ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Phân tích và chấm điểm kịch bản rủi ro APT từ nhật ký mạng và hệ thống bằng các mô hình học máy

NGUYỄN VĂN THÁI
thai.nv215135@sis.hust.edu.vn

Chương trình đào tạo: Công nghệ Thông tin Việt - Nhật

Giảng viên hướng dẫn: TS. Vũ Thị Hương Giang _____

ThS. Nguyễn Mạnh Tuấn _____

Khoa: Khoa học máy tính

Trường: Công nghệ Thông tin và Truyền thông

HÀ NỘI, 07/2025

LỜI CẢM ƠN

Em xin trân trọng gửi lời cảm ơn sâu sắc nhất đến TS. Vũ Thị Hương Giang, người đã trực tiếp hướng dẫn em trong suốt quá trình thực hiện đồ án tốt nghiệp. Kể từ khi bắt đầu, cô không chỉ tận tình chỉ bảo, định hướng cho sự phát triển của đồ án mà còn là điểm tựa tinh thần vững chắc, giúp em vượt qua những rào cản tâm lý và nỗi lo lắng về khả năng hoàn thành. Nhờ sự động viên và hướng dẫn của cô, em đã có thể giải quyết hiệu quả các vấn đề phát sinh. Em cũng xin bày tỏ lòng biết ơn chân thành đến ThS. Nguyễn Mạnh Tuấn, giảng viên đồng hướng dẫn. Dù ở vai trò đồng hành, thầy đã đóng góp vô cùng quý báu bằng việc mang đến những góc nhìn đa chiều, giúp em hoàn thiện hơn các giải pháp cho vấn đề. Thời gian ba học kỳ làm việc cùng hai thầy cô không chỉ giúp em tích lũy kiến thức chuyên môn mà còn rèn luyện nhiều kỹ năng quan trọng trong công việc và phát triển bản thân theo hướng chuyên nghiệp hơn. Em xin kính chúc hai thầy cô luôn mạnh khỏe và thành công trong sự nghiệp.

Em cũng xin chân thành cảm ơn gia đình – những người luôn âm thầm ở bên, là nguồn động viên lớn lao, tạo điều kiện tốt nhất để em được học tập và phát triển bản thân trong môi trường Đại học Bách khoa Hà Nội. Bên cạnh đó, em xin gửi lời cảm ơn đến các anh chị, bạn bè đã luôn sát cánh, chia sẻ và hỗ trợ em trong những lúc khó khăn để em có thể hoàn thành tốt đồ án tốt nghiệp này.

Sinh viên thực hiện

(Ký và ghi rõ họ tên)

LỜI CAM KẾT

Họ và tên sinh viên:	Nguyễn Văn Thái
Điện thoại liên lạc:	0979561810
Email:	thai.nv215135@sis.hust.edu.vn
Lớp:	Việt-Nhật 03 - K66
Hệ đào tạo:	Cử nhân

Tôi – *Nguyễn Văn Thái* – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS. Vũ Thị Hương Giang*. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

Hà Nội, ngày tháng năm

Tác giả ĐATN

Nguyễn Văn Thái

TÓM TẮT NỘI DUNG ĐỒ ÁN

Trong bối cảnh an ninh mạng hiện tại, phần lớn các giải pháp cho rủi ro trong quá trình Devops tập trung vào việc phát hiện và ngăn chặn tấn công trên các hệ thống đang hoạt động. Tuy nhiên, cách tiếp cận này mang tính phản ứng và chỉ được kích hoạt sau khi cuộc tấn công đã xảy ra. Điều này có thể dẫn đến những thiệt hại đáng kể do thiếu sự chuẩn bị và phản ứng chậm trễ. Các hệ thống phát hiện tấn công truyền thống hoặc sử dụng mô hình học máy hiện tại thường chỉ có khả năng phân biệt các thông tin đơn giản về cuộc tấn công, chủ yếu là xác định có bị tấn công hay không. Chưa cung cấp được các thông tin về nhiều góc độ như giai đoạn bên tấn công, bên phòng thủ, .v.v

Hiện nay thì hệ thống RIDX được xây dựng nhằm đánh giá rủi ro của hệ thống trước và sau khi triển khai dựa trên mạng Bayes. Hệ thống có đánh giá rủi ro sẽ có đầu vào gồm 3 subnet về bên tấn công, phòng thủ và thông tin tài sản trong hệ thống. Hiện tại các thông tin về bên tấn công và bên phòng thủ được đánh giá thủ công dựa trên kiến thức của chuyên gia dẫn đến mất nhiều công sức. Để giải quyết vấn đề này thì trong phạm vi DATN đề xuất 2 biện pháp:

Một là ứng dụng học máy để phân tích và đánh giá dữ liệu mạng và log cảnh báo trên hệ thống mô phỏng tấn công APT. Biện pháp này được thực hiện bằng cách xây dựng các mô hình học máy như XGBoost, Random Forest,... để nhận diện và phát hiện các hành vi bất thường. Kết quả giúp tổng hợp và đưa ra các thông tin liên quan đến cuộc tấn công mà hệ thống mô phỏng đã ghi nhận.

Hai là xây dựng mô hình học máy để tổng hợp thông tin mô hình trên đã phân tích được thành điểm đánh giá cho subnet 1 và subnet 2 của mạng Bayes đánh giá rủi ro. Do dữ liệu phân tích không thể sử dụng trực tiếp, cần có một cơ chế ánh xạ phù hợp để biến đổi thông tin thu được thành định dạng đầu vào có thể xử lý.

Từ hai biện pháp trên, hệ thống trong đồ án sẽ xây dựng ba chức năng chính gồm: (i) Phát hiện tấn công trong quá trình hệ thống hoạt động, (ii) Phân tích và tổng hợp thông tin cuộc tấn công, (iii) Đánh giá điểm tự động làm đầu vào cho mạng Bayes đánh giá rủi ro.

Keysword: Advanced Persistent Threats (APT), Bayesian network

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Thuật ngữ	Ý nghĩa
ĐHBKHN	Đại học Bách khoa Hà Nội
API	Giao diện lập trình ứng dụng (Application Programming Interface)
APT	Advanced Persistent Threat
ATTT	An toàn thông tin
CNTT	Công nghệ thông tin
CPE	Định danh nền tảng chung (Common Platform Enumeration)
CVE	(Common Vulnerabilities and Exposures)
DATN	Đồ án tốt nghiệp
HTTP	Giao thức truyền tải

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1 Đặt vấn đề

Trong thập kỷ qua, mô hình phát triển CI/CD đã trở thành xương sống của quá trình chuyển đổi số, cho phép các tổ chức triển khai phần mềm liên tục với tốc độ chưa từng có. Tuy nhiên, sự phụ thuộc ngày càng lớn vào tự động hóa đã vô hình mở rộng bề mặt tấn công (attack surface) – từ kho lưu trữ mã nguồn, pipeline build/test đến cơ sở hạ tầng đám mây. Điều này tạo điều kiện cho Advanced Persistent Threats (APT) – những cuộc tấn công có chủ đích, đa giai đoạn – khai thác các lỗ hổng trong quy trình DevOps để xâm nhập sâu vào hệ thống.

Các phương pháp đánh giá rủi ro hiện tại như Behavior Profiling, Kill Chain Analysis, v.v. thường hoạt động theo cách tiếp cận phản ứng, chỉ kích hoạt sau khi cuộc tấn công đã xảy ra. Hệ thống IDS/IPS và SIEM hay nhiều hệ thống sử dụng học máy hiện tại chỉ cung cấp cảnh báo nhị phân (có/không tấn công) mà thiếu thông tin chi tiết về: Bối cảnh tấn công, tác động đa chiều, dự báo rủi ro động, v.v. Hạn chế này dẫn đến việc các tổ chức không thể chuẩn bị phòng thủ hiệu quả và thường phản ứng chậm trễ khi sự cố xảy ra.

Để khắc phục hạn chế đó, cần một hướng tiếp cận mới – đánh giá rủi ro một cách chủ động dựa trên dữ liệu thực tế từ hệ thống, đặc biệt là nhật ký mạng và host logs. Thay vì chỉ phát hiện tấn công, hệ thống cần phân tích hành vi tấn công, xác định các giai đoạn trong tiến trình APT, từ đó hình thành các kịch bản rủi ro cụ thể giúp tổ chức xây dựng chiến lược phòng thủ phù hợp.

Trước tình hình đó, các tổ chức ngày càng nhận thức rõ tầm quan trọng của việc đánh giá rủi ro bảo mật ngay từ giai đoạn thiết kế và triển khai hệ thống. Tuy nhiên, việc đánh giá này lại gặp nhiều thách thức khi dữ liệu thực tế từ các hệ thống sản xuất thường chứa thông tin nhạy cảm, không thể chia sẻ công khai để phân tích.

Với nhu cầu cần thiết như hiện tại thì hệ thống RIDX đã ra đời với mục đích giải quyết vấn đề bài toán đánh giá rủi ro trong quá trình Devops trước và sau khi triển khai một cách tự động và hiệu quả. RIDX là hệ thống cho phép người dùng có thể đánh giá rủi ro của hệ thống có thể xảy ra dựa vào kịch bản triển khai (Deployment Scenario), tự động ánh xạ tài sản với các lỗ hổng bảo mật đã biết (CVE) thông qua thông tin thiết bị và mối quan hệ giữa chúng. Dựa trên những dữ liệu này, hệ thống có thể mô phỏng các kịch bản tấn công tiềm năng và đánh giá rủi ro dựa trên mô hình mạng Bayes, gồm ba lớp subnet: bên tấn công (subnet 1), bên phòng thủ (subnet 2), và kịch bản triển khai (subnet 3).

Tuy nhiên, trong giai đoạn hiện tại, việc xác định và gán điểm đánh giá cho các subnet 1 và 2 vẫn chủ yếu dựa vào kiến thức chuyên gia và thao tác thủ công. Cách tiếp cận này không chỉ tiêu tốn nhiều thời gian và công sức, mà còn thiếu tính linh hoạt khi áp dụng cho các hệ thống thực tế với quy mô lớn và dữ liệu thay đổi liên tục. Điều này đặt ra nhu cầu cấp bách về một cơ chế đánh giá rủi ro tự động dựa trên dữ liệu thực tế, đặc biệt là từ các dấu vết tấn công như network flows và host logs.

Hiện nay, các hệ thống học máy đã được ứng dụng vào lĩnh vực an toàn thông tin nhằm phát hiện các cuộc tấn công mạng. Tuy nhiên, hầu hết các hệ thống này vẫn mang tính chất phản ứng, chỉ đưa ra cảnh báo khi tấn công đã xảy ra, và thường chỉ dừng lại ở mức phát hiện có hay không có tấn công. Điều này khiến các tổ chức khó có thể chủ động trong việc bảo vệ hệ thống. Vì vậy, cần có một hướng tiếp cận mới tập trung vào việc đánh giá rủi ro một cách chủ động, trước khi các sự cố xảy ra, từ đó xây dựng các chiến lược phòng thủ hiệu quả và giảm thiểu thiệt hại tiềm tàng. Các thông tin về quá trình tấn công sẽ cần được tổng hợp để phục vụ cho quá trình nghiên cứu, đưa ra các biện pháp giải quyết phù hợp.

Trong phạm vi đồ án sẽ giải quyết hai vấn đề quan trọng ở đây sẽ là: (i) Xây dựng các mô hình học máy có khả năng phát hiện tấn công hiệu quả thông qua việc kết hợp dữ liệu từ network flows và host logs. Điều này sẽ cung cấp cái nhìn toàn diện hơn về hành vi và giai đoạn của cuộc tấn công. (ii) Đánh giá tự động điểm cho kịch bản rủi ro dựa trên dữ liệu đã được các mô hình học máy phân tích. Điều này sẽ giảm đáng kể công sức đánh giá thủ công, tăng tính linh hoạt và khả năng mở rộng của RIDX. Đồ án sẽ tập trung giải quyết hai vấn đề trên sẽ giúp RIDX hoạt động hiệu quả hơn, cung cấp khả năng đánh giá rủi ro tự động và chi tiết, từ đó hỗ trợ các tổ chức xây dựng chiến lược phòng thủ tốt hơn trong môi trường DevOps đầy thách thức hiện nay.

1.2 Mục tiêu và phạm vi đề tài

Phần 1.1 đã đưa ra thách thức chung trong việc đánh giá rủi ro bảo mật trong môi trường CI/CD và DevOps, đặc biệt là những hạn chế của các phương pháp phản ứng hiện có. Cụ thể, việc xác định và gán điểm cho các subnet tấn công (subnet 1) và phòng thủ (subnet 2) vẫn phụ thuộc đáng kể vào kiến thức chuyên gia và quy trình thủ công, gây tốn thời gian, thiếu tính linh hoạt và khó mở rộng cho các hệ thống phức tạp. Ngoài ra cần cung cấp góc nhìn chi tiết về cuộc tấn công thông qua kịch bản rủi ro để xây dựng các chiến lược ứng phó.

Để khắc phục những hạn chế đó, đồ án này em đề xuất hướng tiếp cận nhằm nâng cao khả năng phân tích, đánh giá rủi ro tự động của hệ thống RIDX, dựa trên

phân tích dữ liệu thực tế từ các cuộc tấn công mạng. Cụ thể, đồ án tập trung vào các mục tiêu sau:

- Xây dựng mô hình học máy nhằm phát hiện giai đoạn tấn công APT. Mô hình khai thác dữ liệu đầu vào gồm network flows và host logs đồng thời tích hợp thông tin về vùng mạng (public, middle, private) trong kịch bản triển khai của tổ chức / doanh nghiệp. Kết quả của mô hình học máy cho thấy độ chính xác đạt trên 90
- Xây dựng module phân tích và trực quan hóa thông tin tấn công chi tiết bao gồm sự thay đổi của các giai đoạn tấn công, mức độ phòng thủ của hệ thống và phân theo từng loại đối tượng tấn công.
- Cuối cùng, đồ án phát triển cơ chế ánh xạ tự động từ kết quả phân tích thành thông số đầu vào cho mạng Bayesian trong RiDX, cho phép đánh giá rủi ro dựa trên dữ liệu mô phỏng tấn công mạng. Từ đó tổng hợp các đánh giá xây dựng kịch bản rủi ro cho cuộc tấn công.

Phạm vi đồ án tập trung vào các hệ thống doanh nghiệp sử dụng mô hình DevOps với kiến trúc mạng phân tầng. Các thử nghiệm được thực hiện trên ba bộ dữ liệu mô phỏng: Unraveled, DAPT-2020, và SCVIC-APT-2021, đại diện cho các kịch bản tấn công APT phổ biến trong môi trường doanh nghiệp.

1.3 Định hướng giải pháp

Từ các nhiệm vụ đã đưa ra ở phần 1.2 các định hướng giải pháp đưa ra sẽ đạt được ba mục tiêu đã trình bày.

Đầu tiên xây dựng các mô hình học máy cho việc phân tích thông tin của cuộc tấn công. Áp dụng các mô hình học máy như XGBoost, Random Forest để so sánh kết quả huấn luyện để đưa ra được mô hình tối ưu. Điểm đặc biệt của giải pháp là việc bổ sung thông tin ngữ cảnh vùng mạng (network zone context) vào feature vector, giúp mô hình hiểu rõ hơn về môi trường tấn công và giảm tỷ lệ nhầm lẫn giữa các giai đoạn tấn công. Phương pháp này dựa trên giả thuyết rằng hành vi tấn công có tính đặc trưng khác nhau tùy theo vùng mạng mà kẻ tấn công đang hoạt động. Ngoài ra còn mô hình được xây dựng kết hợp cả dữ liệu network flows và host logs nhằm cung cấp nhiều các cách tiếp cận khác nhau.

Hơn nữa với kết quả mà các mô hình phân tích ra sẽ cần được thống kê lại để có thông tin chi tiết về cuộc tấn công. Cung cấp góc nhìn về sự thay đổi của cuộc tấn công được thể hiện qua các biểu đồ về giai đoạn tấn công, mức độ phòng thủ theo thời gian. Đặc biết ở đây có sử dụng phương pháp đánh giá theo windows time. Nghĩa là sẽ thống kê giai đoạn tấn công trong khoảng thời gian nhất định và với

các bước nhảy thời gian sẽ lần lượt là các thống kê trong thời gian tiếp theo. Qua đây có thể thấy rõ được sự thay đổi của các giai đoạn tấn công theo sau các bước nhảy thời gian. Các thông tin đánh giá theo từng đối tượng tấn công sẽ được tổng hợp lại thành kịch bản rủi ro chứa thông tin về điểm đánh giá cung cấp góc nhìn khác cho quá trình tấn công.

