng.

Mục tiêu chính của MCP là tạo ra sự tách biệt rõ ràng về trách nhiệm giữa ba thành phần cốt lõi: LLM (lý luận), Tools (hành động), và Data Sources (tri thức). Điều này giúp cho:

- Tách biệt (Decoupling): MCP cho phép phát triển và cập nhật LLM độc lập với việc phát triển các công cụ và cơ sở dữ liệu. Điều này giúp hệ thống trở nên linh hoạt và dễ bảo trì hơn, vì sự thay đổi trong mô hình không ảnh hưởng đến cách thức các công cụ bên ngoài hoạt động, và ngược lại.
- Dễ mở rộng và tích hợp: Bằng cách sử dụng một chuẩn giao thức thống nhất, các công cụ mới (ví dụ: một API dự báo thời tiết mới, một hệ thống quản lý cơ sở dữ liệu mới) có thể được tích hợp vào hệ thống AI Agent mà không cần lập trình lại cơ chế gọi hàm của LLM.
- Chuẩn hóa hành vi: MCP chuẩn hóa cách thức LLM đưa ra các quyết định hành động trong môi trường thực tế, từ đó hỗ trợ việc kiểm soát, giám sát và đảm bảo an toàn cho các tác vụ của AI Agent.



Hình 12: Kiến trúc MCP

Kiến trúc cốt lõi của Giao thức Giao tiếp Mô hình Khách hàng (MCP) được xây dựng dựa trên mô hình Client – Server tiêu chuẩn. Trong kiến trúc này, các ứng dụng sử dụng Mô hình Ngôn ngữ Lớn (LLM) như ChatGPT hoặc các tiện ích mở rộng của trình soạn thảo (editor plugins) đóng vai trò là Client. Client chịu trách nhiệm lý luận và xác định nhu cầu hành động hoặc truy xuất dữ liệu. Ngược lại, MCP Server đóng vai trò là nhà cung cấp dịch vụ tập trung, quản lý và điều phối quyền truy cập vào các tài nguyên bên ngoài.

Chương 2: Các nghiên cứu liên quan

2.1. Một số công trình nghiên cứu trong nước

Các nghiên cứu trong nước thời gian gần đây tập trung vào việc ứng dụng các mô hình tiên tiến như deep learning và xử lý ngôn ngữ tự nhiên (NLP) nhằm giải quyết những thách thức đặc thù trong quản lý sự kiện và thông tin an ninh (SIEM) tại Việt Nam. Việc phân tích các công trình này giúp hình thành cơ sở kỹ thuật và bối cảnh ngôn ngữ quan trọng cho quá trình phát triển các ai agent có khả năng hỗ trợ hiệu quả các truy vấn SIEM.

Công trình "Nghiên cứu ứng dụng deep learning trong phân tích log hệ thống để phát hiện tấn công mạng" [13](2023) tập trung vào xử lý khối lượng lớn dữ liệu log phi cấu trúc bằng các mô hình học sâu. Phương pháp chính sử dụng mạng nơ-ron hồi quy (RNN) và mạng nơ-ron tích chập (CNN) nhằm trích xuất đặc trưng tuần tự và cục bộ từ các dòng log. Các đặc trưng vector hóa được tạo ra sau quá trình tiền xử lý đóng vai trò làm đầu vào cho mô hình phân loại hoặc phát hiện điểm bất thường, từ đó xác định những sự kiện có dấu hiệu tấn công. Kết quả thực nghiệm cho thấy mô hình deep learning đạt độ chính xác cao hơn đáng kể so với phương pháp thống kê truyền thống nhờ khả năng nắm bắt các mẫu ẩn trong ngữ cảnh log. Mặc dù có ưu thế trong việc học các quan hệ phức tạp, mô hình vẫn gặp hạn chế liên quan đến chi phí tính toán lớn và khả năng giải thích còn hạn chế, gây khó khăn khi tích hợp vào cơ chế lý giải của ai agent.

Nghiên cứu "Xây dựng mô hình xử lý ngôn ngữ tự nhiên tiếng Việt cho truy vấn dữ liệu an ninh mạng" của Phạm Văn Thắng (2024) [14]giải quyết trực tiếp thách thức về ngôn ngữ trong tương tác với SIEM. Phương pháp được áp dụng là xây dựng bộ dữ liệu song ngữ (tiếng Việt – truy vấn SIEM) và triển khai kiến trúc sequence-to-sequence (seq2seq) dựa trên transformer. Mục tiêu của mô hình là ánh xạ các thuật ngữ và cấu trúc ngữ nghĩa tiếng Việt trong lĩnh vực an ninh mạng sang các toán tử và cú pháp logic của ngôn ngữ truy vấn SIEM, đóng vai trò như một bộ chuyển dịch tự động. Kết quả đánh giá cho thấy mô hình có khả năng diễn giải và chuyển đổi chính xác các truy vấn tiếng Việt phức tạp, góp phần nâng cao khả năng "hỗ trợ truy vấn SIEM" bằng tiếng mẹ đẻ.

Tuy nhiên, việc thích ứng với thay đổi trong ngôn ngữ chuyên ngành và đảm bảo an toàn dữ liệu huấn luyện vẫn là những thách thức cần được xem xét.

Công trình "Thiết kế và triển khai hệ thống tự động hóa phản ứng và báo cáo sự kiện an ninh mạng (soar) sử dụng học máy" của Ngô Quang Lợi (2022) [15] cung cấp nền tảng quan trọng cho giai đoạn đầu ra của hệ thống ai agent. Phương pháp chủ đạo là tích hợp các module học máy vào quy trình phản ứng và báo cáo sự kiện, trong đó các thuật toán phân loại và tóm tắt được sử dụng để tạo cấu trúc cho thông tin thu được từ truy vấn SIEM. Kiến trúc được triển khai theo hướng tự động hóa quy trình, cho phép hệ thống tự đưa ra quyết định phản ứng và tổng hợp báo cáo. Kết quả thực nghiệm chứng minh khả năng giảm thiểu đáng kể thao tác thủ công và rút ngắn thời gian xử lý sự cố. Ưu điểm nổi bật là khả năng sinh báo cáo tự động theo yêu cầu, đáp ứng tốt các tiêu chí về tính kịp thời và nhất quán. Tuy nhiên, hệ thống vẫn gặp hạn chế trong việc xử lý các sự kiện không có tiền lệ hoặc các yêu cầu báo cáo phi chuẩn, khiến việc điều chỉnh thủ công trở nên cần thiết.

2.2. Một số công trình nghiên cứu ngoài nước

Trong lĩnh vực ứng dụng các mô hình ngôn ngữ lớn (LLM) và tác nhân thông minh (AI Agent) vào quản lý sự kiện và thông tin an ninh (SIEM), các công trình nghiên cứu gần đây đã đặt nền tảng quan trọng cho việc tự động hóa phân tích và nâng cao hiệu quả vận hành an ninh. Các nghiên cứu quốc tế tiêu biểu có thể được phân thành ba hướng chính: khảo sát tổng quan vai trò của LLM trong an ninh mạng, thiết kế kiến trúc agent cho vận hành tự động, và phát triển giao diện truy vấn ngôn ngữ tự nhiên dành cho hệ thống SIEM.

Công trình "Cybersecurity with large language models: A survey" của Li et al. (2024) [16] đưa ra một phân tích toàn diện về cách thức LLM được tích hợp vào nhiều tác vụ an ninh mạng khác nhau. Phương pháp chính của nghiên cứu là tổng hợp, phân loại và hệ thống hóa các ứng dụng LLM theo ba nhóm chức năng cốt lõi: phát hiện, phòng chống và điều tra. Các tác giả nhấn mạnh khả năng xử lý ngôn ngữ tự nhiên (NLP) của LLM trong việc hiểu ngữ cảnh, tóm tắt sự kiện, suy diễn quan hệ, cũng như sinh mã và câu

lệnh hỗ trợ điều tra. Kiến trúc agent được đề cập đặt LLM vào vai trò bộ điều khiển trung tâm (controller), có khả năng thực thi chuỗi hành động như đọc log, phân tích cảnh báo và sinh truy vấn tự động. Kết quả tổng hợp cho thấy LLM đặc biệt hiệu quả đối với dữ liệu phi cấu trúc quy mô lớn. Điểm mạnh nổi bật là giảm tải đáng kể khối lượng phân tích thủ công. Tuy vậy, nghiên cứu cũng chỉ ra rủi ro từ hiện tượng sinh ảo (hallucination), đặc biệt khi LLM tạo các câu lệnh đặc thù như KQL hoặc SQL, đòi hỏi cơ chế xác thực và kiểm tra chặt chẽ trước khi triển khai.

Nghiên cứu "Autonomous security operations with AI Agents" của K. et al. (2023) [17] tập trung vào việc thiết kế các kiến trúc agent nhằm tự động hóa toàn diện chu trình vận hành an ninh (soc). Phương pháp tiếp cận dựa trên mô hình tác tử–môi trường (agent–environment model), trong đó agent tương tác với các công cụ soc như SIEM hoặc soar thông qua các hành động được điều khiển bởi LLM. Cấu trúc mô hình gồm ba tầng: lớp nhận thức (perception layer) thu thập dữ liệu SIEM, lớp lập kế hoạch (planning layer) dùng LLM để xây dựng chuỗi hành động, và lớp thực thi (execution layer) đảm trách việc gửi truy vấn và tạo báo cáo. Nghiên cứu cho thấy khả năng tự động hóa các tác vụ lặp lại, tối ưu thời gian phản ứng và hỗ trợ sinh báo cáo theo yêu cầu. Ưu điểm đáng chú ý là khả năng ra quyết định tự trị trong môi trường động. Tuy nhiên, công trình cũng chỉ ra thách thức liên quan đến quản lý trạng thái (state management), kiểm soát hành động tự trị và đảm bảo an toàn khi triển khai trong môi trường sản xuất thực tế.

Công trình "Natural language interface for security information and event management systems" của K. et al. (2023) [18] tập trung giải quyết vấn đề tương tác người–máy trong ngữ cảnh SIEM. Phương pháp chính là chuyển đổi ngôn ngữ tự nhiên (NL) sang ngôn ngữ truy vấn có cấu trúc (SQL hoặc các biến thể truy vấn SIEM). Nghiên cứu sử dụng mô hình học sâu kết hợp cơ chế chú ý (attention mechanism) để ánh xạ các khái niệm an ninh, ràng buộc logic và ngữ cảnh câu hỏi thành cú pháp truy vấn chính xác. Kiến trúc Seq2Seq được tinh chỉnh trên tập dữ liệu NL–query nhằm cải thiện năng lực suy diễn. Kết quả thực nghiệm cho thấy độ chính xác vượt trội so với phương pháp dựa trên từ khóa, đồng thời mở rộng khả năng tiếp cận hệ thống SIEM cho các nhà phân tích không chuyên về truy vấn. Tuy nhiên, nghiên cứu cũng ghi nhận hạn chế về khả

năng mở rộng và tính thích ứng khi schema SIEM hoặc ngôn ngữ truy vấn thay đổi, đòi hỏi chi phí tái huấn luyện đáng kể.

Nhìn chung, các công trình trên cho thấy xu hướng hội tụ giữa LLM, AI Agent và SIEM nhằm tạo ra thế hệ hệ thống vận hành an ninh tự động, linh hoạt và thông minh hơn. Các kết quả nghiên cứu không chỉ cung cấp cơ sở lý thuyết vững chắc mà còn định hướng rõ ràng cho các hệ thống SIEM thế hệ mới, nơi tương tác ngôn ngữ tự nhiên, tự động hóa tác vụ và khả năng suy luận của LLM trở thành trung tâm của quá trình phân tích và ra quyết định.

Chương 3: Xây dựng hệ thống AI AGENT hỗ trợ truy vấn SIEM và xuất kết quả theo cầu

3.1. Yêu cầu hệ thống

a) Yêu cầu chức năng

Hệ thống được thiết kế nhằm cung cấp một cơ chế tương tác cho phép người dùng diễn đạt yêu cầu trực tiếp bằng ngôn ngữ tự nhiên. Cách tiếp cận này giúp giảm đáng kể rào cản kỹ thuật trong quá trình khai thác dữ liệu an ninh, cho phép người dùng tập trung vào mục tiêu phân tích thay vì phải nắm vững cú pháp hoặc ngôn ngữ truy vấn chuyên biệt của từng nền tảng SIEM. Nhờ đó, hệ thống mở rộng khả năng tiếp cận cho nhiều nhóm người dùng với trình độ kỹ thuật khác nhau.

Trên cơ sở đó, hệ thống phải có khả năng phân tích ngữ nghĩa và cấu trúc của các câu lệnh ngôn ngữ tự nhiên, từ đó chuyển đổi chúng thành một cấu trúc truy vấn chung ở dạng chuẩn hóa. Cấu trúc truy vấn này đóng vai trò như một lớp trung gian trừu tượng, độc lập với các nền tảng SIEM cụ thể, giúp tách biệt giai đoạn diễn giải yêu cầu người dùng khỏi giai đoạn thực thi truy vấn. Việc tiêu chuẩn hóa này không chỉ nâng cao tính nhất quán trong xử lý truy vấn mà còn tạo nền tảng cho khả năng mở rộng hệ thống.

Bên cạnh khả năng chuẩn hóa, hệ thống cần hỗ trợ nhiều nền tảng SIEM khác nhau, tối thiểu bao gồm Splunk và Elasticsearch. Để đáp ứng yêu cầu này, hệ thống phải thực hiện quá trình ánh xạ và dịch cấu trúc truy vấn chung sang ngôn ngữ truy vấn đặc thù của từng nền tảng. Cơ chế dịch này bảo đảm rằng cùng một yêu cầu ở mức trừu tượng có thể được thực thi chính xác trên các hệ thống SIEM khác nhau, đồng thời duy trì tính tương thích và hiệu quả trong vận hành.

Ngoài chức năng sinh truy vấn, hệ thống còn phải cung cấp khả năng diễn giải lại câu lệnh truy vấn đã được tạo ra dưới dạng ngôn ngữ tự nhiên. Chức năng này cho phép người dùng hiểu rõ mục đích phân tích, phạm vi dữ liệu được truy vấn cũng như logic xử lý được áp dụng. Việc giải thích truy vấn trước khi thực thi góp phần nâng cao tính minh bạch, giảm thiểu rủi ro sai lệch yêu cầu và tăng mức độ tin cậy của hệ thống.