Về việc tích hợp với ứng dụng thì hệ thống cần phải đưa ra đánh giá rủi ro ứng với các thông tin tấn công phân tích ra. Các kết quả mà các mô hình học máy phân tích ra không thể sử dụng trực tiếp cho mạng đánh giá rủi ro Bayes của hệ thống RIDX được. Vì thể giải pháp đưa ra ở đây là ánh xạ từ các dữ liệu phân tích ra để chuyển thành các điểm đánh giá của subnet 1 và subnet 2 cho mạng đánh giá rủi ro. Mô hình học máy RandomForest được áp dụng nhằm đóng vai trong là chuyên gia cho việc đánh giá từ kết quả phân tích thành điểm đánh giá.

Các đóng góp chính của đồ án sẽ bao gồm: (i) Xây dựng mô hình học máy phân tích dữ liệu mạng thu được từ cuộc tấn công với độ chính xác cao, (ii) Trực quan hoá các thông tin đã phân tích về thông tin mà mô hình phân tích và kịch bản rủi ro về cuộc tấn công, (iii) Xây dưng mô hình để ánh xạ từ các dữ liệu mà mô hình học máy phân tích thành điểm đánh giá cho subnet 1, 2. Từ đó tổng hợp kết quả phân tích cho kịch bản rủi ro và tích hợp vào mạng đánh giá rủi ro Bayes để đưa ra kết quả đánh giá.

1.4 Bố cục đồ án

Phần còn lại của báo cáo đồ án tốt nghiệp này được tổ chức như sau.

Chương 2 trình bày về khảo sát và phân tích yêu cầu của đồ án. Chương này sẽ trình bày về chức năng phân tích dữ liệu tấn công dựa trên dữ liệu thu được trên các hệ thống thực tế tích hợp vào hê thống RIDX. So sánh mô hình hiện tại với các mô hình khác và cung cấp thông tin liên quan đến các ứng dụng tương tự.

Chương 3 giới thiệu nền tảng lý thuyết và công nghệ sử dụng, bao gồm mô hình đánh giá rủi ro ATTT mạng Bayes, các công cụ hỗ trợ đánh giá an toàn thông tin và bộ dữ liệu áp dụng. Mục tiêu là cho người học hiểu sâu hơn về các công nghệ sử dụng.

Chương 4 trình bày về thiết kế kiến trúc hệ thống cho chức năng đánh giá rủi ro tấn công. Phần thiết kế sẽ trình bày chi tiết về kiến trúc microservices được triển khai trên hệ thống RIDX. Các cấu trúc hệ thống client, server và triển khai.

Chương 5 trình bày về đóng góp chính của đồ án trong việc giải quyết các vấn đề thực tế nhằm giải thích các phương pháp sử dụng và kết quả thu được. Trình bày chi tiết hơn về các đóng góp chính, các chắc năng tích hợp vào hệ thống RIDX,

cách cải thiện các mô hình học máy trong việc phát hiện tấn công.

Chương 6 trình bày kết luận và định hướng phát triển trong tương lai cải thiện hệ thống hiện tại.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

2.1 Khảo sát hiện trạng

2.1.1 Khảo sát các hệ thống đánh giá rủi ro

Trong lĩnh vực an toàn thông tin, nhiều framework và công cụ đã được phát triển nhằm phục vụ việc đánh giá và quản lý rủi ro. Mỗi hệ thống đều có phương pháp luận, phạm vi áp dụng và mức độ phức tạp riêng, phù hợp với từng mục tiêu sử dụng cụ thể như phát hiện mối đe dọa APT, định lượng rủi ro, hay hỗ trợ trực quan hóa thông tin.

Bảng 1 dưới đây trình bày sự so sánh giữa một số framework và công cụ bảo mật phổ biến, dựa trên các tiêu chí như loại hình, phương pháp luận, khả năng phát hiện APT, khả năng định lượng rủi ro và độ phức tạp trong triển khai.

Ứng Dụng/Framework	**Loại Bình**	**Phương Pháp Luận**	**Phát Hiện APT**	**Định Lượng Rủi Ro**	**Độ Phức Tạp**
MITRE ATT&CK Framework	Threat Modeling	TTPs-based Analysis	Có	Một phần	Cao
ART (APT Risk Tool)	Risk Assessment	4-Step Process	Có	Có	Thấp
NIST RMF	Risk Management	6-Step Lifecycle	Một phần	Có	Cao
TARA (MITRE)	Threat Assessment	AV/CM Catalogs	Có	Có	Cao

Bảng 2.1: So sánh các Framework và Công cụ Bảo mật

MITRE ATT&CK Framework: Đây là một công cụ mô hình hóa mối đe dọa (Threat Modeling) nổi bật, sử dụng phương pháp phân tích dựa trên kỹ thuật – chiến thuật – quy trình (TTPs-based Analysis). Framework này có khả năng phát hiện APT (Advanced Persistent Threats) một cách hiệu quả nhưng chỉ định lượng rủi ro ở mức một phần. Tuy nhiên, độ phức tạp trong triển khai cao, đòi hỏi người dùng có kiến thức chuyên sâu.

ART (APT Risk Tool): Đây là một công cụ đánh giá rủi ro (Risk Assessment) đơn giản với quy trình 4 bước, có khả năng phát hiện APT và định lượng rủi ro đầy đủ. Độ phức tạp thấp, dễ áp dụng cho các tổ chức nhỏ hoặc các nhóm bảo mật cần đánh giá nhanh.

NIST RMF (Risk Management Framework):Là một framework quản lý rủi ro

tổng thể, hoạt động theo vòng đời 6 bước. Nó có khả năng định lượng rủi ro và hỗ trợ một phần trong việc phát hiện APT. Tuy nhiên, độ phức tạp cao, phù hợp với tổ chức lớn hoặc trong các hệ thống cần tuân thủ tiêu chuẩn quốc gia.

TARA (MITRE):Là công cụ đánh giá mối đe dọa (Threat Assessment) của MITRE, sử dụng các danh mục AV/CM để xác định và lượng hóa rủi ro. TARA có khả năng phát hiện APT và định lượng rủi ro đầy đủ, nhưng phức tạp trong triển khai, đòi hỏi phải có sự hiểu biết sâu về hệ thống mục tiêu.

Tóm lại, việc lựa chọn công cụ phù hợp phụ thuộc vào mục tiêu sử dụng, nguồn lực kỹ thuật, và mức độ chi tiết cần thiết trong đánh giá rủi ro của từng tổ chức.

2.1.2 Khảo sát các giải pháp phát hiện tấn công hiện tại

Việc phát hiện và phân loại các cuộc tấn công mạng là một thành phần quan trọng trong hệ thống phòng thủ an ninh mạng hiện đại. Nhiều hệ thống và bộ dữ liệu đã được phát triển nhằm phục vụ mục đích huấn luyện, đánh giá và triển khai các mô hình học máy phát hiện tấn công hiệu quả.

Các hệ thống này có sự khác biệt rõ rệt về nguồn dữ liệu đầu vào, mức độ chi tiết của đầu ra cũng như hiệu quả đánh giá tương ứng. Một số sử dụng dữ liệu từ host logs kết hợp với network flows, trong khi phần lớn tập trung vào luồng mạng (network flows). Đầu ra của các hệ thống có thể bao gồm các thông tin như giai đoạn tấn công, hành động cụ thể, hay nhãn phân loại theo mức độ tấn công được thể hiện trong bảng dưới.

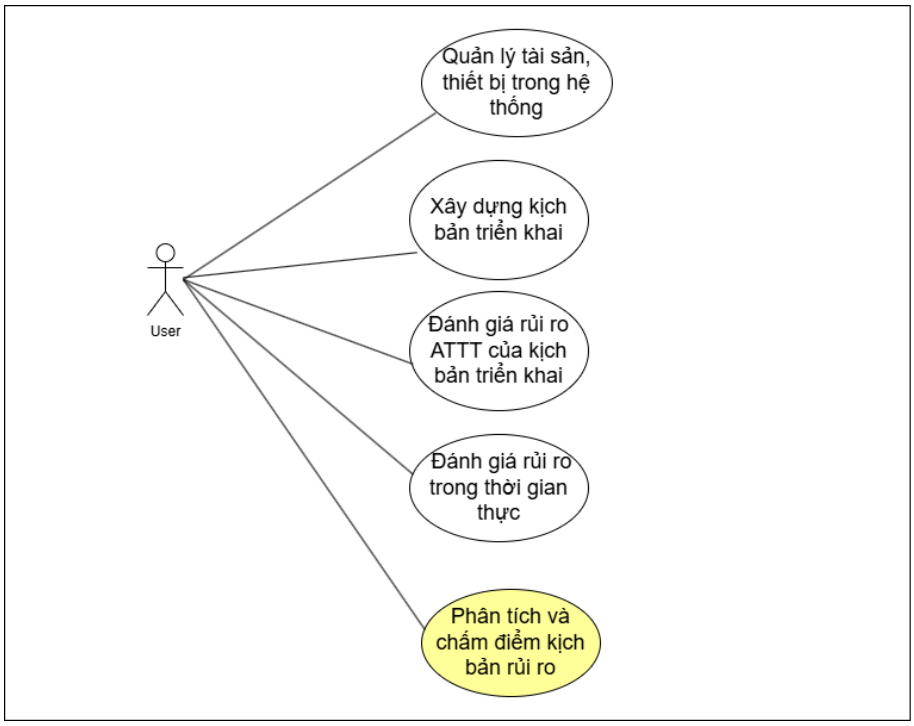
Hệ thống/bộ dữ liệu	**Đầu vào**	**Đầu ra chi tiết**	**Đánh giá kết quả ứng với đầu ra**
Unraveled	Host logs, Network flows	Stage, Signature, Defender response, Activity	Thấp
Dapt2020	Host logs, Network flows	Stage, Action	Cao
Scvic-APT-2021	Networks flows	Stage	Cao
CIC-Collection-ML&DL	Networks flows	Action: {DDoS, Botnet, Bruteforce, Infiltration, ...}	Trung bình
NSL KDD	Networks flows	Label: {Normal (bình thường), Attack (tấn công)}	Cao

Bảng 2.2: So sánh mô hình xây dựng với các bộ dữ liệu mô phỏng tấn công

2.2 Tổng quan chức năng

2.2.1 Biểu đồ use case tổng quát

Hệ thống RIDX được thiết kế và phát triển nhằm mục tiêu đánh giá rủi ro an toàn thông tin (ATTT) của hệ thống dựa trên các kịch bản triển khai cụ thể. Trước khi triển khai, hệ thống tiến hành đánh giá rủi ro thông qua phân tích các thông tin liên quan đến tài sản và cấu hình trong kịch bản triển khai. Sau khi triển khai, dữ liệu thực tế thu thập được trong quá trình vận hành sẽ tiếp tục được sử dụng làm đầu vào để cập nhật và đánh giá rủi ro thực tế. Các chức năng chính của hệ thống được thể hiện trong sơ đồ use case tổng quan ở Hình 2.1.

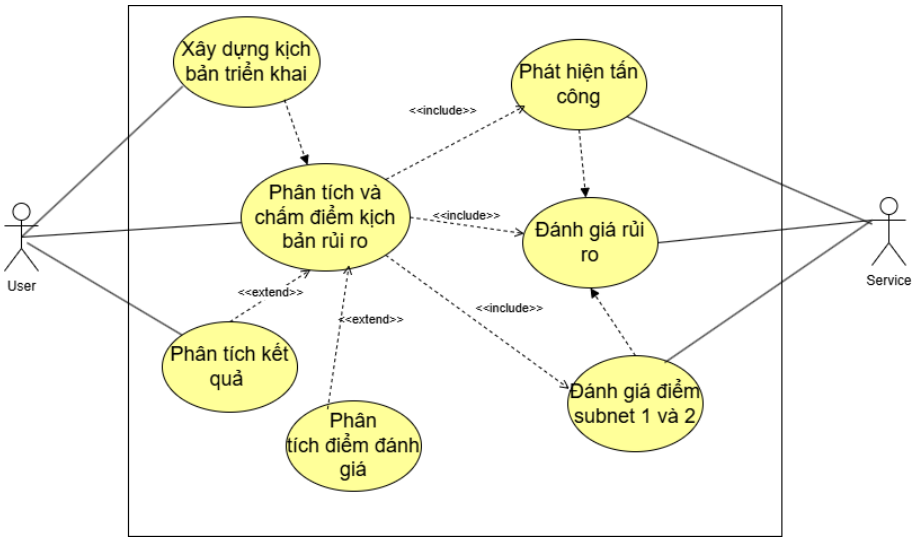


Hình 2.1: Biểu đồ usecase tổng quan chức năng

- Quản lý tài sản, thiết bị trong hệ thống: Hệ thống RIDX có chức năng quả lý các tài sản trong hệ thống với các thông tin về CVE, CPE của từng thiết bị. Các tài sản sẽ được để phục vụ cho chức năng xây dựng kịch bản triển khai.
- Xây dựng kịch bản triển khai: gồm thôn tin của các tài sản trong hệ thống, các mối quan hệ giữa các tài sản. Từ đây sẽ có kịch bản về luồng tấn công và biện pháp phòng thủ của hệ thống. Phục vụ cho việc đánh giá rủi ro cho hệ thống dựa trên các thông tin đó.
- Đánh giá rủi ro ATTT của kịch bản triển khai: Dựa trên kịch bản triển khai chứ thông tin về các tài sản, CPE, CVE làm đầu vào cho mạng đánh giá rủi ro Bayes. Từ đó đánh giá được tỷ lệ xảy ra rủi ro và mức độ rủi ro để phòng chống tấn công.

- Đánh giá rủi ro trong thời gian thực: Đánh giá quá trình rủi ro trong quá trình vận hành của hệ thống. Dựa vào dữ liệu thực tế đưa ra đánh giá, phát hiện kịp thời để ngăn chặn các hành vi bất thường.
- Phân tích và chấm điểm kịch bản rủi ro: Dựa vào dữ liệu quá khứ tại thời điểm nhất định cũng có vai trò lớn trong việc tìm ra biện pháp, điểm yếu của hệ thống nhằm nâng cao chất lượng hệ thống khi đưa vào triển khai. Ở đây sẽ thực hiện phân tích và đánh giá điểm cho cuộc tấn công để đánh giá rủi ro. Trong phần bên dưới sẽ trình bày chi tiết hơn về chứ năng này.

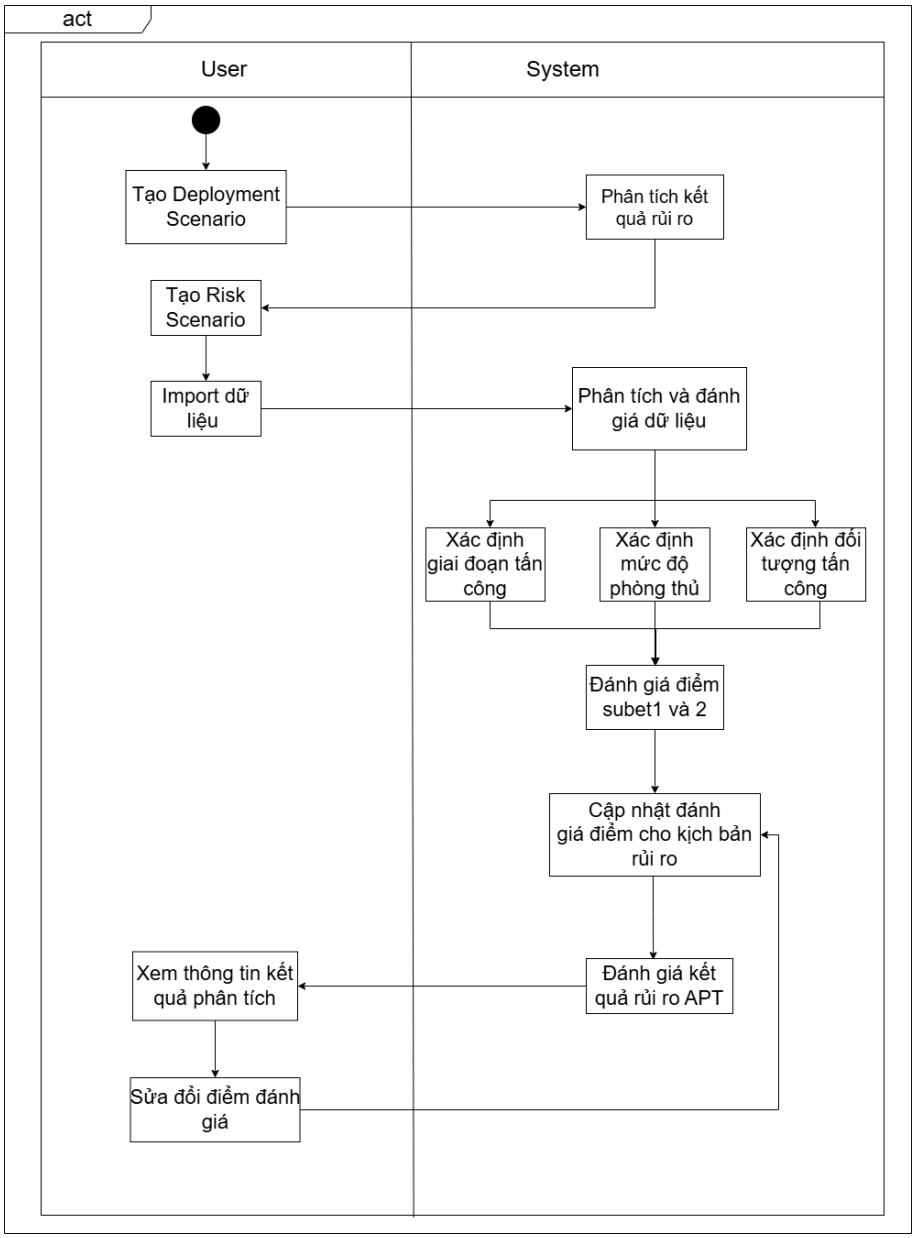
2.2.2 Biểu đồ use case phân rã chức năng phân tích chi tiết thông tin tấn công



Hình 2.2: Phân rã usecase phân tích và chấm điểm kịch bản rủi ro

Hình 2.2 là biểu dồ phân rã cho chức năng phân tích và chấm điểm kịch bản rủi ro. Ở đây người dùng sẽ cần phải xây dựng kịch bản triển khai trước với thông tin của các tài sản, mối quan hệ, CPE, CVE, v.v. Sau khi xây dựng kịch bản và đánh giá rủi ro trên đó thì sẽ đến bước phân tích thông tin tấn công. Người dùng sẽ truyền gửi lên file dữ liệu network flows của hệ thống. Các dữ liệu này sẽ được truyền xuống Core service thông qua socket api được xây dựng. Tại Core service sẽ bắt đầu phân tích và đưa ra các thông tin mà các mô hình học máy phát hiện trong dữ liệu network flows. Các thông tin sẽ được tổng hợp lại và truyền lại về phía người dùng. Đối với chức nằng đánh giá điểm thì cho phép người dùng có thể sửa lại điểm và gửi lại xuống để tính toán lại kết quả.

2.2.3 Quy trình nghiệp vụ



Hình 2.3: Quy trình nghiệp vụ chức năng phân tích và chấm điểm kịch bản rủi ro

Hình 2.3 mô tả luồng thực hiện cho quy trình phân tích chi tiết thông tin của cuộc tấn công APT. Sau khi đánh giá rủi ro của hệ thống thông qua deployment scenario thì kết quả sẽ được lưu lại phục vụ cho phần đánh giá tiếp theo khi mà hệ thống đã đi vào triển khai. Sau khi chọn hệ thống thích hợp thì người dùng sẽ upload dữ liệu network flows lên để phân tích. Dữ liệu sẽ được truyền đến hệ thống để phát hiện các thông tin về tấn công, phòng thủ và đối tượng tấn công. Từ đấy đưa ra điểm đánh giá cho subnet 1 và subnet2. Sau khi có điểm đánh giá thì kết quả sẽ là đầu vào cho mạng đánh giá rủi ro Bayes đánh giá và đưa ra kết quả về mức độ rủi ro cũng như tỷ lệ xảy ra rủi ro. Kết quả bao gồm về thông tin tấn công, kết quả đánh giá rủi ro sẽ được hiển thị ra màn hình giúp cho người dùng có thể dễ

dàng quan sát. Nếu người dùng muốn sửa đổi điểm thì điểm của subnet1 và subnet 2, điểm đó sẽ được truyền lại cho mạng đánh giá rủi ro và trả về kết quả tương ứng.

2.3 Đặc tả chức năng

2.3.1 Đặc tả use case đánh giá điểm risk scenario

Tên usecase	**UC001**		**Đánh giá điểm risk scenario**
Tác nhân	Người dùng, hệ thống		
Tiền điều kiện	Người dùng đã thực hiện phân tích dữ liệu tấn công		
Luồng sự kiện chính	**STT**	**Thực hiện bởi**	**Hành động**
	1	Người dùng	Chọn đối tượng muốn xem đánh giá điểm risk scenario
	2	Hệ thống	Đánh giá điểm và kết quả đánh giá rủi ro
	3	Hệ thống	Màn hình hiển thị kết quả đánh giá điểm và kết quả đánh giá rủi ro
	4	Người dùng	Lựa chọn chế độ sửa điểm đánh giá
	5	Hệ thống	Chuyển sang chế độ đánh giá điểm
	6	Người dùng	Đánh giá điểm mới và thực hiện gửi điểm đánh giá
	7	Hệ thống	Xử lý và trả về kết quả tương ứng với điểm đã sửa đổi
Luồng sự kiện thay thế	4a	Người dùng	Quay lại và lựa chọn xem kết quả của đối tượng khác
	5a	Hệ thống	Hiển ra điểm đánh giá và kết quả đánh giá rủi ro tương ứng
Hậu điều kiện	Không có		

Bảng 2.3: Bảng đặc tả usecase đánh giá điểm risk scenario

2.3.2 Đặc tả use case phân tích dữ liệu tấn công

Tên usecase	**UC002**		**Phân tích dữ liệu tấn công**
Tác nhân	Người dùng, hệ thống		
Tiền điều kiện	Người dùng đã tạo kịch bản rủi ro.		
Luồng sự kiện chính	**STT**	**Thực hiện bởi**	**Hành động**
	1	Người dùng	Lựa chọn Risk Scenario đã được tạo và chọn chức năng phân tích.
	2	Người dùng	Upload file dữ liệu network flows và ấn submit.
	3	Hệ thống	Xử lý dữ liệu nhận được, sử dụng các mô hình học máy đã xây dựng để đưa ra thông tin tấn công.
	4	Hệ thống	Hiển thị kết quả phân tích qua các biểu đồ thống kê.
	5	Người dùng	Upload dữ liệu host logs và ấn submit.
	6	Hệ thống	Hiển thị kết quả phân tích rủi ro trên các tài sản có trong host logs.
	7	Người dùng	Lựa chọn các windows time và step để thống kê kết quả rồi submit.
	8	Hệ thống	Xử lý và đưa ra biểu đồ thống kê các giai đoạn tấn công theo windows time.
Luồng sự kiện thay thế	2a	Người dùng	Người dùng upload dữ liệu log hoặc chọn windows time và step trước.
	3a	Hệ thống	Đưa ra thông báo cần nhập dữ liệu network flows trước.
Hậu điều kiện	Không có		

Bảng 2.4: Bảng đặc tả usecase phân tích chi tiết thông tin tấn công

2.3.3 Đặc tả use case tạo Risk scenario

Tên usecase	**UC003**		**Quản lý Risk scenario**
Tác nhân	Người dùng, hệ thống		
Tiền điều kiện	Người dùng đã tạo kịch bản triển khai cho hệ thống		
Luồng sự kiện chính	**STT**	**Thực hiện bởi**	**Hành động**
	1	Người dùng	Chọn chức năng risk scenario
	2	Hệ thống	Hiển thị màn hình ở trang risk scenario
	3	Người dùng	Chọn tạo mới risk scenario
	4	Hệ thống	Hiển thị màn hình tạo mới với thông tin như tên, mô tả, các kịch bản triển khai
	5	Người dùng	Điền thông tin và chọn mô hình sử dụng cho kịch bản rủi ro rồi submit
	6	Hệ thống	Xử lý thông tin và tạo mới risk scenario
	7	Hệ thống	Trả về thông báo tạo thành công và hiển thị danh sách risk scenario mới
Luồng sự kiện thay thế	3a	Người dùng	Lựa chọn các kịch bản rủi ro và xem chi tiết
	4a	Hệ thống	Hiển thị các kịch bản triển khai có chung risk scenario và mô hình theo bộ dữ liệu
	5a	Người dùng	Lựa chọn chức năng như xem kết quả hoặc đánh giá dữ liệu thực tế
	6a	Hệ thống	Hiển thị màn hình chức năng theo yêu cầu người dùng
Hậu điều kiện	Không có		

Bảng 2.5: Bảng đặc tả usecase quản lý Risk scenario

2.4 Yêu cầu phi chức năng

Để đảm bảo hoạt động ổn định cho hệ thống và cho người dùng thì cũng cần phải đảm bao các yêu cầu phi chức năng. Dưới đây là các yêu cầu phi chức năng cho hệ thống RIDX:

- Khả năng mở rộng: Hệ thống RIDX được xây dựng theo kiến trúc microservices, bao gồm nhiều dịch vụ nhỏ độc lập. Các dịch vụ này giao tiếp với nhau thông qua API Gateway, giúp việc phát triển và mở rộng hệ thống trở nên linh hoạt và dễ dàng hơn, do từng dịch vụ có thể được triển khai, nâng cấp hoặc mở rộng riêng biệt mà không ảnh hưởng đến toàn bộ hệ thống.
- Hiệu năng: Hệ thống cần đảm bảo các yêu cầu về hiệu năng, tốc độ phản hồi của các xử lý. Với các tình năng tính toán phức tạp và cần nhiều tài nguyên thì cần đảm bảo việc xử lý về tính toán. Các công thức tính toàn phức tạp của mạng Bayes và áp dụng các mô hình học máy vào hệ thống có kiến trúc lớn cần đảm bảo kích thước của các file được tối ưu. Cần có cơ chế lưu lại dữ liệu đang chạy và tái sử dụng tránh việc phải tải lại trang hoặc tính toán lại liên tục.
- Bảo mật: Cần có cơ chế bảo mật, xác thực người dùng. Cơ chế xác thực, phân quyền, mã hoá thông tin người dùng thành các token để đảm bảo không bị lộ, rò rỉ thông tin.
- Giao diện người dùng: Giao diện được thiết kế đơn giản, trực quan và dễ sử dụng. Hệ thống cung cấp thông tin phù hợp với từng vai trò người dùng, đặc biệt là các đối tượng như DevOps và System Engineer, giúp họ dễ dàng theo dõi và quản lý hệ thống hiệu quả. Các biểu đồ cần phải hiển thị thêm chi tiết các thông tin khi mà trỏ vào.

CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG

3.1 Nền tảng lý thuyết

Trong chương trước đã trình bày về chức năng chính của hệ thống RIDX hiện tại và các chắc năng được tích hợp thêm vào hệ thống. Trong chương này sẽ trình bày về cơ sở lý thuyết và công nghệ sử dụng được sử dụng để xây dựng hệ thống. Hệ thống RIDX được xây dựng với mục đích là giúp cho người dùng có thể đánh giá được xác xuất xảy ra rủi ro của hệ thống thông qua kịch bản triển khai. Dưới đây là cơ sở lý thuyết để xây dựng chức năng đánh giá rủi ro tấn công APT trong quá trình Devops.

3.1.1 Kịch bản triển khai

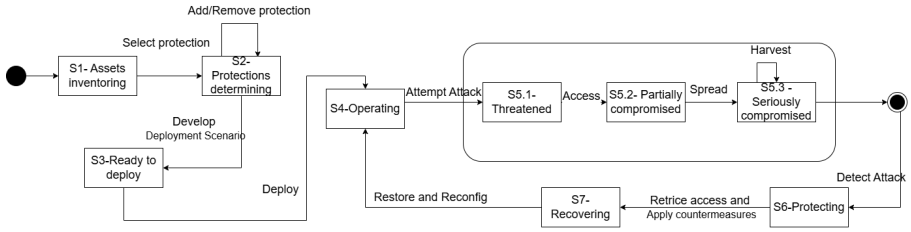
Trong quá tình trước khi triển khai của hệ thống, thông tin của thiết kế của hệ thống như tài sản, mỗi quan hệ giữa các tài sản, v.v. là thông tin của kịch bản triển khai đã được trong nghiên cứu [1]. Kịch bản triển khai sẽ bao gồm:

- Thông tin của thiết bị trong hệ thống: Mỗi thiết bị sẽ chứa thông tin chi tiết ví dụ cấu hình. Từ đó sẽ ánh xạ được CPE và CVE phù hợp với cấu hình máy.
- Sơ đồ hệ thống: Thể hiện mối quan hệ giữa các tài sản, kết nối giữa các thiết bị.
- Thông tin về biện pháp phòng thủ: Chứa thông tin của thiết bị sẽ bảo vệ hệ thống khỏi tấn công có CVE tương ứng.
- Rủi ro có trong hệ thống: Chứa thông tin về con đường mà kẻ tấn công có thể thực hiện để tấn công vào hệ thống. Giả lập việc bị tấn công để đự đoán rủi ro.

Với các thông tin chi tiết về hệ thống dự định sẽ xây dựng thì có thể đánh giá rủi ro xảy ra của hệ thống thông qua mạng đánh giá rủi ro Bayes. Hệ thống sẽ gồm có các trạng thái: giai đoạn phân tích yêu cầu, triển khai và vận hành. Kịch bản rủi ro sẽ bắt đầu từ giai đoạn phân tích yêu cầu. Sau khi bước vào các giai đoạn tiếp theo mạng Bayes tĩnh sẽ được mở rộng thành mạng Bayes động đánh giá rủi ro của hệ thống theo thời gian thực.

3.1.2 Máy chuyển trạng thái của hệ thống

Trong nghiên cứu [1] trình bày về chín trạng thái hệ thống trong suốt vòng đời DevOps. Các trạng thái hệ thống chia thành hai giai đoạn chính là trước khi đưa vào triển khai vào sau khi đã vận hành hệ thống. Chi tiết về các trạng thái của hệ thống được biểu diễn như hình 3.1.

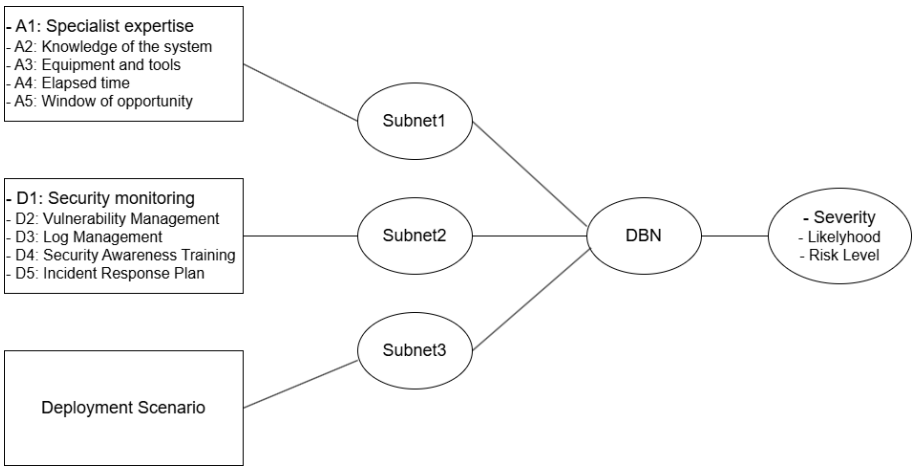


Hình 3.1: Máy chuyển trạng thái hệ thống

Trong giai đoạn tiền triển khai hệ thống, quy trình vận hành bắt đầu từ S1 - Assets Inventoring và tiếp tục đến S3 - Ready to Deploy. Ở giai đoạn này, hệ thống tiến hành đánh giá rủi ro và xác định các biện pháp bảo vệ dựa trên kịch bản triển khai đã được thiết lập. Việc lựa chọn và điều chỉnh các biện pháp bảo vệ được thực hiện thông qua bước S2 - Protections Determining. Sau khi hệ thống được triển khai (Deploy), nó chuyển sang giai đoạn vận hành (S4 - Operating). Trong giai đoạn này, nếu phát sinh các cuộc tấn công, hệ thống sẽ chuyển sang trạng thái bị đe dọa hoặc bị xâm nhập sẽ nằm từ trạng thái S5.1 đến S5.3. Trong trường hợp tấn công được phát hiện, hệ thống sẽ kích hoạt cơ chế phòng thủ từ S6 đến S7.