Cuối cùng, sau khi truy vấn được thực thi trên các nền tảng SIEM, hệ thống cần thực hiện việc tổng hợp, tóm tắt và trình bày kết quả dưới dạng báo cáo. Quá trình này nhằm chuyển đổi dữ liệu thô thu được thành thông tin có ý nghĩa, giúp người dùng nhanh chóng nắm bắt các điểm quan trọng và hỗ trợ hiệu quả cho hoạt động phân tích, giám sát và ra quyết định trong lĩnh vực an ninh thông tin.

b) Yêu cầu phi chức năng

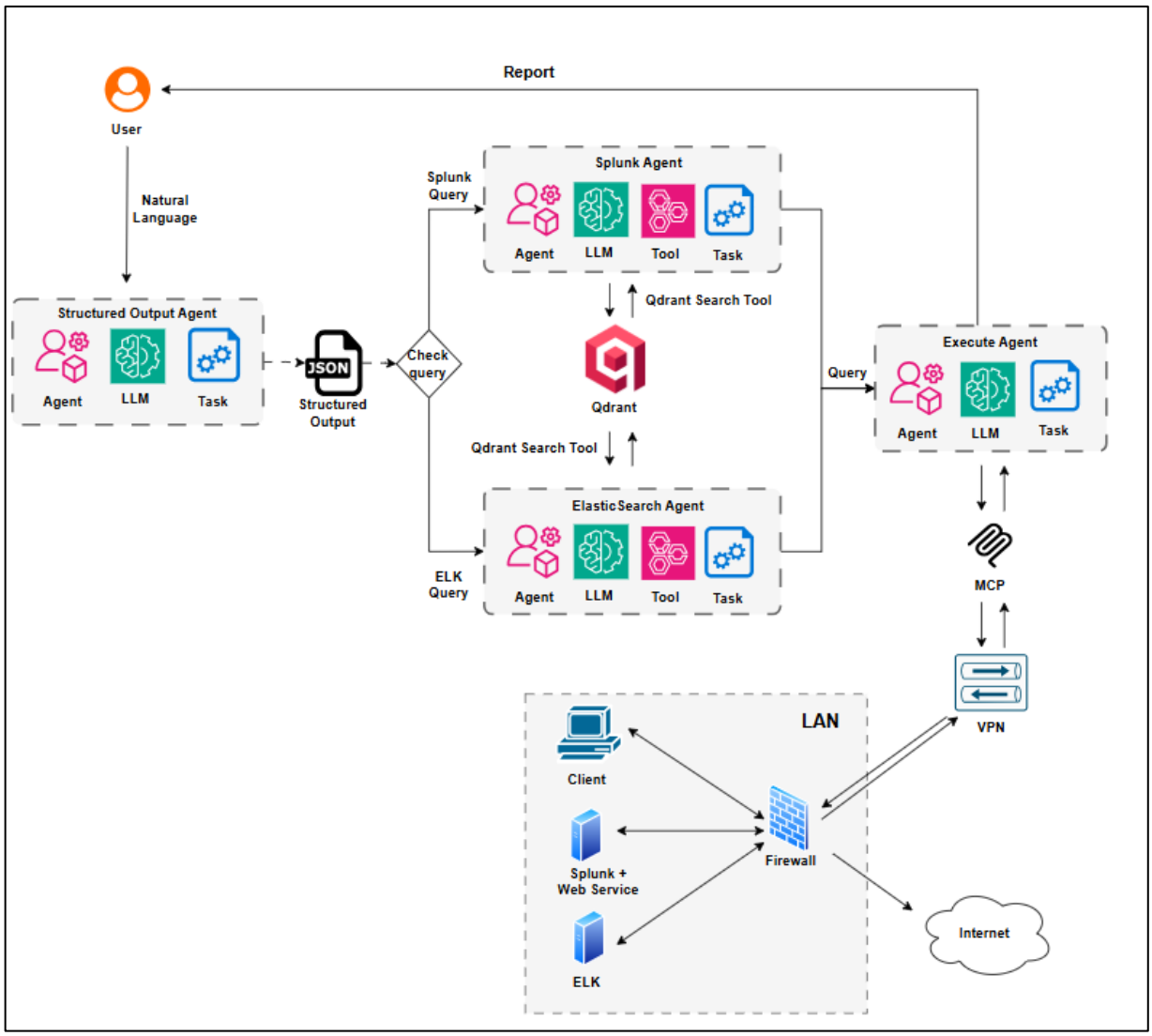
Bên cạnh các yêu cầu về mặt chức năng, hệ thống còn phải đáp ứng một tập hợp các yêu cầu phi chức năng nhằm bảo đảm chất lượng, hiệu quả vận hành và khả năng phát triển bền vững trong môi trường triển khai thực tế. Các yêu cầu bao gồm:

- Về tính chính xác (Accuracy), hệ thống phải bảo đảm rằng các câu lệnh truy vấn được sinh ra phản ánh đúng yêu cầu của người dùng cả về mặt cú pháp lẫn ngữ nghĩa. Điều này đòi hỏi quá trình chuyển đổi từ ngôn ngữ tự nhiên sang truy vấn phải duy trì được ý nghĩa ban đầu, tránh các sai lệch có thể ảnh hưởng đến kết quả phân tích, qua đó bảo đảm độ tin cậy tổng thể của hệ thống.
- Đối với hiệu năng (Performance), hệ thống cần đáp ứng yêu cầu về thời gian phản hồi ngắn trong toàn bộ quá trình xử lý, từ khi tiếp nhận yêu cầu của người dùng cho đến khi tạo ra câu lệnh truy vấn tương ứng. Việc duy trì hiệu năng ổn định và nhanh chóng giúp hạn chế độ trễ không cần thiết, đồng thời nâng cao trải nghiệm sử dụng trong các kịch bản phân tích và giám sát an ninh thời gian gần thực.
- Yêu cầu về tính dễ sử dụng (Usability) tập trung vào việc thiết kế giao diện người dùng theo hướng đơn giản và trực quan. Giao diện cần hỗ trợ người dùng thực hiện các thao tác một cách thuận tiện, rõ ràng và nhất quán, cho phép khai thác hiệu quả các chức năng của hệ thống mà không đòi hỏi quá trình đào tạo phức tạp hay kiến thức kỹ thuật chuyên sâu.
- Cuối cùng, về tính mở rộng (Scalability), kiến trúc hệ thống phải được thiết kế theo hướng linh hoạt, cho phép tích hợp và hỗ trợ thêm các nền tảng SIEM mới trong tương lai. Việc mở rộng này cần được thực hiện mà không yêu cầu thay đổi

đáng kể đối với logic xử lý cốt lõi, từ đó bảo đảm khả năng phát triển lâu dài và thích ứng với sự đa dạng của các hệ thống giám sát an ninh.

3.2. Kiến trúc hệ thống

Kiến trúc tổng thể của hệ thống được thiết kế theo mô hình nhiều tác tử (multi-agent), trong đó mỗi thành phần đảm nhiệm một vai trò độc lập nhưng phối hợp chặt chẽ nhằm bảo đảm quá trình chuyển đổi ngôn ngữ tự nhiên thành truy vấn SIEM diễn ra chính xác, linh hoạt và có thể mở rộng. Dòng xử lý được tổ chức thành chuỗi các bước liên tục, từ tiếp nhận yêu cầu của người dùng, phân tích và chuẩn hóa truy vấn, lựa chọn nền tảng SIEM phù hợp, cho đến thực thi và trả về báo cáo kết quả.



Hình 13: Kiến trúc hệ thống

Ở giai đoạn đầu, người dùng gửi yêu cầu bằng ngôn ngữ tự nhiên. Yêu cầu này được chuyển đến Structured Output Agent, nơi một mô hình ngôn ngữ lớn (LLM) chịu trách nhiệm diễn giải và chuyển đổi đầu vào thành cấu trúc truy vấn chuẩn ở định dạng JSON. Định dạng này đóng vai trò lớp trung gian, bảo đảm tính độc lập với từng nền tảng SIEM cụ thể.

Cấu trúc truy vấn tạo ra sau đó được chuyển đến bộ phận kiểm tra truy vấn (Check Query). Tại đây, hệ thống xác định loại truy vấn cần thực thi và lựa chọn tác tử SIEM phù hợp. Dựa trên kết quả này, truy vấn sẽ được chuyển đến SQLunk Agent hoặc ElasticSearch Agent tương ứng. Mỗi tác tử bao gồm các thành phần LLM, công cụ hỗ trợ (Tool) và tác vụ (Task), cho phép dịch cấu trúc truy vấn chuẩn sang cú pháp truy vấn gốc của từng hệ thống. Đồng thời, việc tìm kiếm các mẫu truy vấn hoặc tham chiếu được hỗ trợ bởi cơ sở dữ liệu vector thông qua Qdrant Search Tool và hệ thống Qdrant.

Sau khi truy vấn được chuyển đổi hoàn chỉnh, hệ thống đưa truy vấn cuối cùng đến Execute Agent. Tác tử này chịu trách nhiệm thực thi truy vấn thông qua MCP (Model Context Protocol) và đường truyền bảo mật VPN kết nối tới hạ tầng nội bộ của doanh nghiệp. Trong hạ tầng LAN, các dịch vụ SQLunk, Elasticsearch (ELK) và các web service liên quan được triển khai sau tường lửa nhằm bảo đảm an toàn thông tin.

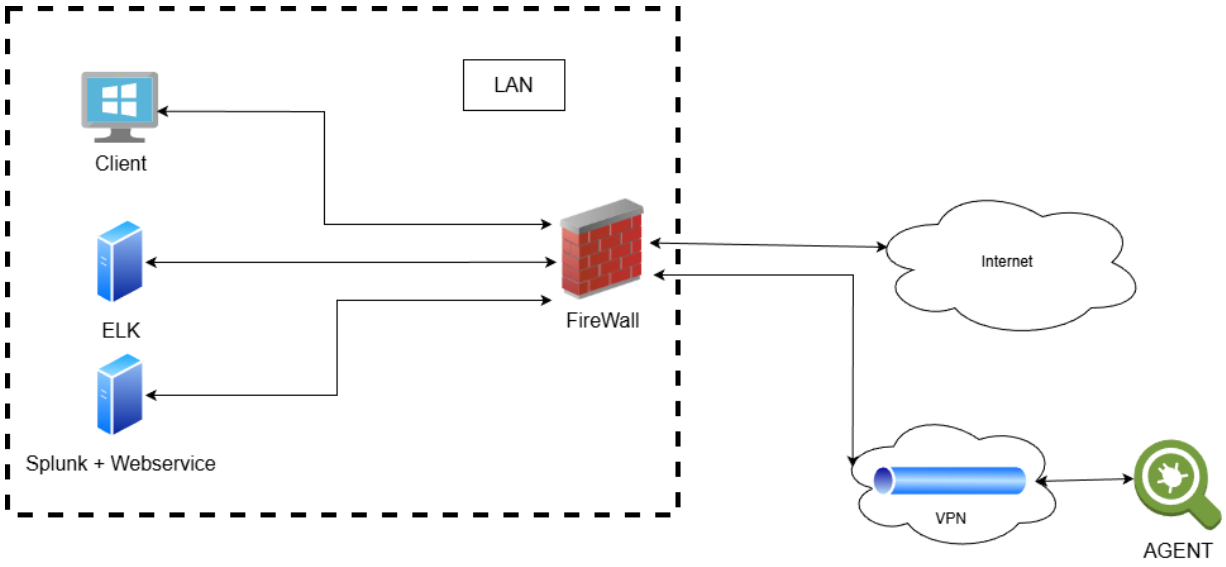
Kết quả truy vấn thu được từ các hệ thống SIEM được gửi ngược lại cho tác tử thực thi để phân tích và tóm tắt. Báo cáo cuối cùng sau đó được truyền trở lại cho người dùng dưới dạng ngôn ngữ tự nhiên. Cách tổ chức này cho phép hệ thống cung cấp khả năng giải thích, tổng hợp và báo cáo một cách hiệu quả và dễ hiểu.

Trong kiến trúc tách biệt rõ ràng giữa các giai đoạn xử lý: phân tích ngôn ngữ tự nhiên, chuẩn hóa truy vấn, dịch sang truy vấn SIEM, thực thi trong môi trường cách ly, và tạo báo cáo. Việc ứng dụng mô hình tác tử giúp tăng tính linh hoạt, khả năng mở rộng cũng như tạo điều kiện tích hợp thêm các hệ thống SIEM khác trong tương lai mà không cần thay đổi cấu trúc cốt lõi của hệ thống.

3.3. Triển khai

3.3.1 Xây dựng hệ thống SIEM

Hệ thống SIEM được triển khai trong môi trường mạng cục bộ (LAN) và được bảo vệ bởi một cổng kết nối trung tâm đóng vai trò là tường lửa. Kiến trúc này nhằm bảo đảm sự cách ly giữa mạng nội bộ và môi trường bên ngoài, đồng thời kiểm soát chặt chẽ luồng lưu lượng ra vào hệ thống.



Hình 14: Hệ thống SIEM

Bên trong mạng LAN, hai nền tảng SIEM được triển khai song song, bao gồm hệ thống ELK (ElasticSearch, Logstash và Kibana) và Splunk. Việc triển khai đồng thời nhiều hệ thống SIEM cho phép thu thập, lưu trữ và phân tích dữ liệu nhật ký từ các nguồn khác nhau, đồng thời tạo điều kiện so sánh và đánh giá khả năng xử lý truy vấn trên các nền tảng SIEM khác nhau.

Các máy trạm trong mạng LAN, được cấu hình dưới dạng các máy chủ hoặc máy khách sử dụng hệ điều hành Windows 10, đóng vai trò là nguồn sinh dữ liệu nhật ký. Các log phát sinh từ các máy này được thu thập và gửi trực tiếp đến cả hai hệ thống SIEM, phục vụ cho mục đích giám sát, phân tích và kiểm thử chức năng của hệ thống.

Tường lửa được triển khai sử dụng nền tảng pfSense, chịu trách nhiệm giám sát và ghi nhận nhật ký lưu lượng mạng giữa môi trường Internet bên ngoài và mạng LAN nội

bộ. Thông qua chức năng này, hệ thống có thể theo dõi và phân tích các hoạt động truy cập, qua đó hỗ trợ phát hiện các hành vi bất thường hoặc tiềm ẩn rủi ro an ninh.

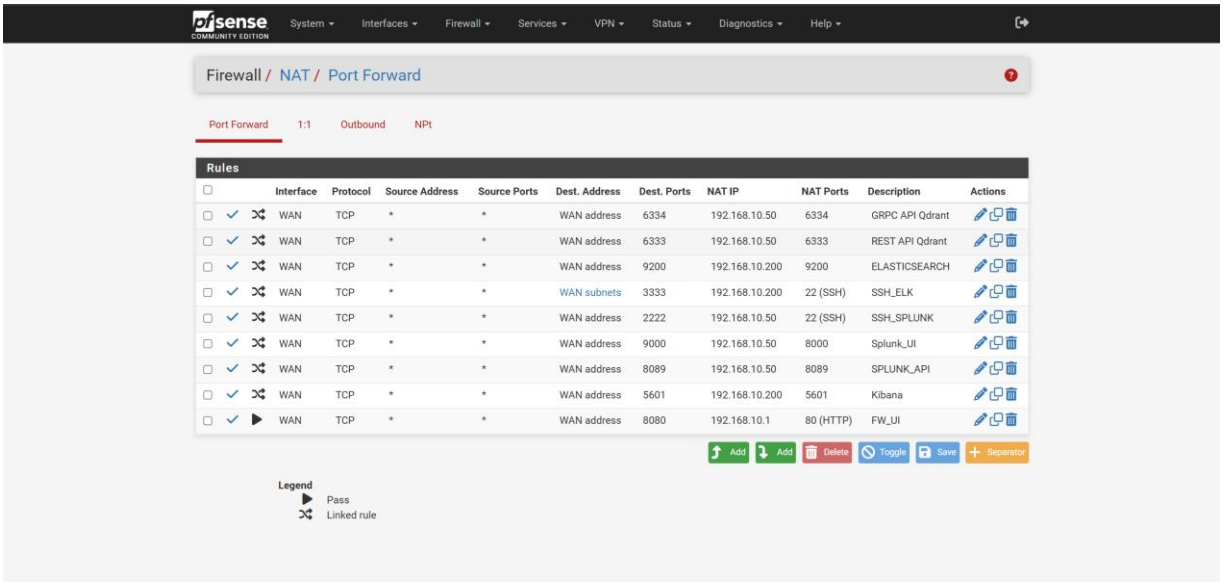
Ngoài chức năng ghi log, tường lửa còn đảm nhiệm vai trò chuyển tiếp cổng (port forwarding), cho phép các dịch vụ bên trong mạng LAN được truy cập gián tiếp từ bên ngoài. Cơ chế này giúp che giấu địa chỉ IP thực của các máy nội bộ, qua đó góp phần nâng cao mức độ an toàn và giảm thiểu nguy cơ bị tấn công trực tiếp từ Internet.

Bảng 1: Địa chỉ IP

Host	Địa chỉ IP
Client	192.168.10.150/24
ELK	192.168.10.200/24
SQLUNK	192.168.10.50/24
FIREWALL	192.168.111.162/24(WAN) 192.168.10.1(LAN)

a) Cài đặt Firewall

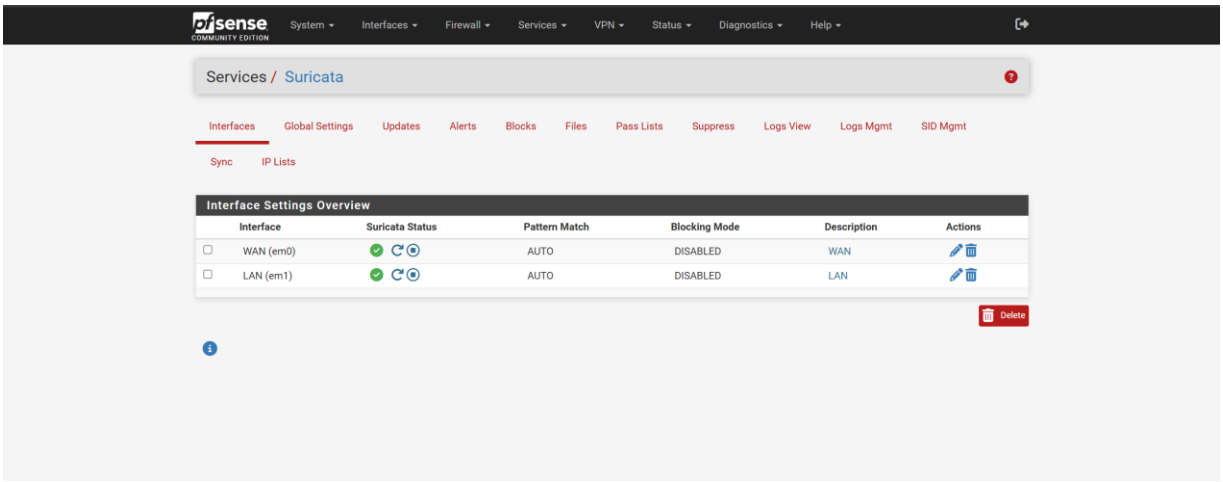
Cấu hình port forward trên firewall từ LAN ←→ WAN, để máy host có thể truy cập các service có trong LAN.