3.1.3 Mạng Bayes

Mạng Bayes được sử dụng trong hệ thống RIDX nhằm đánh giá rủi ro bảo mật của hệ thống được thiết kế với ba nhóm đầu vào chính: thông tin về bên tấn công (A1–A5), bên phòng thủ (D1–D5), và kịch bản triển khai. Cấu trúc chi tiết của các subnet này đã được trình bày trong nghiên cứu [1]. Hình 3.2 minh họa kiến trúc đầu vào mạng Bayes đánh giá rủi ro mới nhất đang được sử dụng trong hệ thống.



Hình 3.2: Mạng Bayes đánh giá rủi ro

Đầu vào cho mạng Bayes gồm có 3 thành phần chính là được biểu diễn như hình 3.2. Chi tiết về ý nghĩa đánh giá các tiêu chí đầu vào đã được xác định trong nghiên cứu [1]. Bảng 3.1 trình bày về các đầu vào bên tấn công A1-A5 và bảng 3.2 trình

bày về đầu vào bên phòng thủ từ D1-D5. Với 2 subnet trình bày ở dưới kết hợp với kịch bản triển khai sẽ là đầu vào cho mạng đánh giá rủi ro tấn công APT.

	Factor	**Ý nghĩa**
A1	Specialist Expertise	Mô tả mức độ chuyên môn và kinh nghiệm của kẻ tấn công.
A2	Knowledge of the System	Phản ánh mức độ thông tin mà kẻ tấn công có về hệ thống mục tiêu.
A3	Equipment and Tools	Mức độ trang bị và công cụ hỗ trợ mà kẻ tấn công có thể tiếp cận.
A4	Elapsed Time	Thời gian cần thiết để thực hiện cuộc tấn công thành công.
A5	Window of Opportunity	Khoảng thời gian mà lỗ hổng tồn tại hoặc hệ thống dễ bị tấn công.

Bảng 3.1: Giải thích các yếu tố về subnet 1

Factor		**Ý nghĩa**
A1	Security Monitoring	Đánh giá mức độ giám sát hệ thống, bao gồm khả năng phát hiện mối đe dọa theo thời gian thực.
A2	Vulnerability Management	Phản ánh năng lực phát hiện và xử lý các điểm yếu trong hệ thống.
A3	Log Management	Mức độ lưu trữ, giám sát và phân tích log để phát hiện hành vi bất thường.
A4	Security Awareness Training	Đánh giá hiệu quả của các chương trình đào tạo nhân viên về an toàn thông tin.
A5	Incident Response Plan	Mức độ sẵn sàng và hiệu quả của tổ chức trong việc đối phó với sự cố an ninh.

Bảng 3.2: Giải thích các yếu tố về subnet 2

3.1.4 Kịch bản rủi ro

Khi hệ thống được đưa vào triển khai, trong quá trình vận hành sẽ thu thập được các dữ liệu của hệ thống cụ thể là dữ liệu về network flows và host logs. Khi hệ thống bị tấn công sẽ được thu được các tín hiệu bất thường của đối tượng tấn công. Kịch bản rủi ro có thể coi là bản tóm gọn diễn tả quá trình tấn công vào hệ thống với các đánh giá về bên tấn công và bên phòng thủ.

TL	SS	Activities	A1	A2	A3	A4	A5	D1	D2	D3	D4	D5	T1
w5.7 to w6.3	S4 S5.1	A: scans ports & apps. on production → network D: monitors traffics	3	2	2	3	4	7	5	7	6	8	2
w6.4	S5.1 S5.2	A: brute-forces SSH credentials; vuls.: → CVE-2018-15473, CVE-2016-0777 D: monitors traffics	4	4	4	3	5	7	5	7	6	8	2
w6.5	S5.2	A: conducts network scans; D: detects brute-force attempts	4	4	4	4	5	7	6	8	6	8	2
w6.6	S5.2 →S6,→S7,→S4	A: gains no valuable info; D: blocks A's IP, reset passwords	4	2	4	4	3	7	6	7	6	8	2

Bảng 3.3: Bảng kịch bản rủi ro của đối tượng AA trong bộ dữ liệu Unraveled

Bảng trên là kịch bản rủi ro của AA trong bộ dữ liệu Unraveled được trình bày trong nghiên cứu [1]. Với từng bước chuyển trạng thái của hệ thống thì tại các mốc thời điểm sẽ có điểm đánh giá, các điểm này sẽ là đầu vào cho mạng đánh giá rủi ro Bayes để xác định rủi ro gây ra tại từng thời điểm.

3.2 Mô hình học máy

3.2.1 Giới thiệu

Mô hình học máy (machine learning model) [2] là hệ thống toán học hoặc thuật toán sử dụng dữ liệu đầu vào để học hỏi và đưa ra dự đoán mà không cần phải lập trình chi tiết từng bước. Thay vì dựa vào các quy tắc cố định, mô hình học máy tự động xác định các mẫu và mối quan hệ ẩn trong dữ liệu để giải quyết các bài toán thực tiễn như phân loại, dự báo, nhận diện hình ảnh, phát hiện gian lận, v.v. Mô hình học máy có thể được huấn luyện bằng dữ liệu đã gắn nhãn (học có giám sát) hoặc không gắn nhãn (học không giám sát). Trong chức năng xây dựng thêm cho hệ thống sẽ sử dụng hai mô hình chính là XGBoot và Random Forest.

3.2.2 Mô hình XGBoot

XGBoost (Extreme Gradient Boosting) là một thuật toán học máy mạnh mẽ, dựa trên nguyên lý boosting, cụ thể là gradient boosting. XGBoost nổi bật nhờ tốc độ huấn luyện nhanh, khả năng xử lý song song, tối ưu hóa bộ nhớ và cơ chế chính quy hóa để hạn chế overfitting.

- XGBoost xây dựng một chuỗi các cây quyết định (decision trees) theo từng bước tuần tự. Mỗi cây mới được huấn luyện nhằm sửa lỗi (residual) còn lại của mô hình trước đó, nhờ đó dần cải thiện độ chính xác tổng thể.
- Thuật toán sử dụng hàm mất mát (loss function) để đo lường sai số, và tối ưu hóa hàm này bằng kỹ thuật gradient descent.
- XGBoost tích hợp điều chuẩn (L1, L2 regularization) vào hàm mục tiêu để

kiểm soát độ phức tạp của mô hình, giảm nguy cơ quá khớp.

- Hỗ trợ xử lý dữ liệu lớn, dữ liệu thưa (sparse data), tự động nhận diện và xử lý giá trị thiếu.

3.2.3 Mô hình Random Forest

Random Forest là một thuật toán học máy thuộc nhóm học có giám sát (supervised learning), được xây dựng dựa trên phương pháp học tổ hợp (ensemble learning), cụ thể là kỹ thuật bagging (bootstrap aggregating).

- Tạo ra nhiều cây quyết định độc lập bằng cách lấy mẫu ngẫu nhiên có hoàn lại (bootstrap) từ tập dữ liệu gốc.
- Tại mỗi nút chia của cây, chỉ một tập con ngẫu nhiên các đặc trưng được chọn để quyết định điểm chia, giúp các cây khác biệt và giảm sự tương quan giữa chúng.
- Đối với bài toán phân loại: kết quả cuối cùng là biểu quyết đa số (majority voting) từ các cây.

3.3 Công nghệ sử dụng

3.3.1 Hệ thống RIDX

Hệ thống RIDX được phát triển bởi nhóm sinh viên Đại học Bách Khoa Hà Nội, dưới sự hướng dẫn của giảng viên Giang V.T.H và Tuan.N.M, nhằm phục vụ mục tiêu đánh giá rủi ro hệ thống một cách hiệu quả. Dựa vào thư viện mạnh mẽ Pysmile có chức năng xây dựng trên các mô hình mạng Bayes giúp tính toán xác xuất xảy ra rủi ro của hệ thống. Ngoài ra còn có thêm các chức năng phân loại phá hiện tấn công dựa vào việc sử dụng các mô hình học máy từ thư viện Sckit-learn. Được thiết kế theo kiến trúc Microservices, hệ thống sử dụng NestJS cho phần backend và ReactJS kết hợp với các thư viện như AntD và UmiJS để tạo ra giao diện người dùng. Cụ thể về các công nghệ sử dụng trong hệ thống sẽ được trình bày tiếp sau đây.

3.3.2 Frontend

a) ReactJs

ReactJS [3] là một thư viện JavaScript mã nguồn mở do Facebook phát triển, ra mắt lần đầu vào năm 2013, chuyên dùng để xây dựng giao diện người dùng (UI) cho các ứng dụng web hiện đại. React tập trung vào việc phát triển các thành phần (component) UI có tính tương tác cao, dễ tái sử dụng, cho phép lập trình viên chia nhỏ giao diện phức tạp thành các phần nhỏ, độc lập để dễ quản lý và bảo trì. React sử dụng cú pháp JSX – một phần mở rộng của JavaScript –

cho phép kết hợp mã HTML và JavaScript trong cùng một tệp, giúp mã nguồn trở nên trực quan và dễ hiểu hơn.

Trong hệ thống RIDX, React được sử dụng để xây dựng giao diện người dùng và kết nối với backend. JSX cùng với thư viện Ant Design (AntD) được sử dụng để hỗ trợ thiết kế giao diện, nội dung chi tiết sẽ được trình bày ở phần dưới.

b) Antd

Ant Design (antd) [4] là một thư viện thiết kế giao diện người dùng (UI) dành cho các ứng dụng web sử dụng ReactJS. Antd cung cấp các thành phần UI như Button, Input, Table, Form,v.v., dưới dạng component giúp việc tích hợp và sử dụng dễ dàng. Thư viện này hỗ trợ tùy chỉnh theme, màu sắc, kiểu chữ và kích thước linh hoạt. Trong hệ thống RIDX, antd được sử dụng chủ yếu để xây dựng khung giao diện và thiết kế các thành phần UI phục vụ cho việc hiển thị thông tin của hệ thống.

c) UmiJs

UmiJS [5] đóng vai trò là một framework hỗ trợ điều hướng (routing) cho các ứng dụng web xây dựng bằng ReactJS. Hệ thống routing của UmiJS dựa trên react-router, cho phép định nghĩa các tuyến đường (route) một cách linh hoạt thông qua cấu hình hoặc tự động dựa vào cấu trúc thư mục trong thư mục pages. Ngoài ra thư viện còn cho phép tuỳ chỉnh layout của từng route, sử dụng các thành phần chung mà không cần phải thiết kế lại. Trong hệ thống RIDX, UmiJS đóng vai trò điều hướng tự động giữa các trang, đồng thời giúp quản lý cấu trúc thư mục pages một cách rõ ràng và cấu hình routing thông qua thư mục config.

3.3.3 Backend

a) NodeJs

NodeJS [6] không phải là một ngôn ngữ lập trình hay framework, mà là một môi trường runtime cho JavaScript, cung cấp các API, module và công cụ để xây dựng ứng dụng phía máy chủ. Hệ sinh thái npm (Node Package Manager) cung cấp nhiều thư viện, module hỗ trợ đa dạng nhu cầu lập trình, thúc đẩy phát triển nhanh, tiết kiệm thời gian và công sức. Trong hệ thống RIDX sử dụng môi trường NodeJS để xây dựng giao diện trong môi trường NodeJS 16. Ngoài ra Backend cũng sử dụng FrameWork của NodeJS là NestJS sẽ được trình bày dưới đây.

b) NestJS

NestJS [7] là một framework mã nguồn mở phát triển trên nền tảng Node.js, sử dụng ngôn ngữ TypeScript để xây dựng các ứng dụng server-side hiệu quả, có khả năng mở rộng cao và dễ dàng bảo trì. NestJS áp dụng DI để quản lý các phụ thuộc giữa các module và component, giúp tăng tính độc lập, dễ kiểm thử và bảo trì ứng dụng. Trong hệ thống sử dụng kiến trúc Microservices của NestJS chia kiến trúc của hệ thống thành các service nhỏ hoạt động độc lập.

3.3.4 Cơ sử dữ liệu MongoDB

MongoDB [8] là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL), mã nguồn mở, được phát triển bởi MongoDB Inc và ra mắt lần đầu vào năm 2009. Khác với các hệ quản trị cơ sở dữ liệu quan hệ, MongoDB lưu trữ dữ liệu dưới dạng tài liệu (document) với cấu trúc BSON (Binary JSON), cho phép lưu trữ dữ liệu linh hoạt, không cần định nghĩa schema trước. Hỗ trợ sharding, giúp phân tán dữ liệu trên nhiều máy chủ hoặc cluster để xử lý khối lượng dữ liệu lớn và tăng khả năng chịu tải rất phù hợp với kiến trúc Microservices mà hệ thống RIDX được thiết kế.

3.3.5 Thuật toán lõi (Core service)

a) Thư viện Pysmile

PySMILE [9] là một thư viện Python được sử dụng để xây dựng, suy diễn (inference) và học (learning) trên các mô hình mạng Bayes (Bayesian Networks). Thư viện SMILE một engine mạnh mẽ và phổ biến cho các mô hình đồ thị xác suất như mạng Bayes, mạng Bayes động. Hỗ trợ đầy đủ các thuật toán suy diễn xác suất tiên tiến. Trong hệ thống RIDX thư viện được sử dụng để tính xác xuất xảy rủi ro của hệ thống và được triển khai bởi mô hình mạng Bayes tĩnh và Bayes động.

b) FastAPI

FastAPI [10] là một framework web hiện đại cho Python, chuyên dùng để xây dựng các RESTful API với hiệu suất cao, tối ưu cho các ứng dụng cần xử lý nhiều yêu cầu đồng thời hoặc các tác vụ I/O-bound. Với cú pháp đơn giản, hỗ trợ gợi ý kiểu dữ liệu (type hint), giúp lập trình viên code nhanh và giảm lỗi. Trong hệ thống các API được xây dựng ở core đều sử dụng FastAPI để gọi đến các các dịch vụ xử lý tính toán phức tạp. Các API được xây dựng đều được khai báo và kết nối đến APIGateway để kết nối đến người dùng.

c) Thư viện Sckit-learn

Scikit-learn [11] (thường được gọi là sklearn) là một thư viện mã nguồn mở nổi bật trong hệ sinh thái Python, chuyên dùng cho các tác vụ học máy (Machine Learning - ML). API của Scikit-learn rất dễ tiếp cận với người dùng và được

thiết kế nhất quán, giúp việc xây dựng, huấn luyện và đánh giá mô hình ML trở nên đơn giản. Scikit-learn cung cấp một bộ công cụ toàn diện phục vụ cho các tác vụ học máy: xử lý dữ liệu, phân loại, hồi quy, v.v.. Trong hệ thống RIDX, thư viện được sử dụng để phục vụ cho việc huấn luyện mô hình học máy và xử lý dữ liệu từ các bộ dữ liệu dataset về tấn công APT. Thư viện giúp xử lý nhanh chóng và dễ dàng trong việc xây dựng các mô hình và đánh giá kết quả thông qua các thư viện được xây dựng trước.

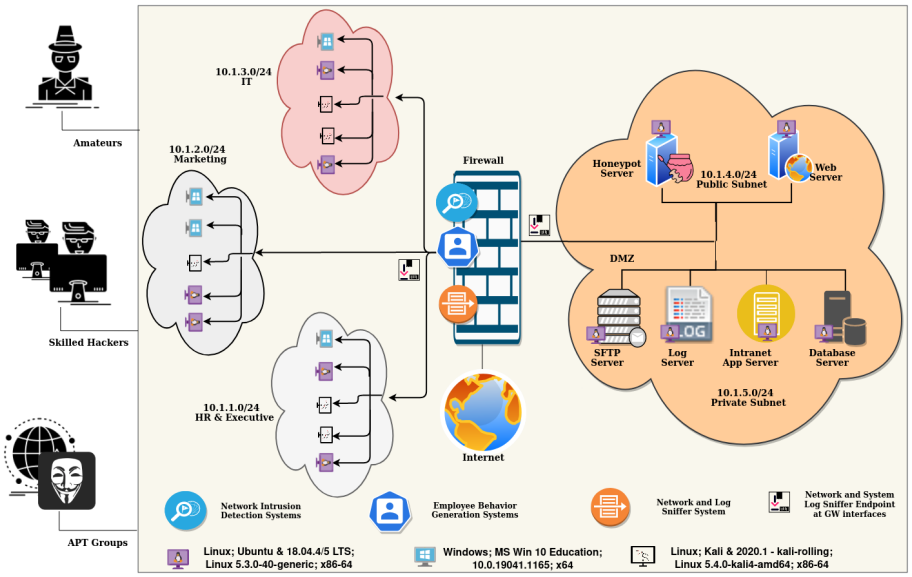
3.4 Bộ dữ liệu

3.4.1 Bộ dữ liệu unraveled

a) Giới thiệu

Unraveled [12] được thiết kế để mô phỏng các cuộc tấn công APT một dạng tấn công mạng tinh vi, kéo dài và khó phát hiện, thường nhằm vào các tổ chức lớn.

b) Thông tin hệ thống



Hình 3.3: Kiến trúc của hệ thống bộ dữ liệu Unraveled

Hình 3.3 là cấu trúc của hệ thống của bộ dữ liệu Unraveled. Hình ảnh thể hiện cấu hình hệ thống, trong đó tường lửa ngăn cách vùng người dùng hệ thống và vùng thiết bị. Có thể chia hệ thống làm 5 vùng hoạt động chính: HR & Executive, Marketing, IT, Production-public, Production-private, Gateway.

c) Thông tin dữ liệu

Dữ liệu thu được gồm có 2 phần chính là dữ liệu về network flows và host logs. Chi tiết về dữ liệu được trình bày dưới đây:

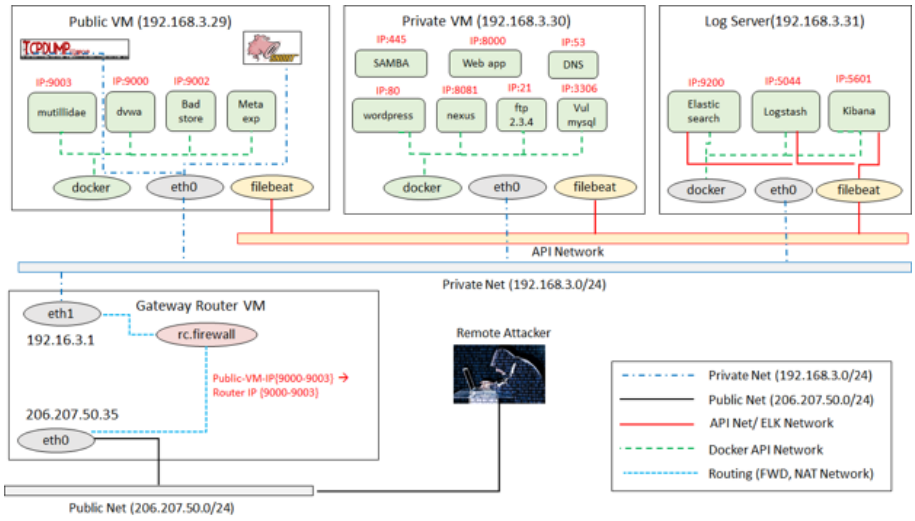
- Dữ liệu network flows thống kê về lưu lượng mạng được xử lý, 6,8 triệu lưu lượng, được lấy từ khoảng 1TB dữ liệu thô được thu thập trong 6 tuần. Trong đó dữ liệu tấn công nhiều nhất nằm ở vùng IT(3.2%). Với dữ liệu network flows gồm có các nhãn: Activity, Stage, Defender Response, Signature.
- Dữ liệu host logs: Nhật ký cấp máy chủ ghi lại hoạt động cấp máy chủ bao gồm nhưng không giới hạn ở các quy trình và lệnh chạy trên hệ thống của nhân viên và các kết nối được thực hiện với hệ thống của họ. Chúng chứa audit logs, auth logs system logs.

3.4.2 Bộ dữ liệu dapt2020

a) Giới thiệu

Bộ dữ liệu DAPT2020 [13] được xây dựng với lưu lượng mạng thu thập trong 5 ngày, trong đó mỗi ngày được thiết kế để mô phỏng tương đương 3 tháng hoạt động trong môi trường thực tế. Bộ dữ liệu này nhằm cung cấp cho các nhà nghiên cứu công cụ để phân tích các điểm bất thường, khám phá mối quan hệ giữa các vectơ tấn công khác nhau và phát hiện các tương quan ẩn, từ đó hỗ trợ nhận diện sớm các cuộc tấn công APT một cách hiệu quả.

b) Thông tin hệ thống



Hình 3.4: Kiến trúc của hệ thống bộ dữ liệu Dapt2020

Hình 3.4 mô tả kiến trúc của hệ thống. Có thể chia thành các vùng: Public VM, Private VM, Log Server VM, Gateway Router VM, Docker Network. Public VM chạy các dịch vụ dễ bị tấn công như DVWA, Bad Store, ánh xạ cổng (ví dụ: 172.16.0.2:80 → 192.168.3.29:9000), sử dụng Snort IDS và TCP Dump để phát hiện tấn công (SYN Flood, Illegal Access). Private VM chứa

các dịch vụ như WordPress, FTP, MySQL từ VulnHub, chỉ kết nối nội bộ. Cả hai dùng Filebeat gửi log (Audit, Access, IDS) đến Log Server chạy ELK Stack (ElasticSearch, Logstash, Kibana) để phân tích. Gateway Router dùng NAT rules để lộ dịch vụ ra mạng công cộng. Docker0 kết nối container, hỗ trợ API (màu xanh) cho lệnh Docker.

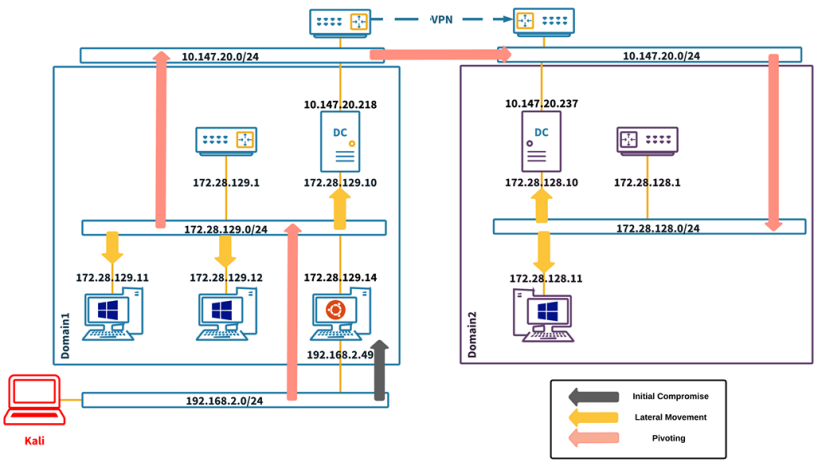
c) Thông tin dữ liệu Bộ dữ liệu gồm có 2 nhãn cung cấp cho người dùng thông tin về các hoạt động độc hại và chúng thuộc giai đoạn nào là nhãn Stage và Activity. Bộ dữ liệu có nhãn là Stage gồm các giai đoạn tấn công của APT: Reconnaissance, Establish Foothold, Lateral Movement, Internal Reconnaissance, Data Exfiltration, Cover Up. Có 76 features được trích xuất bởi CICFlowMeter.

3.4.3 Bộ dữ liệu SCVIC-APT-2021

a) Giới thiệu

Bộ dữ liệu SCVIC-APT-2021 [14] được phát triển bởi Smart Connected Vehicles Innovation Centre (SCVIC) thuộc Đại học Ottawa và quan trọng cho nghiên cứu APT detection.

b) Thông tin hệ thống



Hình 3.5: Kiến trúc của hệ thống bộ dữ liệu SCVIC-APT-2021

Như được minh họa trong Hình 3.5, hệ thống được thiết kế là một môi trường đa miền bao gồm hai miền và bốn mạng con, hỗ trợ mô phỏng các cuộc tấn công APT như di chuyển ngang và chuyển hướng. Miền 1 gồm bốn máy, trong đó có một Bộ điều khiển miền (DC) và ba thiết bị đầu cuối thông thường, là nơi các nạn nhân ban đầu bị tấn công. Miền 2 bao gồm một Bộ điều khiển miền (DC) và một máy tính cá nhân, được xem là mục tiêu cuối cùng để thu

thập thông tin đăng nhập. Hai miền được kết nối qua mạng riêng ảo (VPN). Công cụ Kali được sử dụng để thiết kế và thực hiện các cuộc tấn công, cho phép truy cập vào các máy trong miền 1.

c) Thông tin dữ liệu

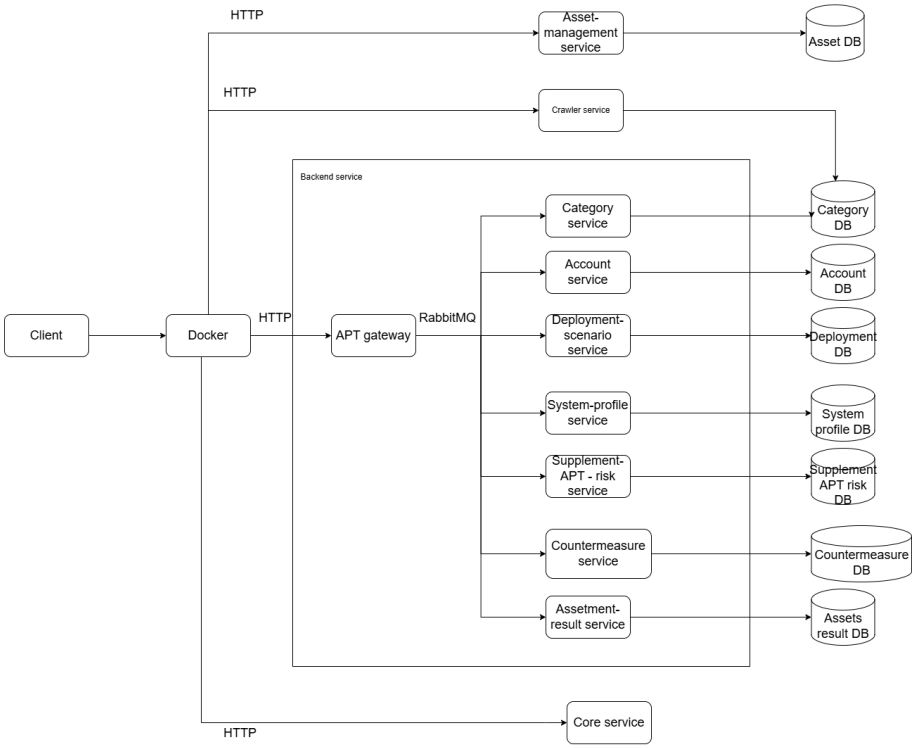
Dữ liệu được mô tả liên quan đến quy trình của một cuộc tấn công APT, bao gồm sáu giai đoạn: thu thập thông tin (reconnaissance), xâm nhập ban đầu (initial compromise), di chuyển ngang (lateral movement), chuyển hướng (pivoting), trích xuất dữ liệu (data exfiltration) và giai đoạn sau tấn công (post-attack stage). Cơ sở tri thức toàn cầu MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) được sử dụng làm nền tảng để lựa chọn các kỹ thuật tấn công phổ biến. Các kỹ thuật tấn công tương ứng. Các nhãn gồm có Stage, Activity.

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

Trong các chương trước đã giới thiệu về chủ đề cũng như các chức năng và công nghệ được sử dụng của hệ thống sử dụng. Chương này sẽ tập trung trình bày chi tiết hơn về thiết kế kiến trúc tổng thể của hệ thống, bao gồm các thành phần chính, cách thức xây dựng phần mềm và phương án triển khai. Qua đó, làm rõ cách hệ thống được tổ chức và vận hành nhằm đạt được các mục tiêu đã đề ra.

4.1 Thiết kế kiến trúc hệ thống

4.1.1 Tổng quan kiến trúc hệ thống RIDX



Hình 4.1: Tổng quan kiến trúc của hệ thống RIDX

Hình 4.1 thể hiện kiến trúc tổng quan của hệ thống RIDX từ người dùng đến các service của hệ thống. Hệ thống được xây dựng theo kiến trúc Microservices gồm nhiều các services hoạt động độc lập kết nối thông quan API Gateway. Theo đó, ứng dụng được phân tách thành các dịch vụ (services) nhỏ, độc lập. Cụ thể, các dịch vụ như "Category service", "Account service", "Deployment-scenario service", "System-profile service", "Supplement-APT-risk service", "Countermeasure service", và "Assetment-result service" đều là các microservice. Mỗi dịch vụ này thực hiện một chức năng nghiệp vụ riêng biệt và thường sở hữu cơ sở dữ liệu (CSDL) riêng, ví dụ: "Category DB", "Account DB", "Deployment DB", "System profile DB", "Supplement APT risk DB", "Countermeasure DB", và "Assets result

DB". Các tương tác của người dùng khi gọi request sẽ được xử lý bởi "Backend service" tập trung ở "API gateway". Các request sẽ gọi đến các service phù hợp với yêu cầu của người dùng. Việc áp dụng "API gateway" giúp đơn giản hóa việc quản lý API nội bộ, đồng thời che giấu hiệu quả cấu trúc phức tạp của hệ thống microservices phía sau khỏi môi trường bên ngoài, tăng cường tính bảo mật và khả năng bảo trì.

Các dịch vụ tương tác thông qua API Gateway:

- **API Gateway:** Đóng vai trò là điểm truy cập duy nhất cho tất cả các yêu cầu từ phía người dùng. API Gateway chịu trách nhiệm chính trong việc xác thực bảo mật, ủy quyền (phân quyền) và định tuyến các yêu cầu đến các dịch vụ phù hợp.
- **Account Service:** Có chức năng quản lý thông tin người dùng, chứa các thông tin như username, password, v.v.Thông tin sẽ được sử dụng để đăng nhập vào hệ thống với vai trò USER hoặc ADMIN.
- **Deployment Scenario Service:** Có chức năng quản lý thông tin liên quan đến kịch bản triển khai như thông tin về tài sản, mối quan hệ giữa các tài sản để có thể tạo được sơ đồ của hệ thống triển khai.
- **System Profile Service:** Quản lý hồ sơ hệ thống, bao gồm danh sách các kịch bản triển khai khả thi có thể được xây dựng cho một hệ thống cụ thể.
- **Category Service:** Cung cấp chức năng quản lý và hỗ trợ người dùng truy vấn các loại dữ liệu an ninh mạng như CVE (Common Vulnerabilities and Exposures), CPE (Common Platform Enumeration) và CWE (Common Weakness Enumeration). Các dữ liệu này đóng vai trò là tài liệu tham khảo quan trọng giúp người dùng xây dựng các kịch bản triển khai hiệu quả.
- **Countermeasure Service:** Cung cấp chức năng quản lý biện pháp phòng thủ bao gồm thông tin về cấu hình. Dịch vụ này cho phép người dùng bổ sung các biện pháp phòng vệ vào kịch bản triển khai để so sánh kết quả đánh giá rủi ro khi có và không có các biện pháp đó.
- **Assessment Result Service:** Chức năng này có vai trò lưu trữ các kết quả mà hệ thống đánh giá thông qua kịch bản triển khai. Chứa các thông tin về các tài sản, mỗi quan hệ và các kết quả đánh giá. Cho phép người dùng có thể xem lại kết quả đánh giá một cách đầy đủ nhất mà không cần phải thực hiện lại.
- **Supplement APT Risk Service:** Có chức năng quản lý thông tin liên quan đến tỷ lệ xảy ra tấn công và mức độ hiệu quả của biện pháp phòng thủ của hệ thống trong quá trình đánh giá. Kết quả dựa trên đánh giá về subnet 1 và

subnet 2 của đầu vào mạng Bayes.