Hình 15: Cấu hình port forward trong firewall

Tích hợp IDS/IPS suricata trên firewall để giám sát lưu lượng mạng và tạo cảnh báo trên firewall.

Mặc định sẽ tắt chức năng block của suricata nhằm tránh việc block IP của Host truy cập vào bên trong LAN. Khi truy cập từ bên ngoài vào LAN cấu hình các service như SQLunk hay ELK cũng có thể dẫn đến suricata block IP đó.

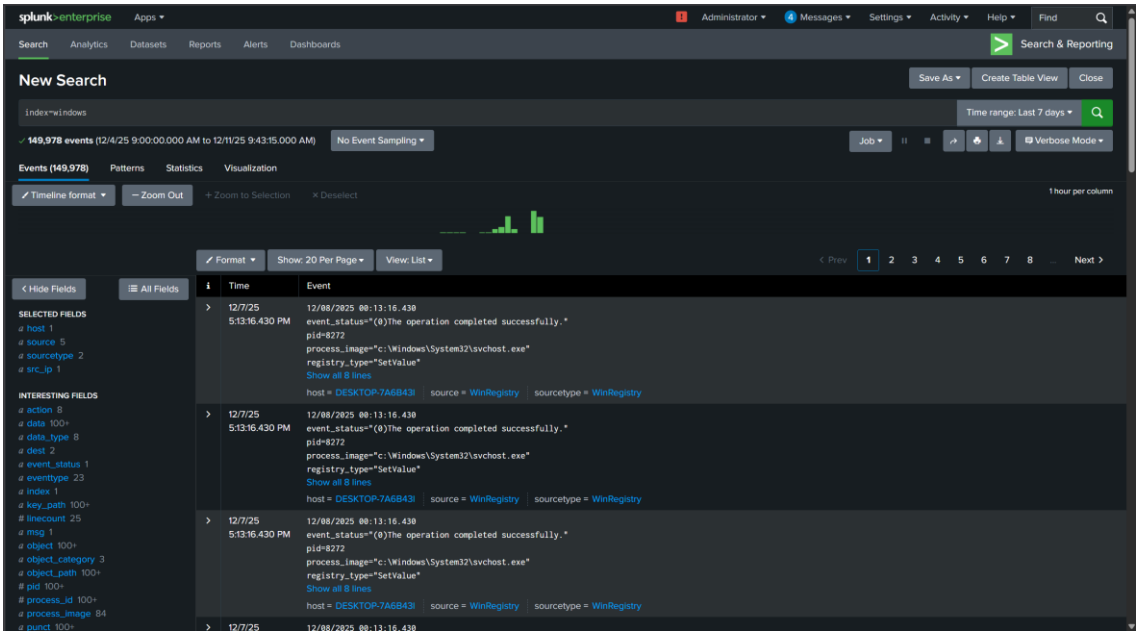


Hình 16: Cài đặt interfaces

b) Cấu hình SQLUNK.

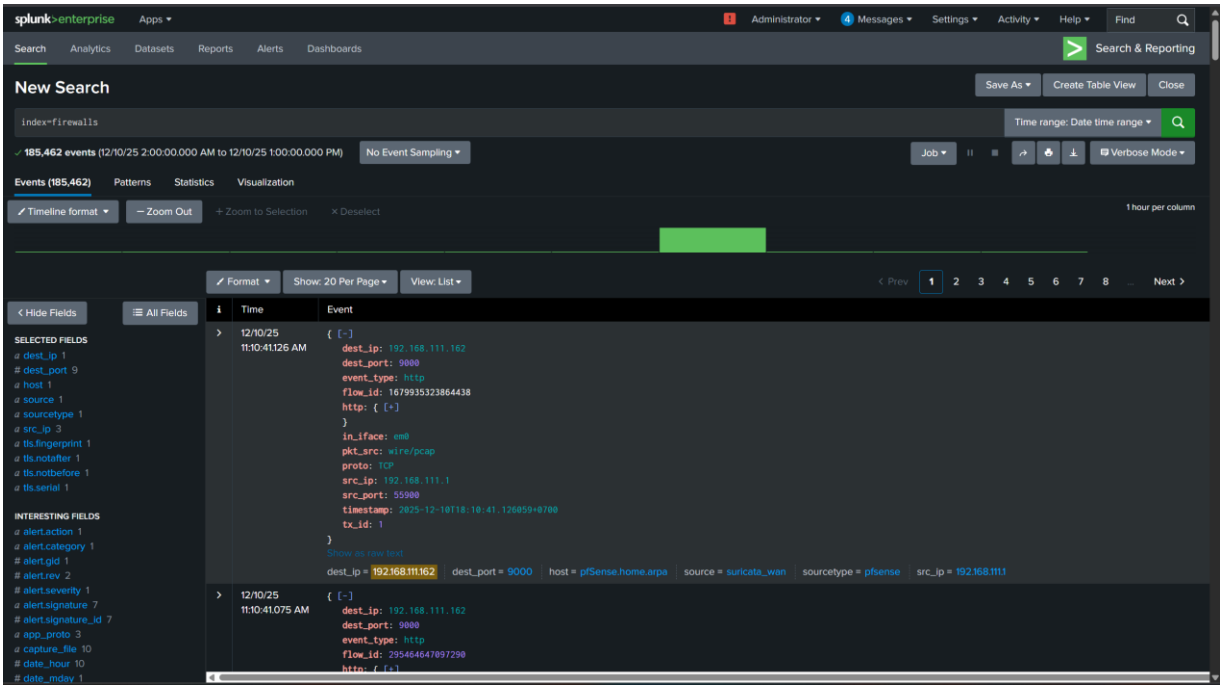
Trên SQLunk cấu hình 2 index: windows và firewall để nhận log từ client vàn firewall pfsense.

Trên index windows, client sẽ gửi các log OS như Application, Security, Powershell, sysmon,..



Hình 17: Log gửi về Splunk

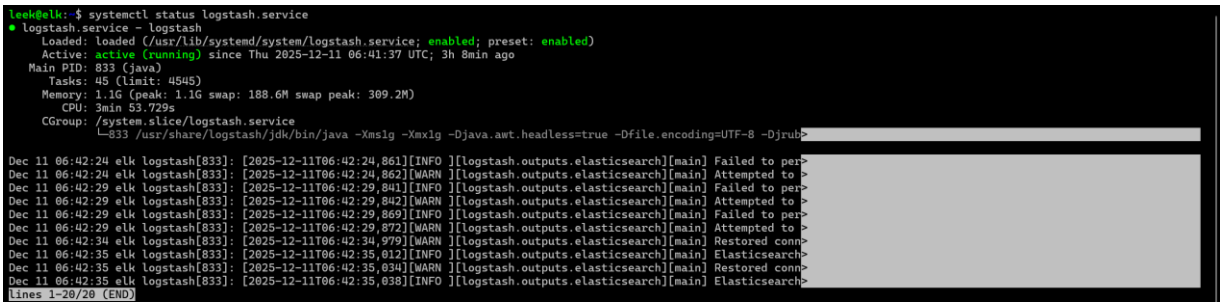
Trên index firewall, pfsense sẽ gửi log trên 2 interface WAN và LAN nhằm thu thập đủ thông tin lưu lượng mạng.



Hình 18: Cấu hình Splunk

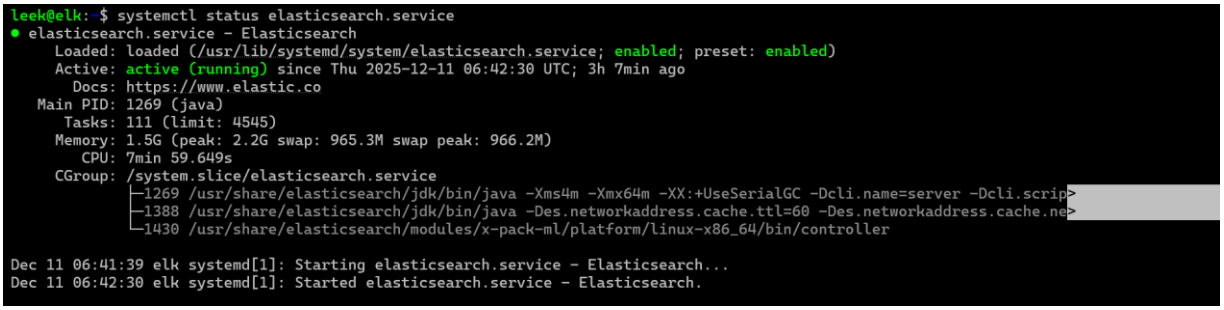
c) Cấu hình ELK

Cài đặt Logstath, “trung gian” nhận log và index cho các log trước khi forward vào elasticsearch.



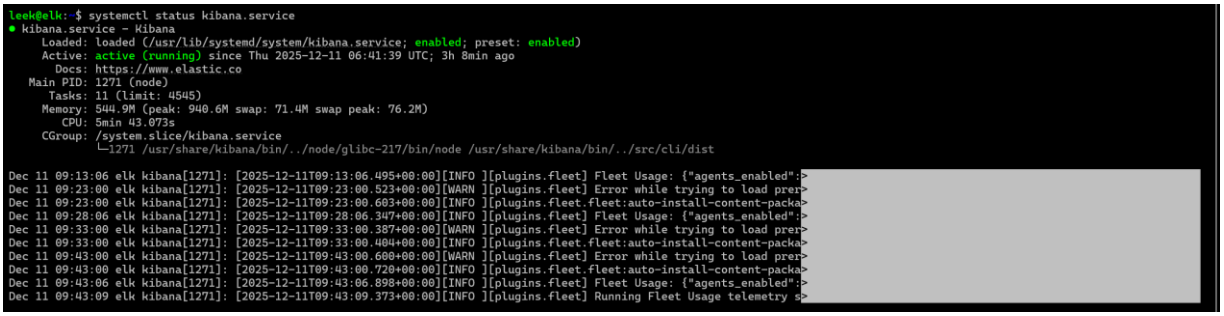
Hình 19: Cài đặt Logstath

Elasticsearch, bộ phần chính trong hệ thống nơi lưu trữ, truy vấn các thông tin index, log.



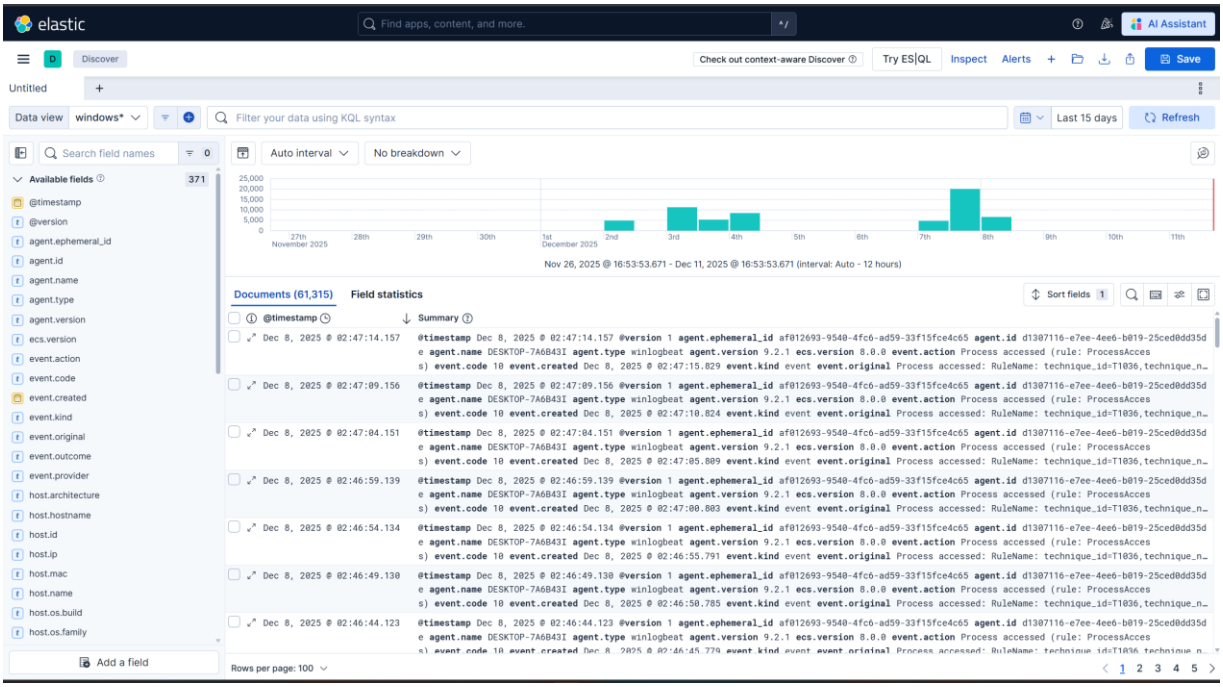
Hình 20: Các service trong Elasticsearch

Kibana, nơi để xem các log từ elasticsearch một các trự quan hơn, tìm kiếm đơn giản với giao diện web.



Hình 21: Dịch vụ kibana

Giao diện trên Kibana

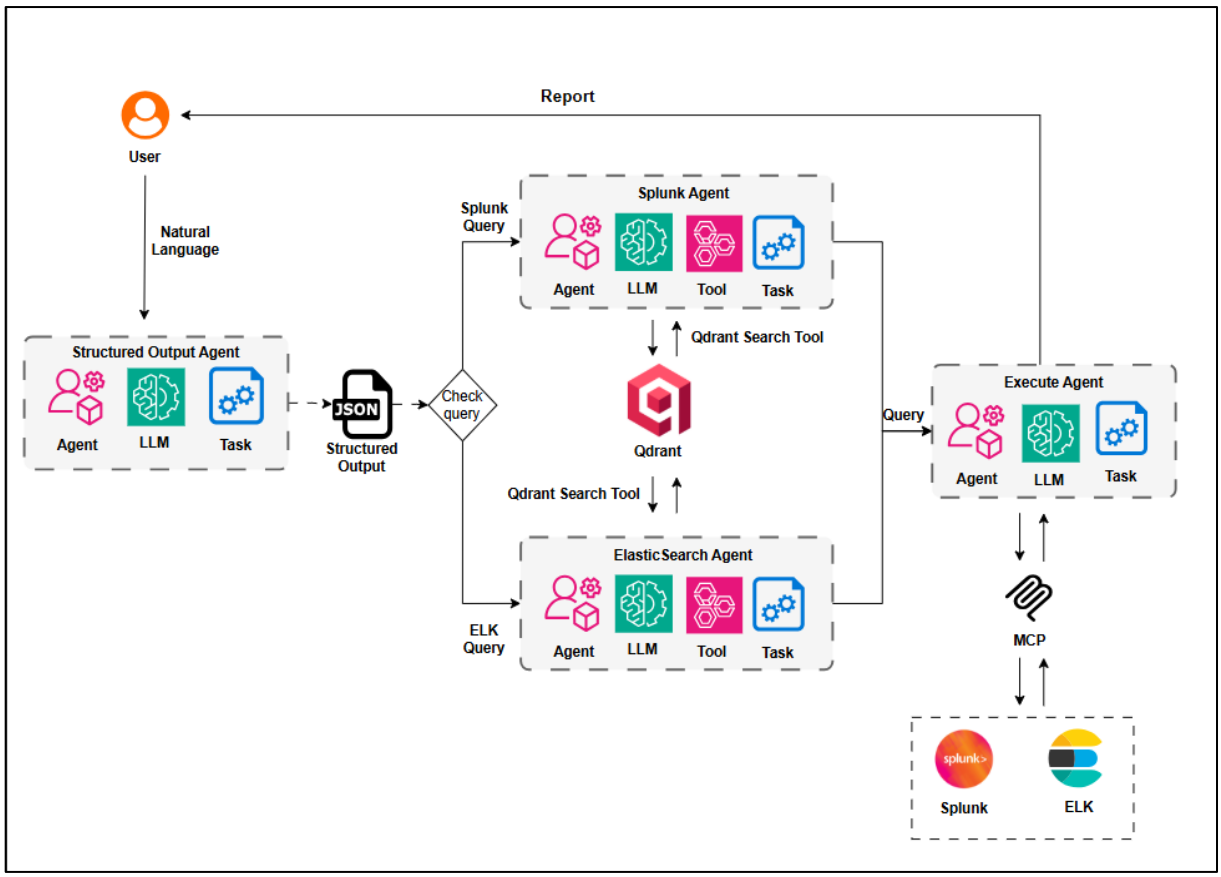


Hình 22: Web UI của kibana

3.3.2 Xây dựng hệ thống AI Agent

Mục tiêu chính của việc thiết kế kiến trúc AI-Agent này là tự động hóa hoàn toàn quá trình chúng ta phân tích các truy vấn log. Nói cách khác, người dùng chỉ cần nhập yêu cầu bằng ngôn ngữ tự nhiên (giống như khi nói chuyện) và hệ thống sẽ tự xử lý.