- **Các dịch vụ hoạt động độc lập (hoặc chưa tích hợp trực tiếp với API Gateway):**

Ngoài các dịch vụ được trình bày ở trên, hệ thống RiDX còn bao gồm ba dịch vụ chưa được tích hợp trực tiếp với API Gateway:

- **Asset Management Service:** Quản lý tài sản của hệ thống. Dịch vụ này bao gồm các chức năng CRUD các loại tài sản khác nhau như phần cứng, phần mềm, hệ điều hành, v.v.
- **Core Service:** Chứa các thuật toán lõi có các chức năng tính toán, đánh giá rủi ro của hệ thống, phát hiện rủi ro.
- **Crawler Service:** Dịch vụ có chức năng là cập nhật thông tin của các tài sản với dữ liệu mới. Với các thiết bị thì luôn luôn có sự thay đổi của các lỗ hổng vì thế cần được cập nhật thường xuyên để đảm bảo tính mới của dữ liệu.

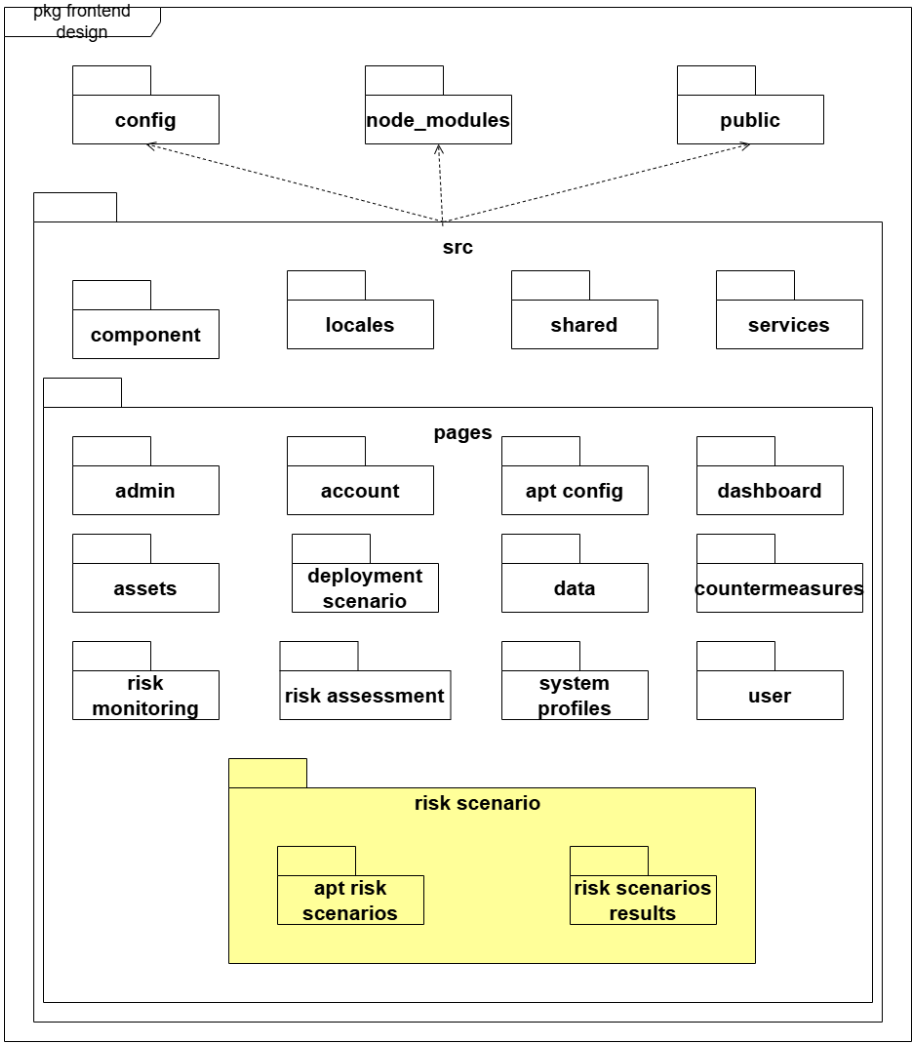
Với kiến trúc Microservices của hệ thống đã trình bày, người dùng sẽ tương tác với hệ thống thông qua API Gateway từ đó các yêu cầu người dùng sẽ được truyền đến các service để xử lý. API Gateway đóng vai trò trung tâm, thực hiện việc định tuyến các yêu cầu đến các dịch vụ tương ứng. API Gateway không chỉ giúp tách biệt giao diện người dùng với các dịch vụ phía sau mà còn góp phần nâng cao khả năng mở rộng và bảo mật của toàn hệ thống. Việc giao tiếp nội bộ giữa API Gateway và các microservice được thực hiện thông qua hệ thống hàng đợi thông điệp RabbitMQ, giúp đảm bảo độ tin cậy và tính linh hoạt trong xử lý bất đồng bộ.

Mỗi dịch vụ (service) trong hệ thống microservice của RiDX được thiết kế theo cấu trúc ba tầng riêng biệt, bao gồm:

- **Tầng API (API Layer):** Chịu trách nhiệm tiếp nhận các yêu cầu từ phía client và chuyển tiếp các yêu cầu này đến tầng nghiệp vụ để xử lý.
- **Tầng nghiệp vụ (Business Logic Layer):** Là tầng nằm ở giữa có chứa các xử lý logic. Tiếp nhận các xử lý từ tầng API và tương tác với tần lưu trữ để xử lý các yêu cầu của người dùng.
- **Tầng Lưu trữ (Data Access Layer/Storage Layer):** Là thành phần tương tác chính với cơ sở dữ liệu (CSDL). Tầng này có chức năng truy xuất, cập nhật dữ liệu theo yêu cầu xử lý từ tầng nghiệp vụ, đảm bảo tính toàn vẹn và nhất quán của dữ liệu.

4.1.2 Thiết kế tổng quan

a) Thiết kế tổng quan thành phần Client



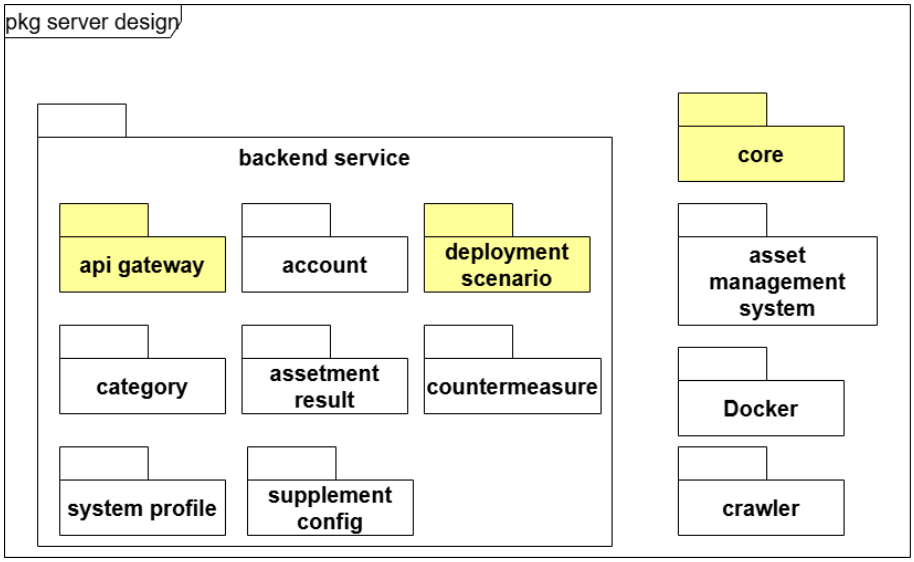
Hình 4.2: Thiết kế tổng quan thành phần Frontend

Hình 4.2 minh họa cấu trúc mã nguồn của giao diện người dùng. Sơ đồ trình bày cấu trúc thư mục cùng với các màn hình chức năng chính của ứng dụng. Các thành phần được tô màu vàng thể hiện những module được phát triển nhằm tích hợp chức năng phân tích và chấm điểm kịch bản rủi ro APT dựa trên dữ liệu từ nhật ký mạng và hệ thống.Thành phần giao diện được tổ chức thành các gói chính sau:

- **config**: Là nơi khai báo các trang màn hình sẽ được sử dụng và định nghĩa các đường dẫn tương ứng. Thông qua sử dụng thư viện Umi liên kết thông minh tìm đến các trang ở thư mục pages.
- **node-modules**: Thư mục lưu trữ các thư viện và module được cài đặt phục vụ cho phần giao diện.
- **components**: Chứa các thành phần giao diện dùng chung, được tái sử dụng ở nhiều phần khác nhau trong ứng dụng.

- **src**: Chứa mã nguồn chính về các trang của ứng dụng được khai báo cũng như các thông tin về API sẽ được sử dụng để gửi, nhận dữ liệu với máy chủ. Mỗi thư mục trong này sẽ chưa file index sẽ được từ động liên kết đến phần khai báo các trang hiển thị thông qua sử dụng thư viện Umi.

b) Thiết kế tổng quan thành phần server



Hình 4.3: Thiết kế tổng quan thành phần server

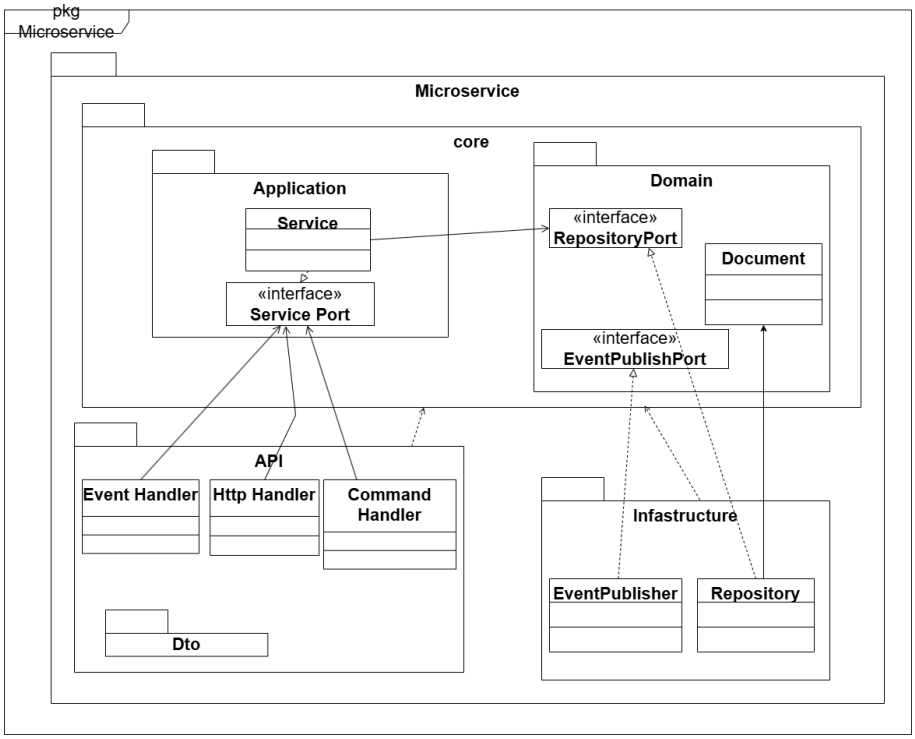
Hình 4.3 biểu diễn cấu trúc thư mục mã nguồn của server. Sơ đồ bao gồm thành phần backend-service thiết kế theo kiến trúc Microservices và ba thành phần kèm theo. Các thành phần của hệ thống server bao gồm:

- Backend-service: Thiết kế theo kiến trúc Microservices, thông qua "API gateway" định tuyến các yêu cầu đến microservice tương ứng được thiết kế riêng biệt. Thiết kế này giúp cho việc vận hành được cân bằng tải đảm bảo độ ổn định cũng như bảo mật trong xác thực và phân quyền. Giúp cho việc vận hành và bảo trì dễ dàng hơn.
- Core-service: Dịch vụ lõi chưa các thuật toán đánh giá rủi ro ATTT và các mô hình đã được huấn luyện phục vụ cho các chức năng phát hiện và đánh giá rủi ro bằng mạng Bayes.
- Asset-management-service: Dịch vụ tách biệt phục vụ cho việc quản lý các tài sản, thiết bị được định nghĩa sử dụng trong hệ thống. Chủ yếu cung cấp CRUD cho các tài sản.
- Crawler-service: dịch vụ thu thập dữ liệu về thông tin của các thiệt bị ứng với cấu hình tương ứng. Với từng thiết bị thông qua thông tin về cấu hình ta có thể ánh xạ để đưa ra được các CPE, CVE phù hợp.

4.1.3 Thiết kế chi tiết gói của các services

Trong phần này, đồ án sẽ trình bày thiết kế chi tiết của gói chung trong kiến trúc microservice, bao gồm các thành phần: (i) thiết kế chung của các service trong Microservices, (ii) thiết kế gói của API Gateway và (iii) thiết kế gói của Core service.

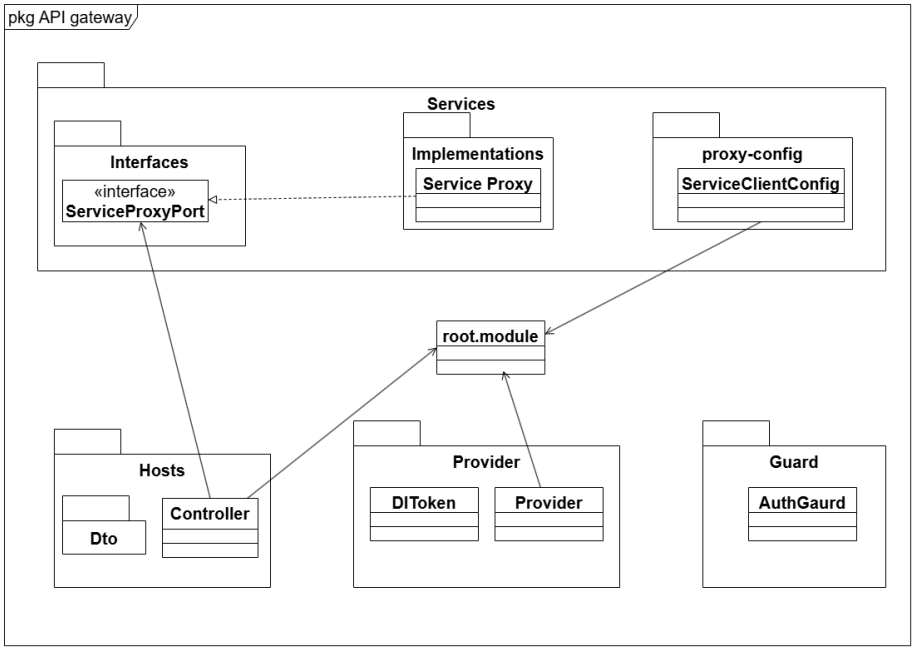
a) **Thiết kế chung của các service trong Microservices**



Hình 4.4: Thiết kế chi tiết gói chung mỗi microservice

Hình 4.4 minh họa thiết kế chi tiết của gói chung trong mỗi microservice. Trong đó, thư mục API đóng vai trò tiếp nhận yêu cầu từ phía người dùng thông qua các hàm controller, sau đó chuyển tiếp đến các chức năng xử lý cụ thể được định nghĩa trong gói Application. Gói Application bao gồm các interface và logic xử lý chính, thực hiện các thao tác CRUD đối với cơ sở dữ liệu thông qua các đối tượng trong gói Domain. Thiết kế này giúp cho hệ thống sửa đổi và bảo trì do các services hoạt động độc lập không ảnh hưởng đến nhau.