Để làm được điều đó một cách hiệu quả, nhóm nghiên cứu đã xây dựng hệ thống theo mô hình chuỗi các Tác nhân (Agent) liên kết chặt chẽ. Mỗi Tác nhân được chuyên môn hóa cho một nhiệm vụ cụ thể. Ví dụ, một Agent có thể chuyên hiểu truy vấn, Agent khác chuyên dịch sang lệnh truy vấn cơ sở dữ liệu (từ SQLunk hay ElasticSearch), và Agent cuối cùng chuyên lấy dữ liệu.



Hình 23: Kiến trúc Agent

Việc điều phối giữa các Agent này được thực hiện thông qua CrewAI. Cách tiếp cận này giúp hệ thống của chúng ta trở nên mô-đun hóa (dễ dàng nâng cấp, thay thế từng phần), dễ mở rộng, và quan trọng nhất là mang lại độ chính xác cao khi trích xuất dữ liệu

a) Structred Output Agent

Toàn bộ quy trình xử lý trong kiến trúc hệ thống được khởi động bởi Tác nhân Xuất Dữ liệu Có Cấu Trúc (Structured Output Agent). Thành phần này giữ vai trò trung tâm trong giai đoạn đầu của chuỗi xử lý, hoạt động như một bộ chuyển đổi sơ cấp có nhiệm vụ ánh xạ các yêu cầu được biểu đạt bằng ngôn ngữ tự nhiên của người dùng—vốn mang tính linh hoạt và tiềm ẩn nhiều mức độ mơ hồ—thành một biểu diễn dữ liệu có cấu trúc ở định dạng JSON đã được chuẩn hóa.

```
Structred_Output_agent = Agent(
    name="Structured Output Agent",
    role="You receive a natural-language query from the user.",
    goal="Your task is to convert it into a concise, normalized JSON object describing the query intent.",
    backstory=(
        """
        You are an expert in understanding user queries and translating them into structured JSON format.
        """
    ),
    llm=load_llm(),
)
```

Hình 24: Triển khai Structred Output Agent

Về mặt triển khai, tác nhân này được xây dựng dựa trên một mô hình ngôn ngữ lớn (LLM) và được cấu hình với mục tiêu rõ ràng là tiếp nhận truy vấn ngôn ngữ tự nhiên, sau đó chuyển đổi truy vấn đó thành một đối tượng JSON ngắn gọn và nhất quán, phản ánh chính xác ý định truy vấn của người dùng. Cấu hình của tác nhân bao gồm vai trò, mục tiêu và ngữ cảnh hoạt động, nhằm bảo đảm khả năng diễn giải chính xác và ổn định trong quá trình xử lý.

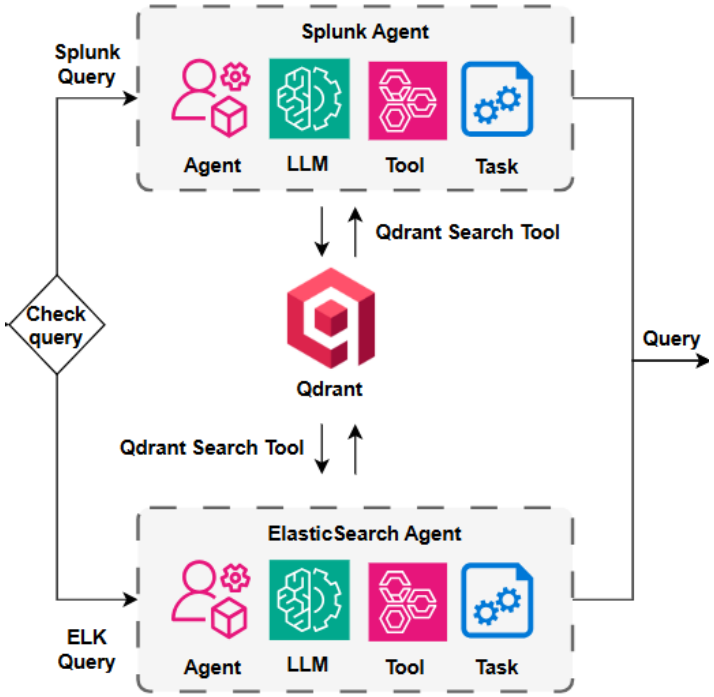
Về mặt logic xử lý, Structured Output Agent thực hiện quá trình phân tích ngữ nghĩa chuyên sâu nhằm xác định và trích xuất các tham số cốt lõi của truy vấn. Các tham số này bao gồm bản chất của yêu cầu phân tích, phạm vi thời gian liên quan, nguồn dữ liệu mục tiêu, các trường dữ liệu cần quan tâm, cũng như mục tiêu phân tích mà người dùng mong muốn đạt được. Việc trích xuất các thành phần này cho phép chuyển đổi một yêu cầu tự do sang một biểu diễn có cấu trúc và dễ xử lý hơn trong các giai đoạn tiếp theo.

Việc chuẩn hóa truy vấn ngay từ giai đoạn đầu được xem là một quyết định mang tính chiến lược trong thiết kế kiến trúc. Cách tiếp cận này giúp loại bỏ sự mơ hồ vốn có của ngôn ngữ tự nhiên, đồng thời thiết lập một giao diện dữ liệu thống nhất giữa các thành phần trong hệ thống. Nhờ đó, các tác nhân chuyên biệt ở các giai đoạn xử lý sau có thể phối hợp hoạt động một cách đồng bộ, hiệu quả và giảm thiểu đáng kể nguy cơ sai lệch trong quá trình chuyển đổi và thực thi truy vấn.

b) Splunk Agent, ELK Agent và Qdrant

Sau khi cấu trúc JSON biểu diễn ý định truy vấn được hình thành một cách rõ ràng và nhất quán, quá trình xử lý được chuyển giao cho các tác nhân chuyên biệt tương ứng với

từng nền tảng SIEM. Trong trường hợp yêu cầu được phân loại thuộc về Splunk, Splunk Agent sẽ đảm nhận vai trò tiếp nhận cấu trúc JSON trung gian và diễn giải nó thành một câu truy vấn SPL (Search Processing Language) hoàn chỉnh.



Hình 25: Kiến trúc Splunk Agent, ELK Agent và Qdrant

Splunk Agent không chỉ thực hiện việc chuyển đổi cú pháp một cách trực tiếp, mà còn sử dụng mô hình ngôn ngữ lớn để xây dựng và tối ưu hóa câu lệnh truy vấn. Để nâng cao độ chính xác và tính hiệu quả của truy vấn được sinh ra, tác nhân này được tích hợp Qdrant Search Tool, cho phép tương tác trực tiếp với hệ thống lưu trữ vector Qdrant. Thông qua công cụ này, Splunk Agent thực hiện các phép tìm kiếm ngữ nghĩa trong không gian vector nhằm truy xuất các câu truy vấn SPL tương tự đã từng được hệ thống sử dụng thành công trong quá khứ.

Cơ chế tra cứu dựa trên không gian vector cho phép hệ thống khai thác tri thức kinh nghiệm đã được tích lũy, bao gồm các mẫu truy vấn hiệu quả, các kịch bản phân tích phổ biến hoặc các cấu trúc truy vấn đặc trưng cho từng loại sự kiện. Việc tái sử dụng các tri thức này góp phần giảm thiểu rủi ro sinh ra truy vấn không phù hợp, hạn chế lặp lại các sai sót trước đó và nâng cao đáng kể chất lượng của câu truy vấn cuối cùng.

Elasticsearch Agent vận hành theo một quy trình logic tương tự. Tuy nhiên, thay vì sinh ra truy vấn SPL, tác nhân này chịu trách nhiệm tạo lập các câu truy vấn DSL (Domain Specific Language) tương thích với hệ sinh thái ELK (Elasticsearch, Logstash, Kibana). Elasticsearch Agent cũng sử dụng Qdrant Search Tool để thực hiện tìm kiếm ngữ nghĩa trong không gian vector, từ đó tham chiếu đến các truy vấn DSL, mẫu phân tích hoặc cấu trúc dữ liệu đã được chứng minh là hiệu quả trong các lần xử lý trước.

Trong toàn bộ kiến trúc, Qdrant được triển khai như một bộ nhớ vector trung tâm, đóng vai trò lưu trữ và quản lý tri thức kinh nghiệm của hệ thống AI-Agent. Các thành phần tri thức quan trọng—bao gồm các truy vấn đã được thực thi, các mẫu tấn công, các cấu trúc sự kiện điển hình và các tác vụ phân tích hiệu quả—được mã hóa dưới dạng vector nhúng (embedding) và lưu trữ trong không gian vector này.

Vai trò của Qdrant không chỉ dừng lại ở chức năng lưu trữ, mà còn đóng vai trò như một lớp tri thức có thể truy vấn được. Thông qua Qdrant Search Tool, các tác nhân có khả năng khai thác tìm kiếm ngữ nghĩa để phát hiện các điểm tương đồng, tái sử dụng tri thức sẵn có hoặc kết hợp các kinh nghiệm trước đó trong quá trình xây dựng truy vấn mới. Cách tiếp cận này góp phần tăng cường tính thích nghi, khả năng học hỏi gián tiếp và độ tin cậy tổng thể của hệ thống.

c) Execute agent

Sau khi câu truy vấn tối ưu được hình thành, Execute Agent đảm nhận vai trò như một cầu nối thực thi giữa tầng nhận thức dựa trên trí tuệ nhân tạo và các hệ thống dữ liệu SIEM trong môi trường triển khai thực tế. Tác nhân này tiếp nhận câu truy vấn đã được tinh chỉnh từ các tác nhân chuyên biệt ở giai đoạn trước, đồng thời chịu trách nhiệm điều phối quá trình thực thi truy vấn trên nền tảng SIEM tương ứng.

Tùy thuộc vào loại hệ thống SIEM được xác định trong bước xử lý trước đó, Execute Agent sẽ lựa chọn và sử dụng giao thức MCP (Model Context Protocol) phù hợp. Cụ thể, trong trường hợp truy vấn được направ tới hệ thống Splunk, tác nhân sẽ sử dụng SplunkMCP để thiết lập ngữ cảnh và thực thi truy vấn theo đúng cơ chế truy cập và ngôn ngữ truy vấn đặc thù của Splunk. Ngược lại, đối với các truy vấn hướng tới hệ thống

ELK, Elasticsearch MCP sẽ được sử dụng nhằm bảo đảm khả năng tương thích với kiến trúc và cơ chế tìm kiếm của Elasticsearch. Cách tiếp cận này cho phép hệ thống duy trì tính linh hoạt trong thực thi, đồng thời tách biệt rõ ràng giữa logic truy vấn ở mức trừu tượng và cơ chế truy cập cụ thể của từng nền tảng SIEM.

Sau khi quá trình thực thi hoàn tất, dữ liệu kết quả thu được từ các hệ thống SIEM sẽ được chuyển ngược trở lại cho Execute Agent để tiếp tục xử lý. Ở giai đoạn cuối cùng, mô hình ngôn ngữ lớn được sử dụng để tổng hợp, diễn giải và trình bày dữ liệu dưới dạng một báo cáo hoàn chỉnh. Quá trình này không chỉ dừng lại ở việc mô tả hay tóm tắt kết quả truy vấn, mà còn bao gồm việc phân tích các xu hướng tiềm ẩn và mối quan hệ có ý nghĩa trong dữ liệu log.

Mục tiêu của giai đoạn báo cáo là hỗ trợ người dùng tiếp cận tri thức ở mức khái quát và chuyên sâu hơn, giúp làm rõ bản chất của các sự kiện an ninh thay vì phải trực tiếp xử lý một tập dữ liệu thô và rời rạc. Thông qua cơ chế hợp tác chặt chẽ và sự chuyên môn hóa rõ ràng giữa các tác nhân trong kiến trúc, hệ thống không chỉ thực hiện hiệu quả việc truy vấn dữ liệu mà còn thành công trong việc chuyển hóa dữ liệu thô thành tri thức có giá trị ứng dụng cao trong bối cảnh giám sát và phân tích an ninh thông tin.

3.4. Tổng kết

Tổng thể, kiến trúc hệ thống được thiết kế với luồng dữ liệu vào và ra được tổ chức theo một trình tự xử lý rõ ràng và nhất quán, đồng thời phân tách các chức năng xử lý thành các tác nhân độc lập nhằm nâng cao độ chính xác và độ tin cậy của kết quả đầu ra. Cách tiếp cận này cho phép mỗi tác nhân tập trung xử lý một phạm vi nhiệm vụ xác định, từ đó giảm thiểu sự chồng chéo chức năng và hạn chế sai lệch trong quá trình xử lý.

Việc chia nhỏ hệ thống thành các mô-đun tác nhân chuyên trách không chỉ giúp đơn giản hóa thiết kế tổng thể mà còn tạo điều kiện thuận lợi cho các hoạt động kiểm thử, đánh giá và tối ưu hóa hiệu năng của từng thành phần một cách độc lập. Nhờ đó, hệ thống có khả năng thích ứng linh hoạt trước các yêu cầu mở rộng hoặc điều chỉnh trong tương lai mà không ảnh hưởng đáng kể đến logic xử lý cốt lõi.

Bên cạnh đó, việc tích hợp CrewAI với vai trò là cơ chế điều phối trung tâm giúp bảo đảm sự phối hợp nhịp nhàng giữa các tác nhân trong toàn bộ vòng đời xử lý truy vấn. CrewAI chịu trách nhiệm quản lý quá trình khởi tạo, điều phối thực thi và tổng hợp kết quả, qua đó duy trì tính nhất quán trong luồng xử lý và ngăn chặn các tình huống tác nhân hoạt động sai lệch hoặc sinh ra kết quả ngoài mong đợi. Sự kết hợp này góp phần củng cố tính ổn định, khả năng kiểm soát và mức độ tin cậy của kiến trúc tổng thể.

Chương 4: Thực nghiệm và đánh giá

4.1. Chuẩn bị

Để triển khai và đánh giá hệ thống theo kiến trúc đề xuất, một môi trường thực nghiệm được thiết lập với các thành phần phần mềm và công nghệ phù hợp cho việc phát triển hệ thống đa tác tử dựa trên mô hình ngôn ngữ lớn và nền tảng SIEM. Các công cụ và nền tảng được lựa chọn nhằm bảo đảm khả năng triển khai linh hoạt, dễ mở rộng và thuận tiện cho quá trình kiểm thử.

Hệ thống được phát triển bằng ngôn ngữ lập trình Python, cho phép xây dựng các module xử lý, tích hợp mô hình ngôn ngữ lớn cũng như kết nối với các dịch vụ bên ngoài một cách hiệu quả. Python đóng vai trò là nền tảng chính cho toàn bộ logic xử lý và điều phối trong hệ thống.

Cơ chế điều phối và quản lý các tác tử được hiện thực hóa thông qua framework CrewAI. Framework này chịu trách nhiệm tổ chức luồng thực thi, phân công nhiệm vụ cho từng agent và điều phối sự phối hợp giữa các tác nhân trong suốt quá trình xử lý truy vấn.