b) **Thiết kế gói của API Gateway**

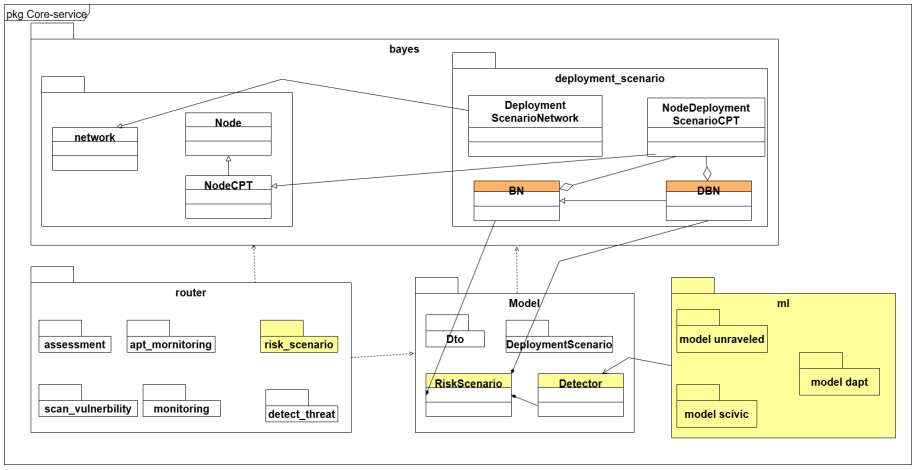


Hình 4.5: Thiết kế gói của APT Gateway

Hình 4.5 mô tả thiết kế chi tiết của gói API Gateway. Thành phần chính root.module.ts đóng vai trò là điểm khởi tạo và cấu hình, nơi khai báo và tích hợp các dịch vụ được sử dụng trong toàn bộ API Gateway. Gói Host chịu trách nhiệm khai báo các API tương ứng với từng dịch vụ cụ thể. Tại đây, các lớp sẽ thực thi các giao diện (interface) được định nghĩa trong ServicePort, đảm bảo tính thống nhất và khả năng mở rộng cho kiến trúc. Gói ServiceClientConfig là nơi cấu hình các microservice bên ngoài mà API Gateway cần kết nối, chẳng hạn như đường dẫn (endpoint), cổng kết nối (port), và các thông tin cấu hình liên quan. Cuối cùng, gói guard chứa lớp AuthGuard, đảm nhận vai trò xác thực (authentication) và phân quyền (authorization) cho người dùng, giúp bảo vệ các route và đảm bảo chỉ người dùng hợp lệ mới có quyền truy cập vào tài nguyên tương ứng.

c) **Thiết kế gói của Core service**

Trong gói Core Service, có chức năng phát hiện và tính toán rủi ro cho hệ thống. Chức năng phát hiện tấn công được triển khai thông qua các mô hình học máy đã được huấn luyện từ dữ liệu mạng thực tế. Các mô hình này được sử dụng trong lớp Detector, nơi tiếp nhận dữ liệu mạng thực và áp dụng mô hình để phát hiện các hành vi tấn công.Chức năng tính toán rủi ro được xây dựng dựa trên mạng Bayes, nhằm ước lượng xác suất xảy ra rủi ro dựa trên các thông tin hệ thống và kết quả từ quá trình phát hiện tấn công.



Hình 4.6: Thiết kế gói của Core service

Hình 4.6 minh họa mối quan hệ phụ thuộc giữa các lớp trong gói Core Service, chi tiết các thành phần như dưới đây.

- Model: Chứa các lớp chính như DeploymentScenario, APTMonitoring, Detector, và gói Dto, nơi định nghĩa các thuộc tính và phương thức hỗ trợ xử lý nghiệp vụ.
- Router: Chịu trách nhiệm khai báo các API được cung cấp bởi hệ thống. Các API này gọi đến các phương thức đã được định nghĩa trong các lớp thuộc gói Model.
- Bayes: Gói này chứa các lớp cốt lõi phục vụ cho việc xây dựng và vận hành mô hình mạng Bayes, được sử dụng để đánh giá mức độ rủi ro tương ứng với từng kịch bản triển khai.
- ML: Bao gồm các mô hình học máy tương ứng với từng hệ thống cụ thể. Các mô hình này được tích hợp và gọi sử dụng tại lớp Detector để thực hiện chức năng phát hiện tấn công.

4.2 Thiết kế chi tiết

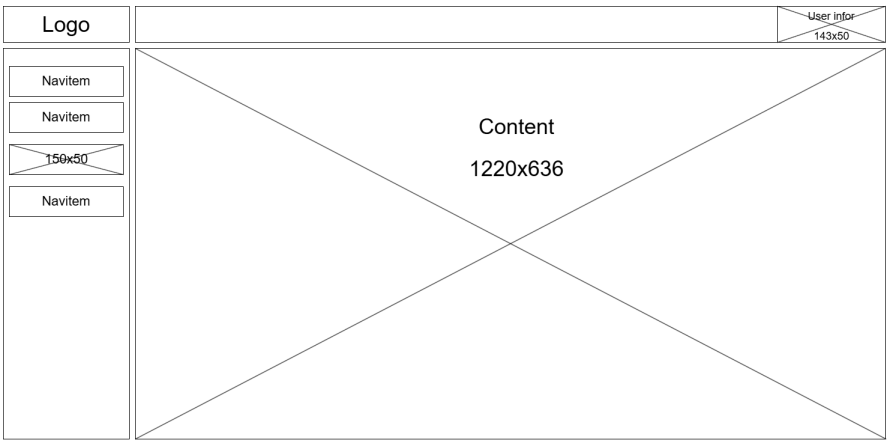
4.2.1 Thiết kế giao diện

a) Thiết kế giao diện tổng quan

Hình 4.7 minh hoạ giao diện tổng thể của hệ thống RIDX được thiết kế. Trên màn hình kích thước 15.6 inch, giao diện hiển thị của ứng dụng trên nền web được tối ưu với độ phân giải khoảng 1466x711 pixel, đảm bảo khả năng hiển thị đầy đủ và trực quan trên các thiết bị phổ biến.

Tông màu chủ đạo của giao diện là trắng, xanh dương, và tím than, nhằm tạo cảm giác hài hòa, hiện đại và dễ chịu cho người sử dụng. Các chức năng trong

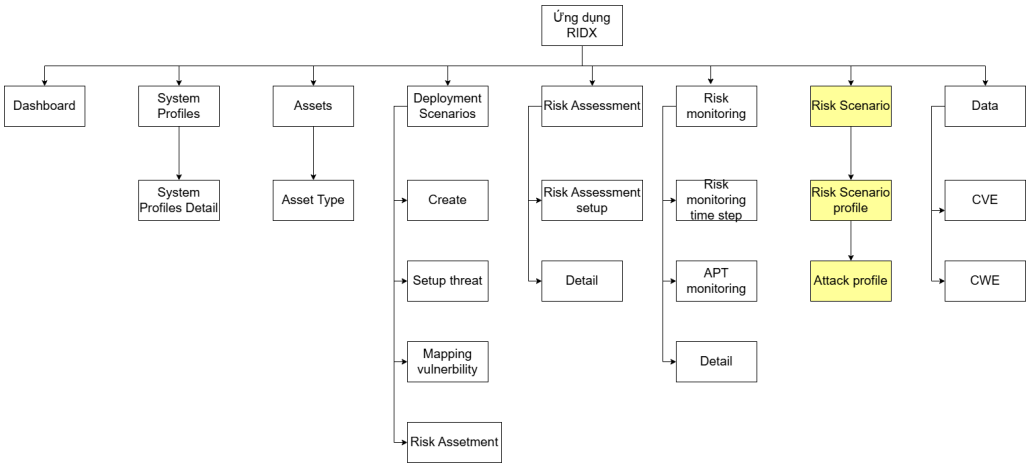
hệ thống được phân biệt rõ ràng thông qua màu sắc đại diện cho từng trạng thái. Ví dụ, các cảnh báo tấn công sẽ được hiển thị bằng màu đỏ, giúp người dùng dễ dàng nhận diện các sự kiện quan trọng. Hệ thống cũng sử dụng các biểu tượng (icon) phù hợp cho từng chức năng, góp phần hỗ trợ người dùng tương tác thuận tiện và dễ hiểu hơn.



Hình 4.7: Bố cục chung của giao diện hệ thống

Về thiết kế thì kích thước các phần đã được thể hiện trong hình 4.7 gồm các thành phần chính: (i) Header: Nằm ở phía trên, hiển thị logo hệ thống và thông tin tài khoản người dùng đã đăng nhập, (ii) Thanh điều hướng (Navbar): Đặt ở bên phải, cho phép người dùng chuyển đổi giữa các màn hình chức năng trong hệ thống, (iii) Khu vực nội dung chính: Là nơi hiển thị chi tiết các chức năng và thông tin tương ứng khi người dùng tương tác.

b) Thiết kế sitemap giao diện hệ thống RiDX

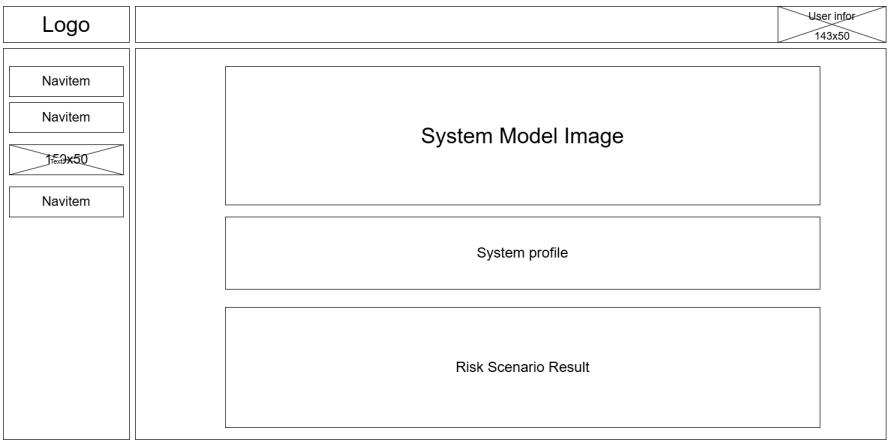


Hình 4.8: Thiết kế sitemap giao diện hệ thống RiDX

Sơ đồ trên mô tả kiến trúc chức năng của hệ thống RIDX, bao gồm nhiều

thành phần như: Dashboard, System Profiles, Assets, Deployment Scenarios, Risk Assessment, Risk Monitoring, Risk Scenario, và Data. Mỗi nhóm chức năng lại được chia nhỏ thành các thành phần con thể hiện quy trình xử lý và luồng dữ liệu trong hệ thống. Phần màu vàng trong biểu dồ biểu thị màn hình mới được xây dựng mô hình hóa rủi ro và hành vi tấn công.

c) Thiết kế giao diện chức năng phân tích và chấm điểm kịch bản rủi ro



Hình 4.9: Thiết kế giao diện chức năng phân tích và chấm điểm kịch bản rủi ro

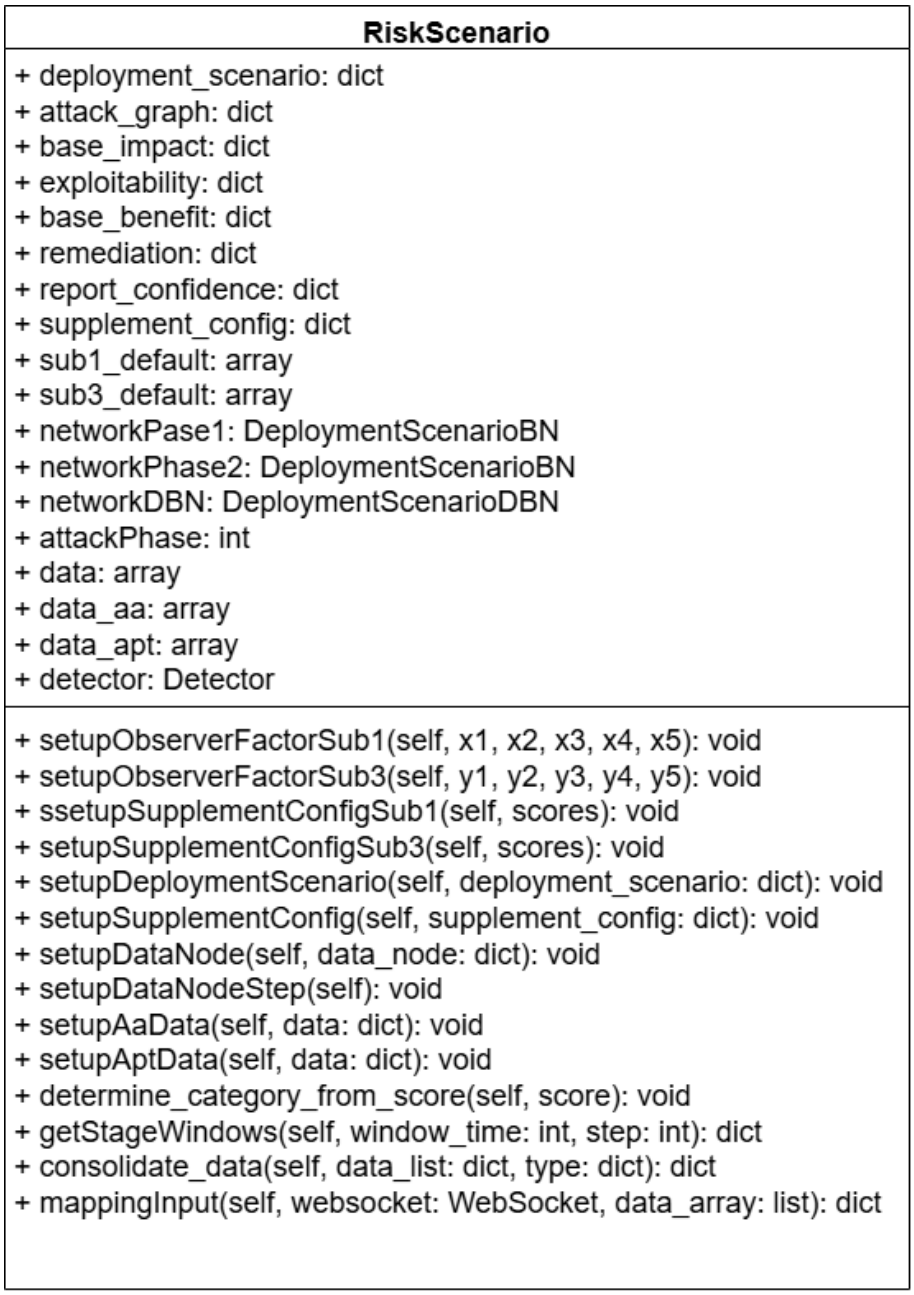
Hình 4.9 mô tả sơ lược về nội dung màn hình phân tích và chấm điểm kịch bản rủi ro được tích hợp vào hệ thống RIDX. Với hai chức năng là phân tích và chấm điểm kịch bản rủi ro thì được tạo thành hai màn hình. Sau khi thực hiện phân tích dữ liệu tấn công sẽ là màn chấm điểm kịch bản rủi ro. Nội dung chính của trang bao gồm: (i) system model image mô tả sơ đồ thiết kế của hệ thống đang theo dõi đánh giá, (ii)system information mô tả các thông tin của hệ thống như về các địa chỉ IP, mức độ truy cập các phân vùng, (iii) risk scenario result kết quả khi phát hiện ra tấn công được thống kê dưới dạng các biểu đồ.

4.2.2 Thiết kế lớp

Trong phần thiết kế lớp, hệ thống RIDX tập trung mô tả hai lớp quan trọng trong gói Core Service, bao gồm: RiskScenario và Detector. Hai lớp này đóng vai trò then chốt trong quá trình đánh giá và phát hiện rủi ro an ninh mạng. Các thuộc tính và phương thức cụ thể của từng lớp sẽ được thể hiện trong các sơ đồ thiết kế chi tiết tại hình 4.10 và hình 4.11.