Về năng lực xử lý ngôn ngữ tự nhiên và suy luận, hệ thống sử dụng mô hình ngôn ngữ lớn Gemini-2.5-Flash được triển khai trên nền tảng Vertex AI. Mô hình này được sử dụng để phân tích yêu cầu ngôn ngữ tự nhiên, sinh cấu trúc truy vấn có tổ chức và hỗ trợ các tác vụ phân tích, tóm tắt kết quả.

Để phục vụ cho kỹ thuật Retrieval-Augmented Generation (RAG), hệ thống tích hợp cơ sở dữ liệu vector Qdrant. Qdrant được sử dụng để lưu trữ và truy xuất các biểu diễn vector của tri thức liên quan đến an ninh mạng, cú pháp truy vấn và các mẫu phân tích, từ đó hỗ trợ quá trình sinh truy vấn chính xác và nhất quán hơn.

Giao diện người dùng được xây dựng dưới dạng một ứng dụng web độc lập, phát triển dựa trên các framework phổ biến như Flask hoặc Django. Giao diện này cho phép người dùng nhập yêu cầu bằng ngôn ngữ tự nhiên, lựa chọn hệ thống SIEM mục tiêu và theo dõi kết quả truy vấn cùng phần giải thích tương ứng.

Cuối cùng, để phục vụ cho mục đích thử nghiệm và đánh giá, các hệ thống SIEM được triển khai trong môi trường mô phỏng, bao gồm Splunk và Elasticsearch (ELK Stack), sử dụng các bộ dữ liệu log mẫu. Môi trường này cho phép kiểm tra khả năng chuyển đổi, thực thi truy vấn và đánh giá hiệu quả hoạt động của hệ thống trong các kịch bản giám sát và phân tích an ninh mạng.

Các kịch bản được triển khai bao gồm:

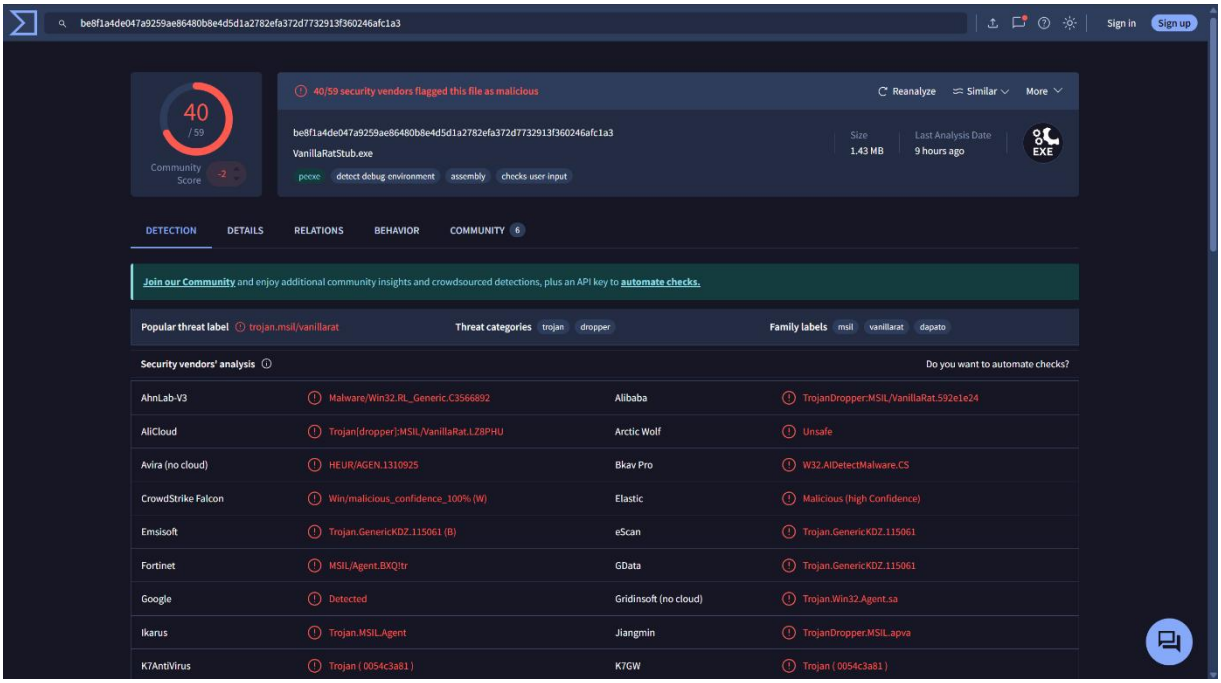
- Kịch bản 1: Host windows chạy file mã độc – Trong đó host window sẽ chạy một file mã độc thật được thu thập trên nền tảng bazzar và mã độc này có một số hành vi nguy hiểm đến hệ thống. Sau đó nhóm tiến hành sử dụng hệ thống AI Agent để điều tra sự cố này.
- Kịch bản 2: Webservice bị tấn công mạng – Host 192.168.10.50 có vai trò webservice bị tấn công bởi một cuộc tấn công mạng. Sau đó, nhóm sẽ tiếp tục tiến hành điều tra sự cố này thông qua hệ thống AI Agent.

4.2. Thực nghiệm

4.2.1. Kịch bản 1: Host windows chạy file malware

a) Thông tin mẫu kiểm thử

Thông tin mẫu	Hành vi
SHA-256: BE8F1A4DE047A9259AE86480B8E4D5D1A2782EFA372D7732913F360246AFC1A3	Hành vi của mẫu này có tác động trực tiếp đến hệ thống bằng cách chỉnh sửa các khóa Registry tại đường dẫn HKLM\SYSTEM\ControlSet001\Services\Tcpip\Parameters, qua đó ảnh hưởng đến cấu hình mạng của hệ điều hành. Ngoài ra, mẫu còn thực hiện các truy vấn DNS tới tên miền apsom.org, đây là một dấu hiệu đáng ngờ cho thấy khả năng liên lạc với máy chủ điều khiển và chỉ huy (Command and Control – C2).



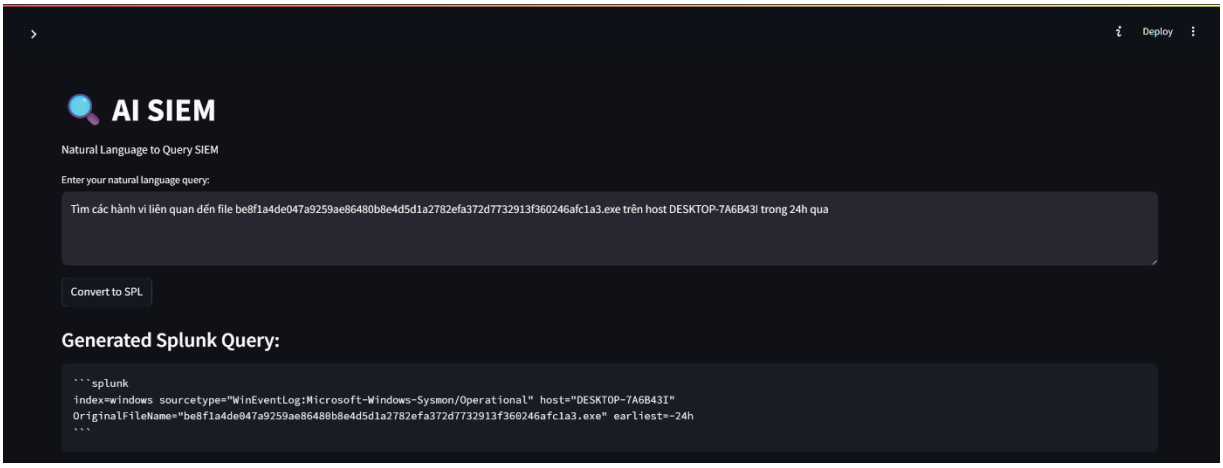
Hình 26: Tra cứu mã độc trên virustotal

b) Kết quả phân tích trên Splunk Agent

Sau khi thực hiện chạy mã độc ở phía host thì truy cập đến hệ thống AI Agent để thực hiện lệnh:

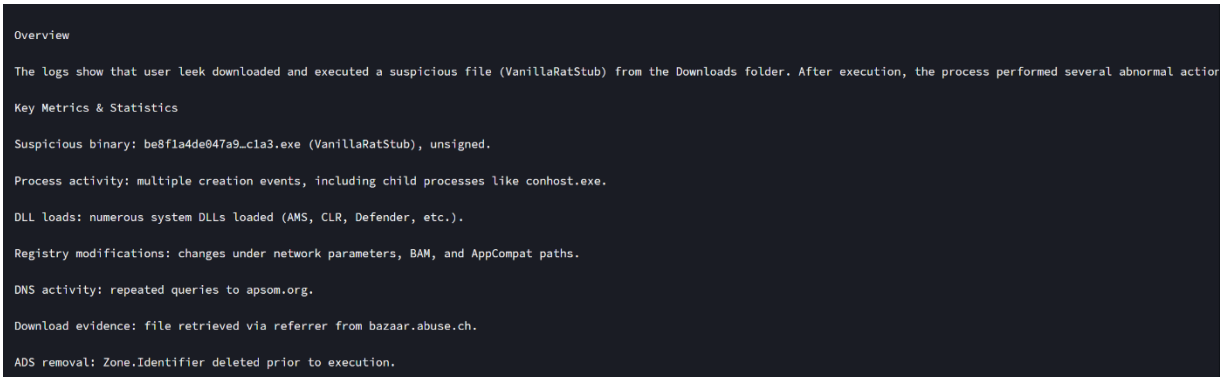
Tìm các hành vi liên quan đến file BE8F1A4DE047A9259AE86480B8E4D5D1A2782EFA372D7732913F360246AFC1A3.exe trên host DESKTOP-7A6843I trong 24h qua

Hình 27: Lệnh truy vấn



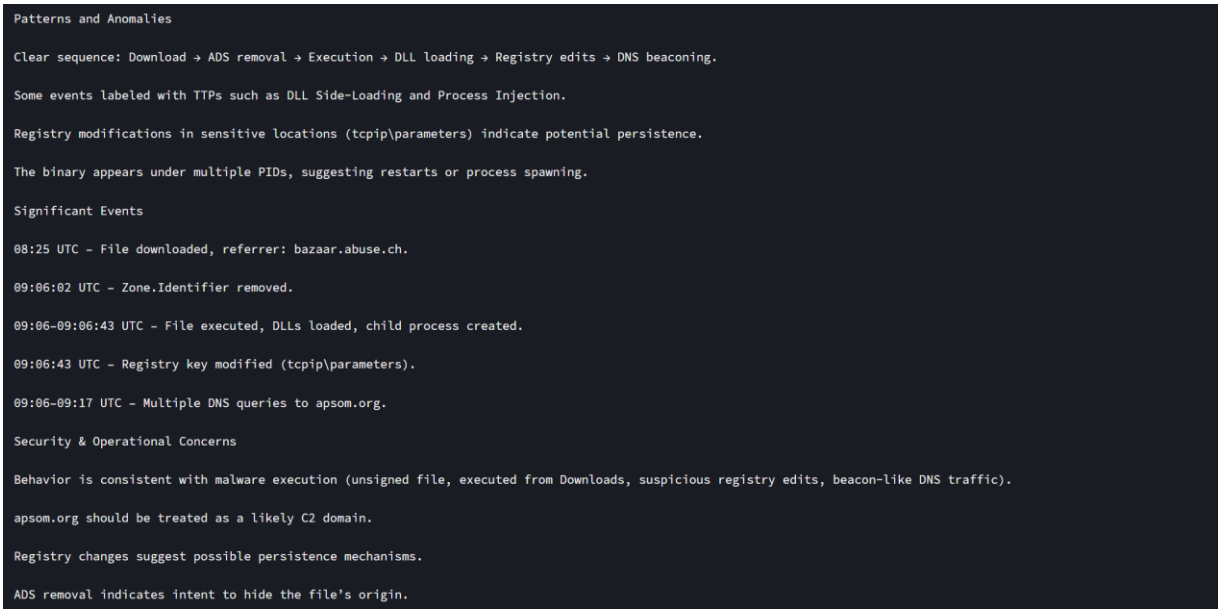
Hình 28: Kết quả Splunk Query của Splunk Agent

Trong kịch bản thử nghiệm với Splunk Agent, hệ thống đã tạo thành công câu truy vấn SPL dựa trên cấu trúc truy vấn trung gian và thực hiện truy xuất dữ liệu từ nền tảng Splunk. Sau khi dữ liệu log được thu thập, tác nhân tiến hành phân tích và tổng hợp kết quả.



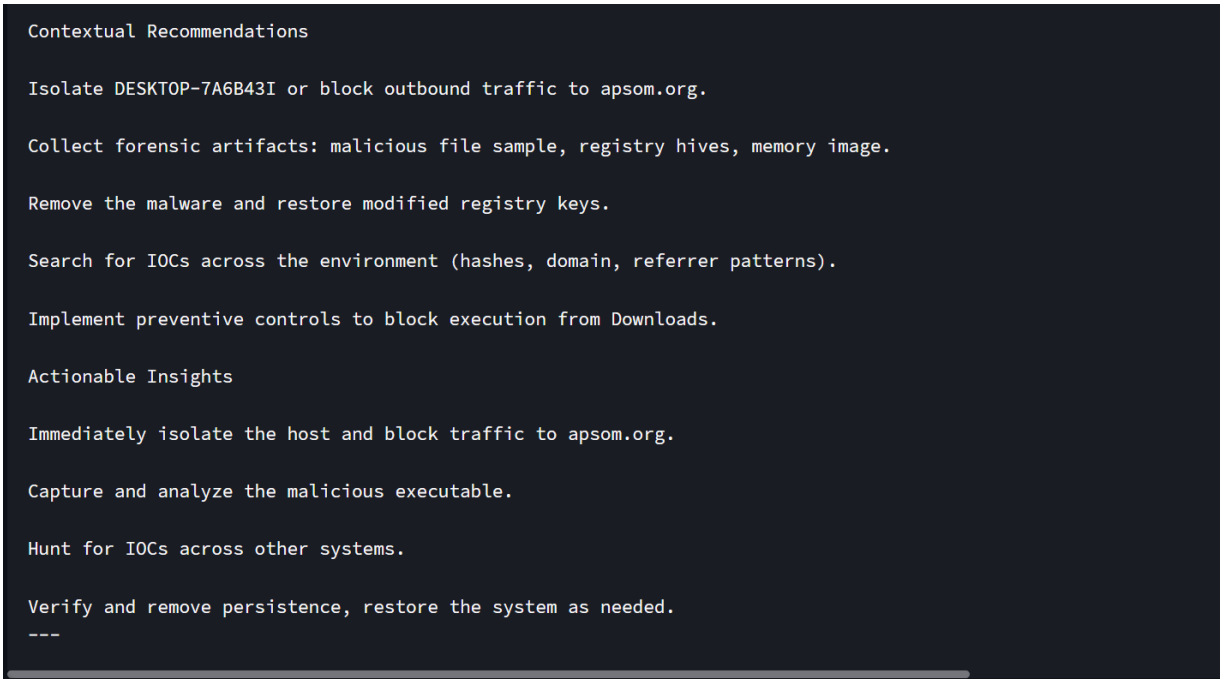
Hình 29: Tổng quan trong report của Splunk Agent

Ở mức tổng quan, các log ghi nhận rằng người dùng có tên *leek* đã tải xuống và thực thi một tệp lạ (VanillaRatStub) từ thư mục *Downloads*. Sau khi được thực thi, tiến trình này tạo ra nhiều hoạt động đáng ngờ, bao gồm việc nạp các thư viện DLL hệ thống, chỉnh sửa Registry và thực hiện truy vấn DNS ra bên ngoài đến tên miền apsom.org. Chuỗi hành vi này cho thấy dấu hiệu rõ ràng của một hoạt động bất thường, nhiều khả năng liên quan đến mã độc dạng Remote Access Trojan (RAT).Các phân tích chính:



Hình 30: Phân tích của Splunk Agent

Phân tích chi tiết cho thấy một chuỗi hành động liên tục theo trình tự: tải xuống tệp, xóa Alternate Data Streams (ADS), thực thi tiến trình, nạp DLL, chỉnh sửa Registry và thực hiện DNS beaconing. Một số sự kiện được gắn nhãn theo các chiến thuật, kỹ thuật và thủ tục (TTP) như DLL Side-Loading và Process Injection. Việc chỉnh sửa các khóa Registry thuộc khu vực nhạy cảm liên quan đến cấu hình TCP/IP cho thấy khả năng thiết lập hoặc duy trì cơ chế tồn tại lâu dài (persistence). Ngoài ra, sự xuất hiện của nhiều PID khác nhau tương ứng với cùng một binary cho thấy khả năng tiến trình được khởi chạy lại hoặc tự sinh tiến trình con.



Hình 31: Đề xuất của Splunk Agent

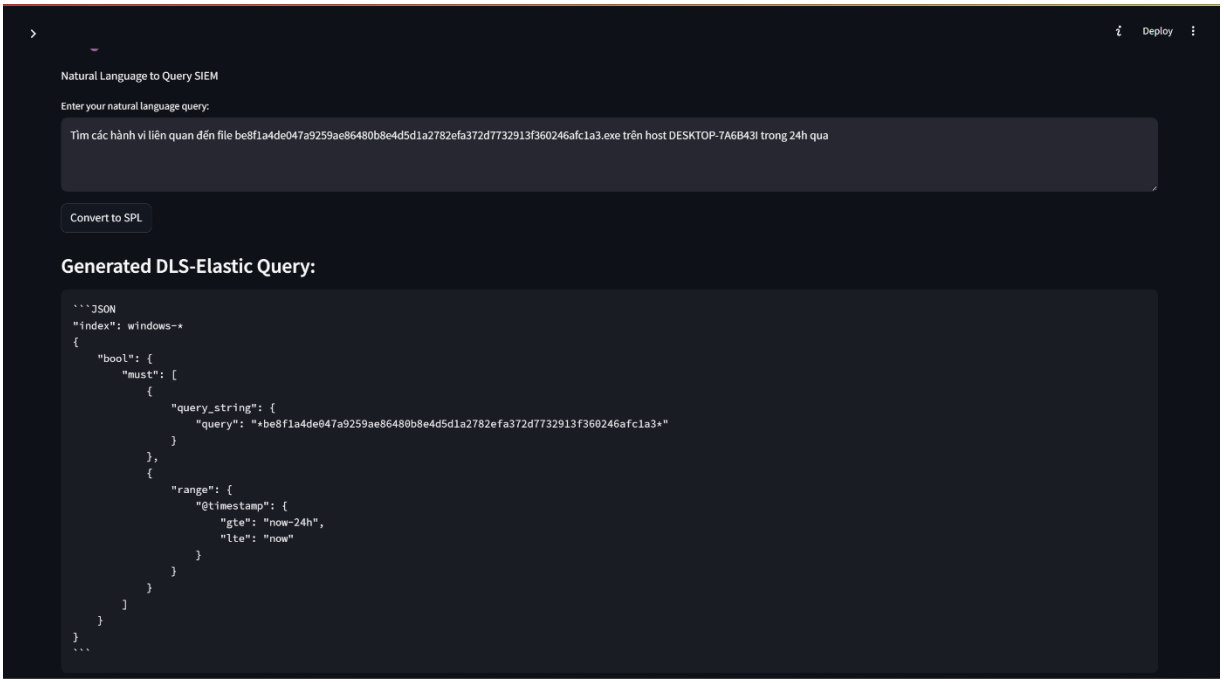
Dựa trên kết quả phân tích, hệ thống đưa ra các khuyến nghị ứng phó, bao gồm cô lập máy trạm bị ảnh hưởng hoặc chặn lưu lượng truy cập ra ngoài tới tên miền apsom.org, thu thập các bằng chứng pháp y như mẫu tệp độc hại, các khóa Registry liên quan và ảnh bộ nhớ, tiến hành loại bỏ mã độc và khôi phục các cấu hình bị thay đổi, đồng thời tìm kiếm các chỉ báo xâm nhập (IOC) trên toàn bộ môi trường. Ngoài ra, hệ thống cũng đề xuất áp dụng các biện pháp phòng ngừa nhằm hạn chế việc thực thi tệp từ thư mục *Downloads*.

c) Kết quả trên ELK Agent

Tiếp tục truy vấn với hệ thống ELK Agent, phần truy vấn cũng tương tự với Splunk Agent:

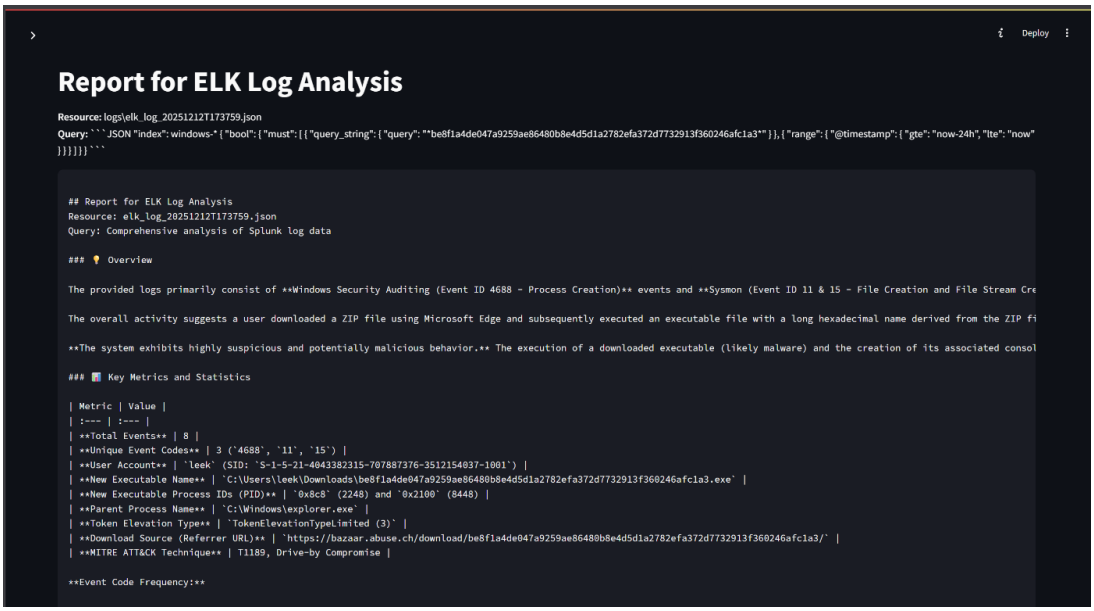
Tìm các hành vi liên quan đến file BE8F1A4DE047A9259AE86480B8E4D5D1A2782EFA372D7732913F360246AFC1A3.exe trên host DESKTOP-7A6843I trong 24h qua

Hình 32: Lệnh truy vấn đến ELK Agent



Hình 33: Kết quả truy vấn trả về của ELK Agent

Đối với ELK Agent, hệ thống đã sinh thành công câu truy vấn DSL nhằm thu thập đầy đủ các bản ghi có liên quan đến tên tệp và tiến trình mục tiêu. Tương tự như Splunk Agent, ELK Agent thực hiện truy xuất dữ liệu từ Elasticsearch và tiến hành phân tích các log thu được.



Hình 34: Report trả về của ELK Agent

Kết quả phân tích dựa trên dữ liệu từ ELK cho thấy mức độ tương đồng cao với kết quả của Splunk Agent. Các chỉ báo xâm nhập được phát hiện, chuỗi hành vi của tiến trình và các nhận định về mục đích hoạt động đều nhất quán giữa hai nền tảng. Các khuyến nghị an toàn và biện pháp ứng phó sự cố được đề xuất cũng tương đồng về nội dung và mục tiêu.

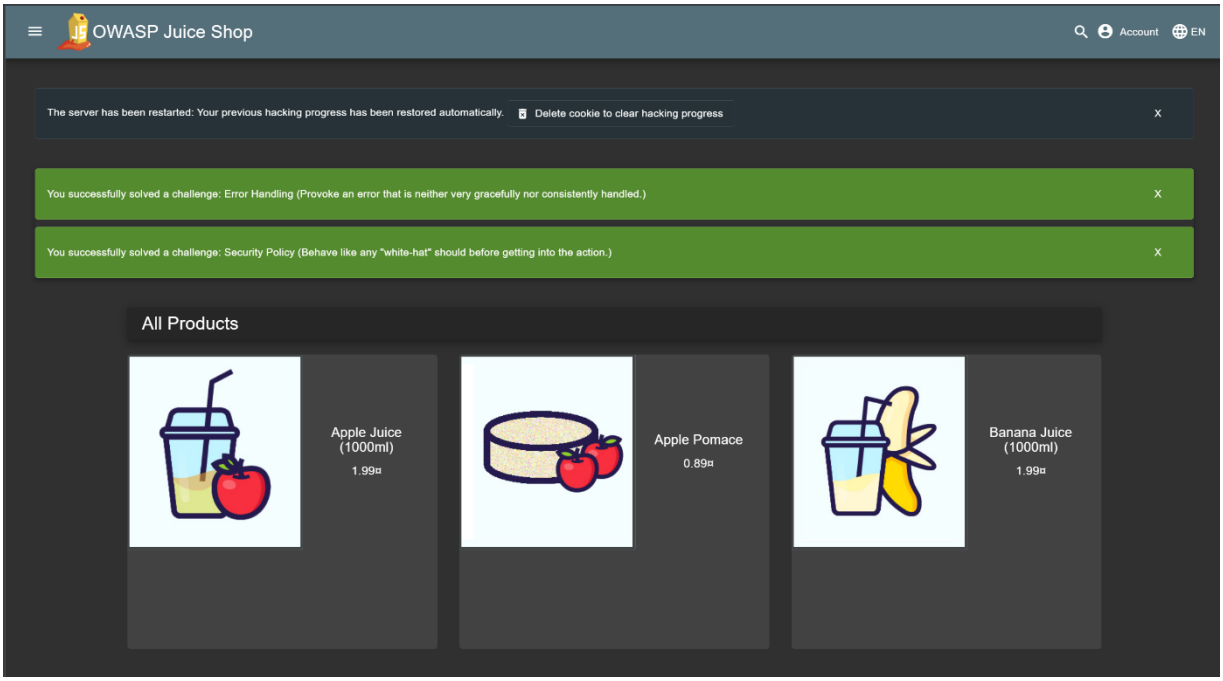
d) Tổng kết

Kết quả thử nghiệm cho thấy hệ thống AI-Agent có khả năng thực hiện hiệu quả việc tìm kiếm và phân tích các sự kiện liên quan đến mã độc hoặc các tệp đáng ngờ trên nhiều nền tảng SIEM khác nhau. Hệ thống không chỉ truy xuất được dữ liệu liên quan mà còn tổng hợp, phân tích và trình bày báo cáo ở mức độ chi tiết và có giá trị thực tiễn. So sánh với các báo cáo phân tích mã độc được công bố bởi các chuyên gia an ninh trên các nguồn công khai, các dữ liệu và nhận định mà hệ thống thu thập được cho thấy mức độ tương đồng cao, qua đó khẳng định tính khả thi và hiệu quả của kiến trúc đề xuất.

4.2.2. Kịch bản 2: Tấn công mạng trên host 192.168.10.50

a) Môi trường

Môi trường mục tiêu trong kịch bản này là ứng dụng OWASP Juice Shop, một ứng dụng web được thiết kế có chủ đích để chứa các lỗ hổng bảo mật. OWASP Juice Shop được xem là một trong những ứng dụng web không an toàn hiện đại và toàn diện, thường được sử dụng trong đào tạo an ninh mạng, nâng cao nhận thức bảo mật, các cuộc thi CTF cũng như trong quá trình kiểm thử và đánh giá các công cụ an ninh. Ứng dụng này bao phủ đầy đủ các nhóm lỗ hổng thuộc danh sách OWASP Top Ten, qua đó phù hợp để mô phỏng các kịch bản tấn công web trong môi trường thử nghiệm.



Hình 35: OWASP Juice Shop

Trong kịch bản này, công cụ `dirsearch` được sử dụng để thực hiện quá trình quét thư mục và tài nguyên trên ứng dụng web mục tiêu. Hoạt động quét này nhằm phát hiện các đường dẫn ẩn, tài nguyên nhạy cảm hoặc các điểm có khả năng bị khai thác, từ đó tạo ra các mẫu lưu lượng mạng đặc trưng của một cuộc tấn công dò quét ứng dụng web.

```
[11:14:33] 500 -    3KB - /api/timelion/run
[11:14:33] 500 -    3KB - /api/v1/swagger.yaml
[11:14:33] 500 -    3KB - /api/v2
[11:14:33] 500 -    3KB - /api/v2/
[11:14:34] 500 -    3KB - /api/v3
[11:14:34] 500 -    3KB - /api/v2/helpdesk/discover
[11:14:34] 500 -    3KB - /api/v2/swagger.json
[11:14:34] 500 -    3KB - /api/version
[11:14:34] 500 -    3KB - /api/v4
[11:14:34] 500 -    3KB - /api/v2/swagger.yaml
[11:14:34] 500 -    3KB - /api/vendor/phpunit/phpunit/phpunit
[11:14:34] 500 -    3KB - /apibuild.pyc
[11:14:34] 500 -    3KB - /api/whoami
[11:14:34] 500 -    3KB - /apidocs
[11:14:34] 500 -    3KB - /apidoc
[11:14:34] 500 -    3KB - /apiserver-aggregator.key
[11:14:34] 500 -    3KB - /apiserver-aggregator.cert
[11:14:34] 500 -    3KB - /apis
[11:14:34] 500 -    3KB - /apiserver-key.pem
[11:14:34] 500 -    3KB - /apiserver-aggregator-ca.cert
[11:14:34] 500 -    3KB - /apiserver-client.crt
[11:14:35] 301 -  156B  - /assets  ->  /assets/
[11:14:42] 200 -    9KB - /common.js
[#########           ] 47%   5482/11460        6/s       job:1/1  errors:0Exception in thread Thread-18 (thread_proc):
```

Hình 36: Scan với dirsearch

b) Kết quả điều tra trên Splunk.

Thực hiện truy vấn trên Splunk Agent để điều tra cuộc tấn công mạng với truy vấn

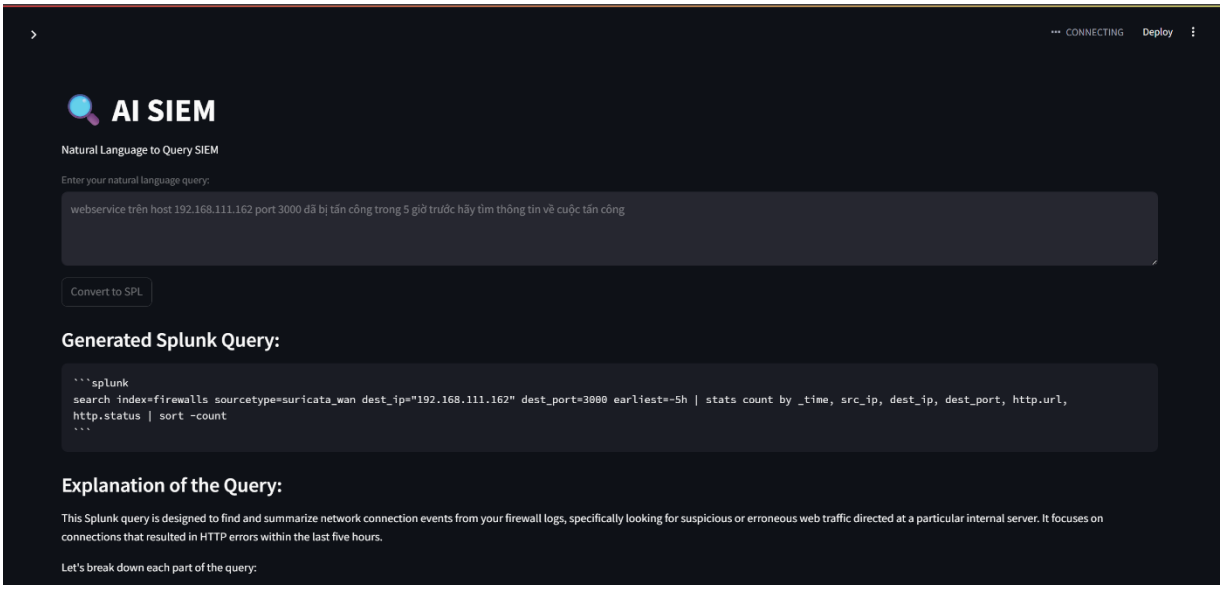
Webservice trên host 192.168.111.162 port 3000 đã bị tấn công trong 5 giờ trước hãy tìm thông tin về cuộc tấn công

Hình 37: Truy vấn đến Splunk Agent

Dữ liệu log thu thập được từ hệ thống giám sát mạng được truy vấn thông qua Splunk Agent bằng câu lệnh SPL như sau:

search index=firewalls sourcetype=suricata_wan dest_ip="192.168.111.162" dest_port=3000 earliest=-5h | stats count by _time, src_ip, dest_ip, dest_port, http.url, http.status | sort -count

Hình 38: Tuy vấn trả về của Splunk Agent

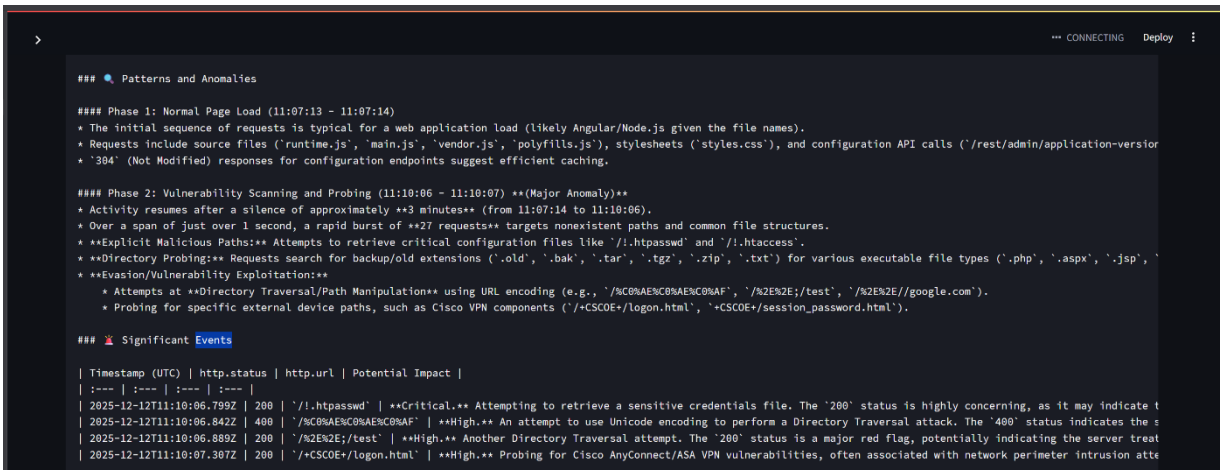


Hình 39: Kết quả trả về của Splunk Agent

Kết quả truy vấn cung cấp đầy đủ các trường thông tin cần thiết phục vụ cho quá trình điều tra, bao gồm thời gian sự kiện, địa chỉ IP nguồn và đích, cổng đích, đường dẫn URL truy cập và mã trạng thái HTTP. Các thông tin này cho phép hệ thống tổng hợp và phân tích hành vi truy cập vào ứng dụng web một cách hiệu quả.

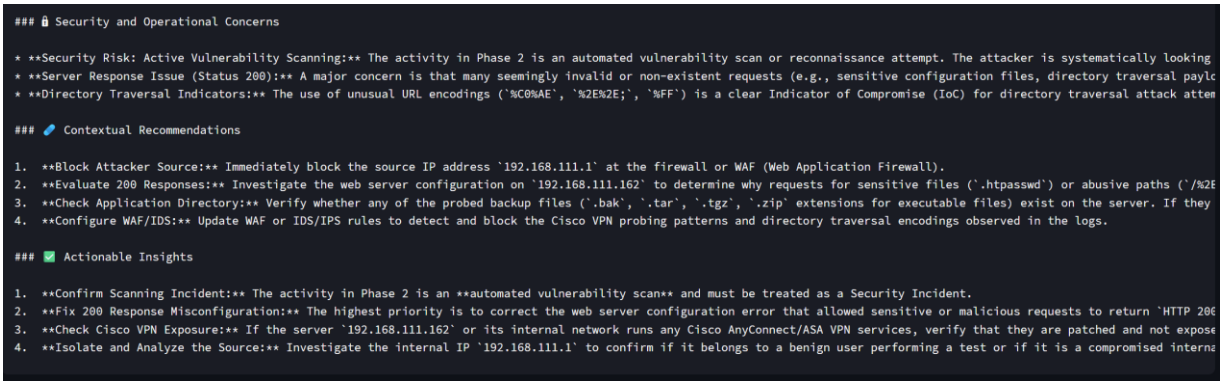
Hình 40: Report trả về của Splunk Agent

Xác định được 2 phase trong các request được trả về, phase 1 gồm các request thông thường và phase 2 gồm các request nguy hiểm



Hình 41: Phân tích của Splunk Agent

Dựa trên kết quả phân tích, hệ thống xác định được 2 phase chính trong các request được ghi nhận. Pha thứ nhất bao gồm các request thông thường, phản ánh hành vi truy cập hợp lệ hoặc không có dấu hiệu rõ ràng của tấn công. Pha thứ hai bao gồm các request mang tính nguy hiểm, thể hiện đặc trưng của hoạt động dò quét và thăm dò cấu trúc ứng dụng.



Hình 42: Phân tích của Splunk Agent

Phân tích chi tiết hai pha cho thấy sự khác biệt rõ rệt về mục tiêu truy cập và tần suất request. Các sự kiện nổi bật trong pha thứ hai được hệ thống làm nổi bật (highlight) do có mức độ rủi ro cao. Trên cơ sở đó, hệ thống tiến hành tóm tắt mức độ ảnh hưởng của cuộc tấn công và đề xuất các hành động khắc phục nhằm giảm thiểu rủi ro và ngăn chặn các hành vi tương tự trong tương lai.

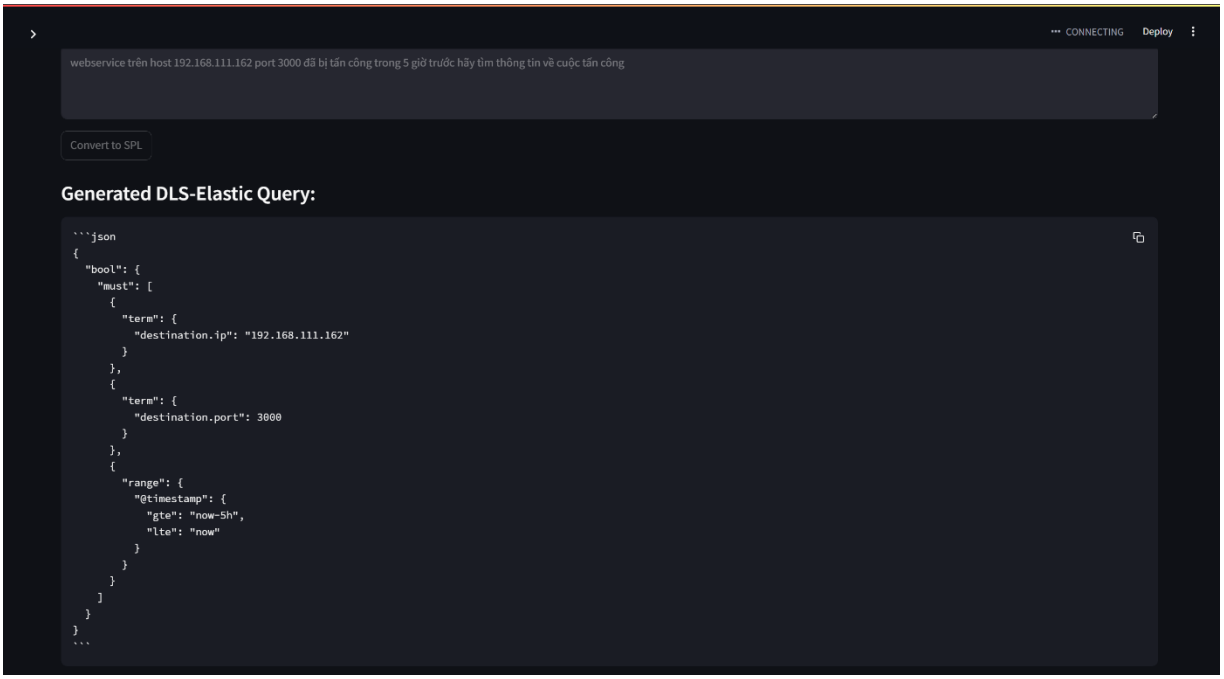
c) Kết quả trên ELK Agent

Thực hiện truy vấn tương tự ở Splunk Agent với ELK Agent, cùng một truy vấn

Webservice trên host 192.168.111.162 port 3000 đã bị tấn công trong 5 giờ trước hãy tìm thông tin về cuộc tấn công

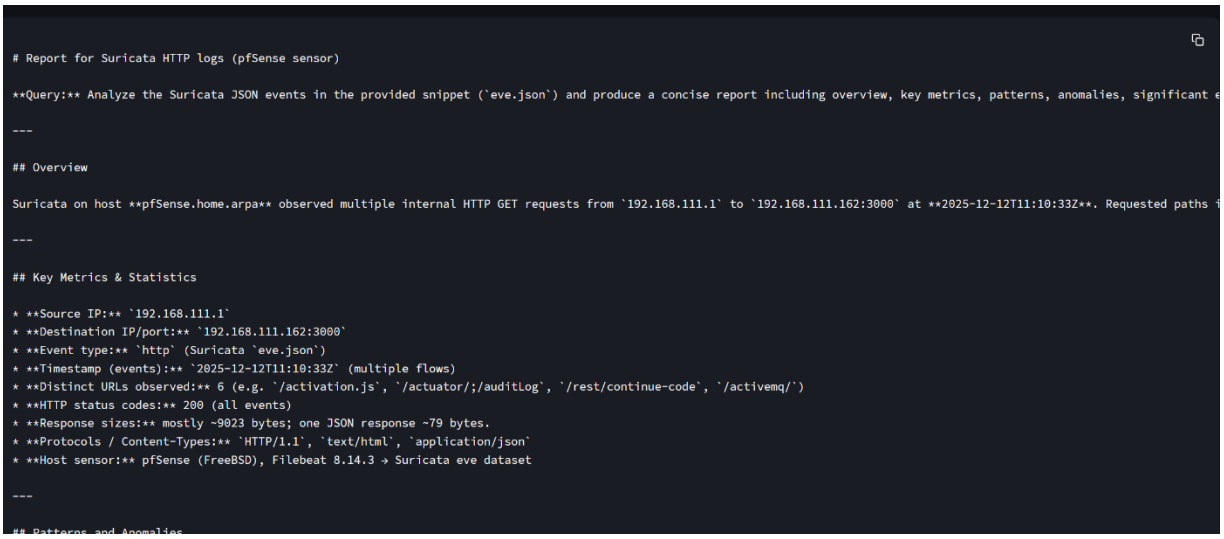
Hình 43: Truy vấn đến ELK Agent

Đối với ELK Agent, hệ thống cũng tạo ra cấu trúc truy vấn DSL đầy đủ, phản ánh chính xác các tham số đầu vào do người dùng cung cấp. ELK Agent tiến hành truy xuất dữ liệu từ Elasticsearch và thực hiện phân tích các log thu thập được.



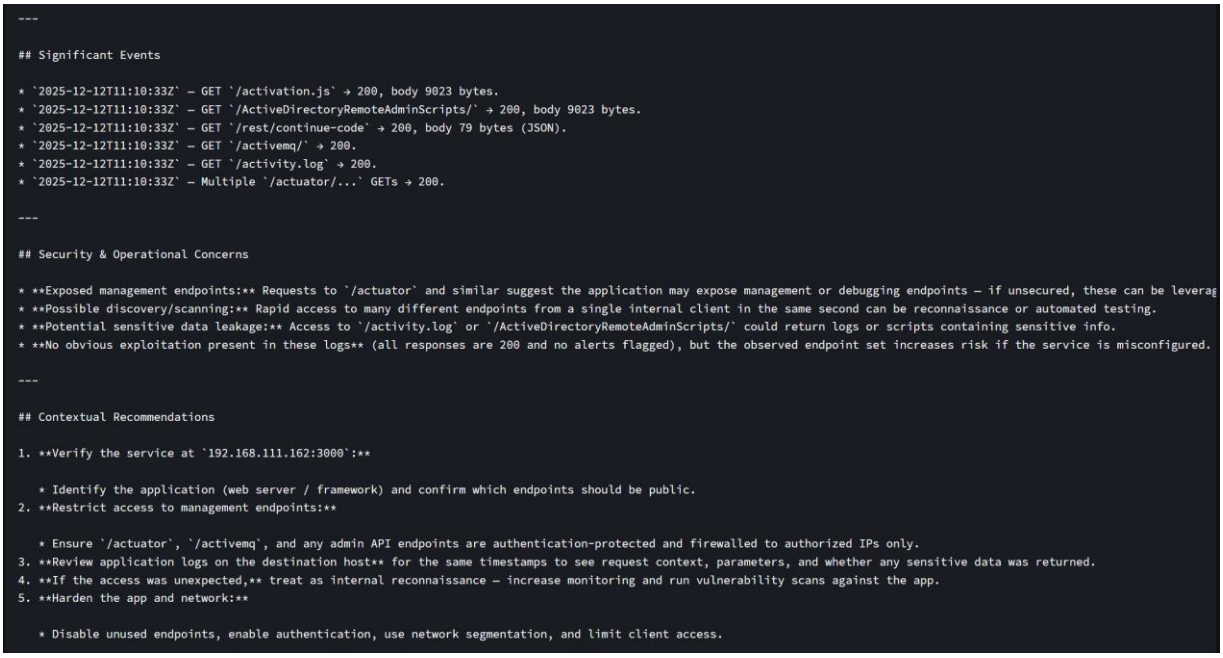
Hình 44: Truy vấn trả về của ELK Agent

Kết quả được trả về với truy vấn trên ElasticSearch để thu thập thông tin trong 5 giờ qua của host 192.168.111.162 port 3000.



Hình 45: Phân tích của ELK Agent

Sau khi trả về câu truy vấn thì ELK Agent thực hiện câu truy vấn và phân tích kết quả được trả về sau khi truy vấn trên ELK Server. Kết quả cho thấy host đã nhận nhiều HTTP request GET giống như một cuộc tấn công mạng vào hệ thống.



```
---

## Significant Events

* `2025-12-12T11:10:33Z` — GET `/activation.js` → 200, body 9023 bytes.
* `2025-12-12T11:10:33Z` — GET `/ActiveDirectoryRemoteAdminScripts/` → 200, body 9023 bytes.
* `2025-12-12T11:10:33Z` — GET `/rest/continue-code` → 200, body 79 bytes (JSON).
* `2025-12-12T11:10:33Z` — GET `/activemq/` → 200.
* `2025-12-12T11:10:33Z` — GET `/activity.log` → 200.
* `2025-12-12T11:10:33Z` — Multiple `/actuator/...` GETs → 200.

---

## Security & Operational Concerns

* **Exposed management endpoints:** Requests to `/actuator` and similar suggest the application may expose management or debugging endpoints — if unsecured, these can be leverag
* **Possible discovery/scanning:** Rapid access to many different endpoints from a single internal client in the same second can be reconnaissance or automated testing.
* **Potential sensitive data leakage:** Access to `/activity.log` or `/ActiveDirectoryRemoteAdminScripts/` could return logs or scripts containing sensitive info.
* **No obvious exploitation present in these logs** (all responses are 200 and no alerts flagged), but the observed endpoint set increases risk if the service is misconfigured.

---

## Contextual Recommendations

1. **Verify the service at `192.168.111.162:3000`:**

   * Identify the application (web server / framework) and confirm which endpoints should be public.
2. **Restrict access to management endpoints:**

   * Ensure `/actuator`, `/activemq`, and any admin API endpoints are authentication-protected and firewalled to authorized IPs only.
3. **Review application logs on the destination host** for the same timestamps to see request context, parameters, and whether any sensitive data was returned.
4. **If the access was unexpected,** treat as internal reconnaissance — increase monitoring and run vulnerability scans against the app.
5. **Harden the app and network:**

   * Disable unused endpoints, enable authentication, use network segmentation, and limit client access.
```

Hình 46: Đề xuất của ELk Agent

Trong lần thử nghiệm này, ELK Agent chủ yếu phân tích được các request thông thường và chưa phát hiện rõ các request mang dấu hiệu tấn công. Tuy nhiên, hạn chế này không xuất phát từ logic xử lý của tác nhân, mà do giới hạn số lượng log được truy xuất trong mỗi lần truy vấn (tối đa 100 bản ghi), dẫn đến khả năng thiếu sót trong quá trình thu thập dữ liệu.

Mặc dù tồn tại giới hạn về số lượng log, kết quả phân tích tổng thể từ ELK Agent vẫn cho thấy mức độ tương đồng đáng kể với Splunk Agent về mặt cấu trúc truy vấn, cách tiếp cận phân tích và nhận định chung đối với hành vi truy cập. Điều này cho thấy tính nhất quán của kiến trúc đề xuất khi được áp dụng trên các nền tảng SIEM khác nhau.

d) Tổng kết

Kịch bản tấn công mạng được triển khai trên host mục tiêu chạy ứng dụng OWASP Juice Shop đã cho phép đánh giá khả năng của hệ thống trong việc phát hiện, truy vấn và phân tích các hành vi tấn công web dựa trên dữ liệu log thu thập từ hạ tầng mạng. Với đặc tính là một ứng dụng cố ý tồn tại nhiều lỗ hổng thuộc OWASP Top Ten, môi trường này phù hợp để mô phỏng các hoạt động thăm dò và khai thác phổ biến trong thực tế.

Kết quả thử nghiệm trên Splunk Agent cho thấy hệ thống có khả năng tạo truy vấn chính xác, truy xuất đầy đủ các trường thông tin cần thiết và hỗ trợ phân tích hành vi tấn công một cách có cấu trúc. Việc phân tách rõ ràng các yêu cầu truy cập thành hai pha—bao gồm pha truy cập thông thường và pha chứa các yêu cầu nguy hiểm—giúp làm nổi bật quá trình leo thang hành vi từ thăm dò sang tấn công. Các sự kiện có mức độ rủi ro cao được xác định và làm nổi bật, từ đó hỗ trợ việc tổng hợp ảnh hưởng của cuộc tấn công cũng như đề xuất các biện pháp khắc phục phù hợp.

Đối với ELK Agent, hệ thống cũng thể hiện khả năng sinh truy vấn DSL đầy đủ và truy xuất dữ liệu từ Elasticsearch để phục vụ phân tích. Tuy nhiên, trong lần thử nghiệm này, ELK Agent chỉ phân tích được các yêu cầu truy cập thông thường và chưa ghi nhận đầy đủ các yêu cầu mang dấu hiệu tấn công. Nguyên nhân được xác định là do giới hạn số lượng log được truy xuất trong mỗi lần lấy mẫu, dẫn đến khả năng bỏ sót một phần dữ liệu liên quan. Vấn đề này xuất phát từ cấu hình thu thập log, không phải từ logic xử lý hay khả năng phân tích của tác nhân.

Mặc dù tồn tại sự khác biệt nhất định về mức độ đầy đủ của dữ liệu phân tích, kết quả tổng thể cho thấy ELK Agent vẫn cung cấp các nhận định tương đồng với Splunk Agent về hành vi truy cập và đặc điểm lưu lượng mạng. Qua kịch bản này, hệ thống AI-Agent đã chứng minh được khả năng hỗ trợ điều tra tấn công mạng, từ khâu truy vấn dữ liệu đến phân tích và tổng hợp kết quả, đồng thời làm rõ các yếu tố ảnh hưởng đến chất lượng phân tích trong môi trường SIEM khác nhau.

4.3. Đánh giá

Dựa trên quá trình thiết kế, triển khai và thử nghiệm, hệ thống đã đạt được các mục tiêu chính đề ra trong nghiên cứu. Trước hết, một Module AI Core đã được xây dựng thành công, cho phép phân tích yêu cầu được biểu đạt bằng ngôn ngữ tự nhiên và chuyển đổi chúng thành một đối tượng truy vấn JSON chung. Đối tượng này đóng vai trò là lớp trung gian, giúp chuẩn hóa ý định truy vấn và loại bỏ sự phụ thuộc trực tiếp vào cú pháp của từng hệ thống SIEM.

Tiếp theo, kiến trúc Adapter Pattern đã được hiện thực hóa thông qua việc xây dựng các module chuyển đổi riêng biệt, bao gồm SplunkAdapter và ElasticsearchAdapter. Các adapter này chịu trách nhiệm ánh xạ cấu trúc truy vấn chung sang ngôn ngữ truy vấn đặc thù của từng nền tảng SIEM, qua đó bảo đảm tính linh hoạt và khả năng mở rộng của hệ thống khi tích hợp thêm các SIEM khác trong tương lai.

Bên cạnh đó, hệ thống đã tích hợp thành công cơ chế Retrieval-Augmented Generation (RAG) bằng cách sử dụng cơ sở dữ liệu vector. Cơ chế này cho phép các tác nhân AI truy xuất tri thức liên quan đến an ninh mạng và cú pháp truy vấn từ kho dữ liệu vector, từ đó nâng cao độ chính xác và tính phù hợp của các truy vấn được sinh ra, đồng thời tận dụng được tri thức và kinh nghiệm đã được tích lũy trước đó.

Cuối cùng, một giao diện người dùng đã được xây dựng nhằm hỗ trợ quá trình tương tác với hệ thống. Giao diện này cho phép người dùng nhập câu lệnh ngôn ngữ tự nhiên, lựa chọn nền tảng SIEM mục tiêu và xem kết quả truy vấn kèm theo phần giải thích. Việc kết hợp giữa khả năng truy vấn tự động và cơ chế giải thích giúp nâng cao tính minh bạch, khả năng sử dụng và giá trị ứng dụng thực tiễn của hệ thống trong bối cảnh giám sát và phân tích an ninh mạng.

PHẦN BA: KẾT LUẬN

1. Kết quả

Nghiên cứu này đã đề xuất và hiện thực thành công một kiến trúc hệ thống dựa trên mô hình đa tác nhân nhằm hỗ trợ việc chuyển đổi yêu cầu ngôn ngữ tự nhiên thành các truy vấn SIEM có thể thực thi trên nhiều nền tảng khác nhau. Kết quả đạt được cho thấy hệ thống có khả năng tự động hóa hiệu quả quá trình phân tích log an ninh, đồng thời giảm đáng kể sự phụ thuộc vào kiến thức chuyên sâu về cú pháp truy vấn đặc thù của từng hệ thống SIEM.

Trước hết, hệ thống đã xây dựng thành công AI Core Module có khả năng tiếp nhận, phân tích và diễn giải yêu cầu người dùng dưới dạng ngôn ngữ tự nhiên, sau đó chuyển đổi thành một đối tượng truy vấn JSON chung. Cấu trúc trung gian này đóng vai trò then chốt trong việc chuẩn hóa ý định truy vấn và tách biệt giai đoạn hiểu ngữ nghĩa khỏi giai đoạn thực thi, qua đó bảo đảm tính nhất quán và khả năng mở rộng của hệ thống.

Tiếp theo, kiến trúc Adapter Pattern được hiện thực hóa thông qua các module chuyển đổi riêng biệt cho từng nền tảng SIEM, cụ thể là Splunk và Elasticsearch. Các tác nhân chuyên biệt (Splunk Agent và ELK Agent) đã chứng minh khả năng sinh ra các câu truy vấn SPL và DSL chính xác, tuân thủ đầy đủ cú pháp và đặc thù dữ liệu của từng hệ thống. Kết quả thử nghiệm trên cả hai nền tảng cho thấy mức độ tương đồng cao về dữ liệu thu thập, chỉ báo xâm nhập (IOC) và kết luận phân tích, từ đó khẳng định tính khả chuyển và độc lập nền tảng của kiến trúc đề xuất.

Việc tích hợp cơ chế Retrieval-Augmented Generation (RAG) thông qua cơ sở dữ liệu vector Qdrant đã góp phần nâng cao chất lượng truy vấn được sinh ra. Thông qua Qdrant Search Tool, các tác nhân có khả năng truy xuất tri thức kinh nghiệm đã được lưu trữ dưới dạng vector nhúng, bao gồm các mẫu truy vấn hiệu quả, hành vi tấn công điển hình và cấu trúc sự kiện thường gặp. Kết quả cho thấy cơ chế này giúp cải thiện độ chính xác, giảm thiểu sai sót và tăng mức độ tin cậy của các truy vấn sinh tự động.

Thông qua các kịch bản thử nghiệm thực tế, bao gồm phân tích hành vi mã độc và điều tra tấn công mạng trên ứng dụng web, hệ thống đã thể hiện khả năng phát hiện, tổng hợp và phân tích chuỗi sự kiện một cách hiệu quả. Các báo cáo sinh ra không chỉ phản ánh chính xác dữ liệu log thu thập được mà còn cung cấp các nhận định có giá trị, bao gồm chuỗi hành vi, mức độ ảnh hưởng và các khuyến nghị ứng phó phù hợp. So sánh với các báo cáo phân tích của chuyên gia an ninh, kết quả thu được cho thấy mức độ tương đồng cao về nội dung và kết luận.

Tổng hợp lại, các kết quả đạt được khẳng định rằng kiến trúc hệ thống đề xuất có tính khả thi cao, đáp ứng tốt các mục tiêu đặt ra về khả năng hiểu ngôn ngữ tự nhiên, chuyển đổi truy vấn đa nền tảng, phân tích log an ninh và hỗ trợ ra quyết định. Đây là cơ sở quan trọng để tiếp tục mở rộng hệ thống trong các nghiên cứu tiếp theo, hướng tới việc hỗ trợ thêm nhiều nền tảng SIEM và các kịch bản an ninh phức tạp hơn trong môi trường triển khai thực tế.

2. Hạn chế của sản phẩm

Mặc dù đạt được các kết quả tích cực, hệ thống vẫn tồn tại một số hạn chế nhất định. Trước hết, chất lượng phân tích phụ thuộc đáng kể vào mức độ đầy đủ và chất lượng của dữ liệu log đầu vào. Trong các kịch bản mà số lượng log thu thập bị giới hạn, khả năng phát hiện đầy đủ các hành vi tấn công có thể bị ảnh hưởng.

Bên cạnh đó, hệ thống hiện mới tập trung hỗ trợ hai nền tảng SIEM phổ biến là Splunk và ELK Stack. Việc mở rộng hỗ trợ thêm các hệ thống SIEM khác vẫn cần được nghiên cứu và triển khai trong các giai đoạn tiếp theo.

Về mặt hiệu năng, việc sử dụng mô hình ngôn ngữ lớn và cơ chế RAG có thể làm gia tăng độ trễ trong một số tình huống xử lý phức tạp hoặc khi khối lượng truy vấn tăng cao. Điều này đòi hỏi các giải pháp tối ưu hóa trong triển khai thực tế.

Cuối cùng, xét về yếu tố tài chính, việc vận hành hệ thống có thể phát sinh chi phí đáng kể, đặc biệt khi sử dụng các dịch vụ mô hình ngôn ngữ lớn trên nền tảng đám mây, hạ tầng lưu trữ vector và các hệ thống SIEM thương mại. Chi phí này có thể trở thành

rào cản đối với các tổ chức nhỏ hoặc các môi trường triển khai thử nghiệm, và cần được cân nhắc kỹ lưỡng trong quá trình áp dụng thực tế.

3. Hướng phát triển

Trong quá trình nghiên cứu, dự án đã đạt được các mục tiêu cốt lõi đề ra, đặc biệt là việc xây dựng thành công một hệ thống có khả năng chuyển đổi ngôn ngữ tự nhiên sang truy vấn SIEM và tự động phân tích dữ liệu log. Tuy nhiên, so với kỳ vọng ban đầu, một số khía cạnh vẫn chưa được tối ưu hoàn toàn, chẳng hạn như khả năng bao phủ đầy đủ các kịch bản tấn công phức tạp hoặc tối ưu hóa chi phí và hiệu năng khi xử lý dữ liệu lớn.

Trong tương lai, hệ thống có thể được mở rộng theo nhiều hướng. Một hướng quan trọng là mở rộng hỗ trợ thêm các nền tảng SIEM khác, từ đó tăng khả năng tương thích và phạm vi ứng dụng của dự án. Việc bổ sung các adapter mới có thể được thực hiện mà không cần thay đổi đáng kể logic xử lý cốt lõi, nhờ kiến trúc tách lớp đã được thiết kế.

Bên cạnh đó, hệ thống có thể được cải tiến bằng cách nâng cao cơ chế thu thập log, tối ưu truy vấn và mở rộng kho tri thức trong cơ sở dữ liệu vector nhằm cải thiện độ chính xác và chiều sâu phân tích. Những hướng phát triển này không chỉ giúp hoàn thiện sản phẩm về mặt kỹ thuật mà còn tăng khả năng ứng dụng thực tiễn trong các hệ thống giám sát và phân tích an ninh mạng quy mô lớn.

TÀI LIỆU THAM KHẢO

[1] M. Cinque, D. Cotroneo, and A. Pecchia, "Challenges and Directions in Security Information and Event Management (SIEM)," in *Proceedings - 29th IEEE International Symposium on Software Reliability Engineering Workshops, ISSREW 2018*, Institute of Electrical and Electronics Engineers Inc., Nov. 2018, pp. 95–99. doi: 10.1109/ISSREW.2018.00-24.

[2] M. Sheeraz *et al.*, "Effective Security Monitoring Using Efficient SIEM Architecture," *Human-centric Computing and Information Sciences*, vol. 13, p. 17, 2023, doi: 10.22967/HCIS.2023.13.017.

[3] M. Hristov, M. Nenova, G. Iliev, and D. Avresky, "Integration of Splunk Enterprise SIEM for DDoS Attack Detection in IoT," in *2021 IEEE 20th International Symposium on Network Computing and Applications, NCA 2021*, Institute of Electrical and Electronics Engineers Inc., 2021. doi: 10.1109/NCA53618.2021.9685977.

[4] "A Comprehensive Study of Elastic Search," *Journal of Research in Science and Engineering*, vol. 4, no. 11, Nov. 2022, doi: 10.53469/jrse.2022.04(11).07.

[5] M. Kumar, "Big Data Analytics in Cybersecurity: Improving Threat Detection and Prevention." [Online]. Available: www.ijirct.org

[6] X. Huang *et al.*, "Understanding the planning of LLM agents: A survey," Feb. 2024, [Online]. Available: http://arxiv.org/abs/2402.02716

[7] OpenAI, "GPT-4 Technical Report."

[8] N. M. Gardazi, A. Daud, M. K. Malik, A. Bukhari, T. Alsahfi, and B. Alshemaimri, "BERT applications in natural language processing: a review," *Artif Intell Rev*, vol. 58, no. 6, Jun. 2025, doi: 10.1007/s10462-025-11162-5.

[9] K. Edemacu and X. Wu, "Privacy Preserving Prompt Engineering: A Survey, Supplemental Document," vol. 1, 2025, doi: 10.24432/C5XW20.

[10] J. Liu *et al.*, "M-A-P A Comprehensive Survey on Long Context Language Modeling."

[11] "Fine_tuning".

[12] R. Sapkota, K. I. Roumeliotis, and M. Karkee, "AI Agents vs. Agentic AI: A Conceptual Taxonomy, Applications and Challenges AI Agents Agentic AI," 2022. [Online]. Available: https://github.com/yoheinakajima/babyagi

[13] H. T. Tuyến and L. H. Hiệp, "NGHIÊN CỨU ỨNG DỤNG MỘT SỐ THUẬT TOÁN HỌC SÂU CHO BÀI TOÁN PHÁT HIỆN SỚM XÂM NHẬP BẤT

THƯỜNG TRONG MẠNG," *TNU Journal of Science and Technology*, vol. 228, no. 15, pp. 3–10, Aug. 2023, doi: 10.34238/tnu-jst.7494.

[14] V. H. H. Phạm Văn Thắng, "Xây dựng mô hình xử lý ngôn ngữ tự nhiên tiếng Việt cho truy vấn dữ liệu an ninh mạng," *Hội nghị Khoa học Quốc gia về Công nghệ Thông tin và Truyền thông (NICS) (2024)*, 2024.

[15] T. V. Đ. Ngô Quang Lợi, "Thiết kế và triển khai hệ thống tự động hóa phản ứng và báo cáo sự kiện an ninh mạng (soar) sử dụng học máy," *Tuyển tập các báo cáo khoa học, Học viện Kỹ thuật Quân sự (2022)*, 2022.

[16] G. de J. C. da Silva and C. B. Westphall, "A Survey of Large Language Models in Cybersecurity," Feb. 2024, [Online]. Available: http://arxiv.org/abs/2402.16968

[17] E. Jenniffer, "The Rise of Autonomous Security Operations Centers (AI-SOCs)."

[18] S. Reddy Mallreddy, Y. Vasa, and C. Author, "Natural Language Querying In Siem Systems: Bridging The Gap Between Security Analysts And Complex Data," 2023.