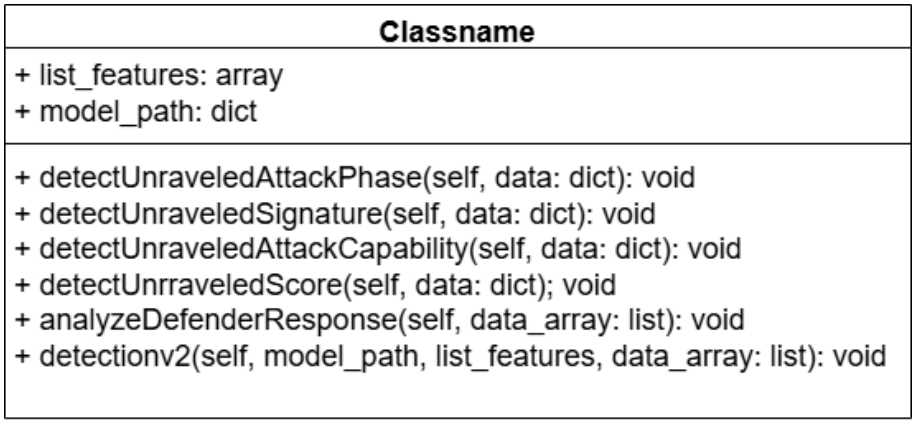
Hình 4.10 lớp RiskScenario có nhiệm vụ tổng hợp và phân tích các thông tin liên quan đến bên tấn công, cơ chế phòng thủ và các đối tượng tấn công. Các dữ liệu truyền xuống sẽ được thực hiển bởi các hàm set dữ liệu. Hàm xử lý chính sẽ là

hàm mappingInput có vai trò xử lý dữ liệu mạng tấn công. Lớp này sẽ gọi đến các xử lý ở lớp Detect để thực hiện gọi đến dịch vụ mô hình học máy dự đoán thông tin. Nhiệm vụ chính của lớp này là đánh giá rủi ro và chấm điểm cho subnet 1 và 2.



Hình 4.10: Thiết kế chi tiết lớp RiskScenario

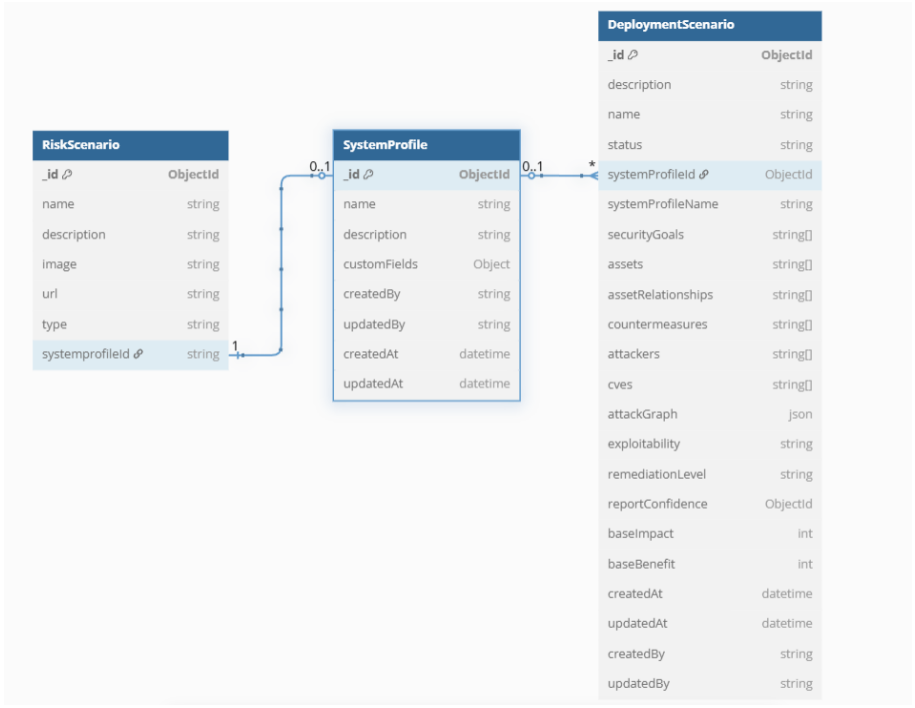
Hình 4.11 đảm nhiệm vai trò phát hiện tấn công dựa trên dữ liệu mạng thực tế. Với đầu vào là dữ liệu mạng được thu thập, các mô hình sau khi được huấn luyện sẽ phân tích và đưa ra cảnh báo về những hành vi bất thường. Lớp Detector sẽ có nhiệm vụ xử lý gọi đến các mô hình học máy để phân tích dữ liêu. Kết quả đầu ra của lớp Detector là kết quả về các giai đoạn tấn công, mức độ phòng thủ, đối tượng tấn công mà thông qua phân tích dữ liệu tấn công thu được.



Hình 4.11: Thiết kế chi tiết lớp Detector

4.2.3 Thiết kế cơ sở dữ liệu

Hệ thống RIDX sử dụng cơ sở dữ liệu không quan hệ MongoDB. Với thiết kế kiến trúc theo microservices thì với mỗi service riêng thì sẽ có có sở dữ liệu riêng. Với chức năng được thêm về tổng hợp thông tin của cuộc tấn công thành kịch bản rủi ro cần bổ sung thêm bảng về kịch bản rủi ro ứng với hệ thống. Hình 4.12 của thể hiện mỗi quan hệ của kịch bản rủi ro.



Hình 4.12: Thiết kế kịch bản rủi ro

Kịch bản rủi ro được tạo ra chứ các thông tin về hệ thống như hình thiết kế hệ thống, mô tả, v.v. Ngoài ra ứng với các hệ thống thì có mối quan hệ với system-profile từ đấy sẽ lấy được thông tin của deploymentscenario, đây là thông tin quan trọng mà một trong những đầu vào cho mạng đánh giá rủi ro. Trong kịch bản triển

khai sẽ chứa các thông tin chi tiết về các tài sản trong hệ thống triển khai.

4.3 Xây dựng ứng dụng

4.3.1 Thư viện và công cụ sử dụng

Bảng 4.1 liệt kê công nghệ sử dụng cũng như tham chiếu và phiên bản sử dụng hiện tại đang có trong hệ thống.

Mục đích	**Công cụ**	**Địa chỉ URL**
IDE lập trình	PyCharm Community Edition version 2023.3.5	`https://www.jetbrains.com/pycharm/`
Editor mã nguồn	Visual Studio Code version 1.100.2	`https://code.visualstudio.com/`
Framework Back-end	NestJS version 9.0.0	`https://nestjs.com/`
Cơ sở dữ liệu	MongoDB version 4.4	`https://www.mongodb.com/`
Triển khai	Docker version 27.0.3	`https://www.docker.com/`
Thư viện tính toán mạng Bayes	PySmile	`https://www.bayesfusion.com/smile`
Lập trình backend	FastAPI version 0.63.0	`https://fastapi.tiangolo.com`
Môi trường lập trình front-end và backend	NodeJs version 16.20.2	`https://nodejs.org`

Bảng 4.1: Danh sách thư viện và công cụ sử dụng

4.3.2 Kết quả đạt được

Thông tin thống kê	**Kết quả thống kê**
Dung lượng phía frontend (có thư viện)	1.84GB
Dung lượng phía backend (có thư viện)	783.8 MB
Số lượng service trong hệ thống	3.94GB
Số file jsx xây dựng ở frontend	14 file
Số file xây dựng và tái cấu trúc ở backend	16 file
Tổng số service đang triển khai	11

Bảng 4.2: Thống kê hệ thống

4.4 Kiểm thử

4.4.1 Kiểm thử tương thích

Bảng 4.3 là các thiết bị đã kiểm thử hệ thống để kiểm tra tính tương thích đối với hoạt động của backend và frontend.

Device	Specifications	UI	Function
Laptop Dell G15	Screen: 15.6 inches FHD, 16GB RAM	Pass	Pass
Laptop Dell	Screen: 15.6 inches FHD, 16GB RAM	Pass	Pass

Bảng 4.3: Device Specifications

4.4.2 Kiểm thử chức năng

a) Chức năng tạo kịch bản rủi ro

STT	Input	Output	Kết quả
1	Các thông tin của kích bản rùi ro	Kích bản rùi ro mới với các thông tin đã nhập	Đạt
2	Nhập thiếu các trường thông tin	Cảnh báo nhập thiếu thông tin và yêu cầu nhập đủ	Đạt

Bảng 4.4: Bảng test case chức năng tạo kịch bản rủi ro

b) Chức năng phân tích dữ liệu

STT	Input	Output	Kết quả
1	Upload dữ liệu mạng (network flows)	Màn hình kết quả phân tích dữ liệu	Đạt
2	Dữ liệu của bộ dữ liệu khác	Thông báo lỗi không đúng định dạng	Đạt
3	Upload dữ liệu log của hệ thống	Màn hình hiển thị thông tin tài sản theo log	Đạt
4	Chọn khung thời gian và bước nhảy thời gian	Biểu đồ về số lượng giai đoạn tấn công theo khung thời gian và bước nhảy thời gian đã nhập	Đạt

Bảng 4.5: Bảng test case chức năng phân tích dữ liệu

c) Chức năng đánh giá, tích hợp điểm đánh giá subnet 1 và subnet2

STT	Input	Output	Kết quả
1	Điểm đánh giá mạng 1 và mạng 2 đã tính toán	Màn hình với kết quả điểm đánh giá mới	Đạt
2	Dữ liệu mạng của hệ thống	Màn hình với thông tin các điểm đánh giá mạng 1 và mạng 2	Đạt

Bảng 4.6: Bảng test case chức năng đánh giá, tích hợp điểm đánh giá subnet 1 và subnet2

4.5 Triển khai

Hệ thống RiDX, với bản cập nhật bổ sung chức năng Phân tích và chấm điểm kịch bản rủi ro đã được triển khai tại local host. Ngoài ra, người dùng có thể triển khai hệ thống cục bộ trên máy cá nhân thông qua Docker. Toàn bộ hướng dẫn cài đặt, bao gồm cách dựng môi trường, cấu hình các dịch vụ liên quan và cách chạy hệ thống trên localhost, đã được trình bày chi tiết trong tệp README đi kèm mã nguồn. Việc sử dụng Docker giúp đơn giản hóa quy trình triển khai, đảm bảo tính nhất quán của môi trường phát triển và dễ dàng mở rộng hệ thống trong các môi trường thực tế.

CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT

Trong chương trước đã trình bày về các thiết kế, chức năng tổng quan của hệ thống RIDX. Chương này sẽ tập trung trình bày các đóng góp cốt lõi của đồ án, bao gồm: (i) Xây dựng mô hình học máy để phân tích kịch bản rủi ro tự động từ bộ dữ liệu tấn công mạng (ii) Xây dựng chức năng đánh giá điểm cho cuộc tấn công (iii) Tích hợp chức năng phân tích và đánh giá rủi ro tấn công

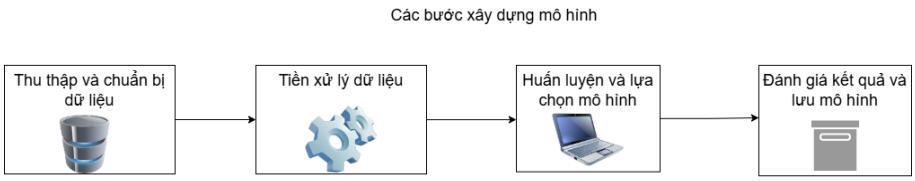
5.1 Xây dựng mô hình học máy để phân tích thông tin tự động từ dữ liệu tấn công mạng

5.1.1 Vấn đề

Để phân tích rủi ro của hệ thống từ các dữ liệu mạng thực tế thì việc thực hiện bằng thủ công rất khó khăn và mất thời gian mà hiện quả đem lại thì không cao. Với số lượng dữ liệu lớn thì khó có thể phân tích và yêu cầu chi phí rất lớn. Vì thế việc sử dụng các mô hình học máy (machine learning) vào là rất cần thiết. Hiện nay việc áp dùng các mô hình học máy đã được xây dựng nhưng vẫn còn rất nhiều hạn chế về: độ chính xác, chất lượng nhãn phân biệt, v.v. Vì thế cần có phương pháp tối ưu để phát hiện các rủi ro trong quá trình triển khai.

5.1.2 Giải pháp

Từ những phân tích trên, vấn đề cốt lõi cần giải quyết là xây dựng một mô hình phát hiện tấn công hiệu suất cao, cung cấp thông tin chi tiết về các giai đoạn tấn công. Để đạt được mục tiêu này, chúng tôi đề xuất hai phương pháp huấn luyện mô hình dựa trên các loại dữ liệu khác nhau: Huấn luyện chỉ với network flows, Huấn luyện với dữ liệu kết hợp network flows và host logs. Việc huấn luyện và đánh giá mô hình sẽ được thực hiện trên ba bộ dữ liệu mô phỏng: Unraveled, DAPT-2020, và SCVIC-APT-2021. Những bộ dữ liệu này được lựa chọn vì chúng đại diện cho các kịch bản tấn công APT phổ biến trong môi trường doanh nghiệp và có độ tương đồng cao về kiến trúc dữ liệu, tạo điều kiện thuận lợi cho việc nghiên cứu và so sánh. Phương pháp huấn luyện kết hợp dữ liệu sẽ được thực hiện với bộ dữ liệu Unraveled do có đủ cả hai loại dữ liệu, còn các bộ dữ liệu còn lại sẽ thực hiện với huấn luyện với network flows. Hình 5.1 biểu diễn các bước huấn luyện mô hình chung được tham khảo từ nghiên cứu [2].



Hình 5.1: Các bước huấn luyện mô hình

a) **Các bước xây dựng mô hình chỉ sử dụng network flows**:

- Thu thập và chuẩn bị dữ liệu: Dữ liệu được thu thập từ các bộ dữ liệu mô phỏng các cuộc tấn công APT, bao gồm các dòng lưu lượng mạng (network flows) với thông tin chi tiết về nguồn, đích, giao thức, cổng kết nối, thời gian và các chỉ số thống kê.
- Tiền xử lý dữ liệu:Do dữ liệu ban đầu thường mất cân bằng nghiêm trọng giữa các nhãn (các giai đoạn tấn công), quá trình resampling được áp dụng: giảm số lượng của các nhãn chiếm nhiều và tăng cường các nhãn hiếm nhằm đảm bảo sự cân bằng trong tập huấn luyện. Một điểm nổi bật của giải pháp là việc tích hợp thông tin ngữ cảnh về vùng mạng (public, middle, private) vào vector đặc trưng. Giả thuyết được đưa ra là hành vi của kẻ tấn công có sự khác biệt rõ rệt tùy vào vùng mạng mà chúng đang hoạt động. Do đó, việc đưa thông tin phân vùng mạng vào mô hình giúp giảm nhầm lẫn giữa các giai đoạn tấn công gần giống nhau về hình thức nhưng khác nhau về ngữ cảnh.
- Lựa chọn mô hình: Nhiều mô hình học máy được thử nghiệm như Random Forest, XGBoost, v.v. Các mô hình này được huấn luyện với dữ liệu đã xử lý, sau đó đánh giá hiệu suất qua các chỉ số như độ chính xác, độ bao phủ và độ tin cậy để chọn ra mô hình tối ưu.
- Đánh giá kết quả mô hình: Mô hình được đánh giá bằng các chỉ số phổ biến như Precision, Recall, F1-score, Macro Avg và Weighted Avg, dựa trên ma trận nhầm lẫn, từ đó đánh giá khả năng phân biệt các giai đoạn tấn công.

b) **Các bước xây dựng mô hình sử dụng network flows kết hợp với host logs**:

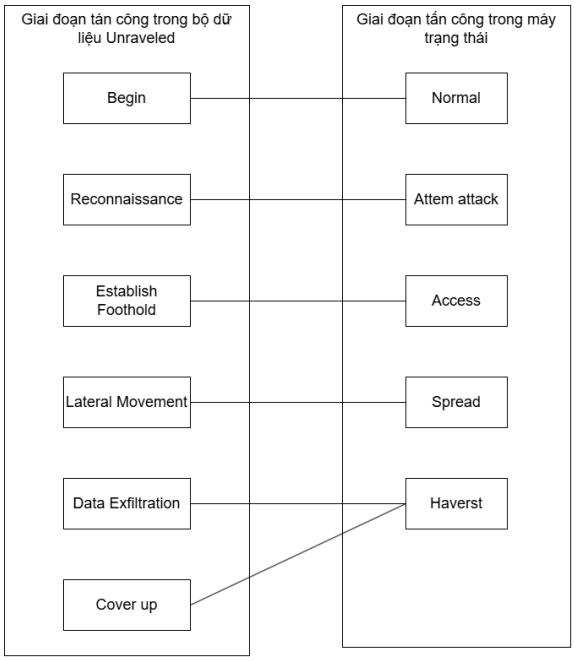
- Thu thập và chuẩn bị dữ liệu: Nguồn dữ liệu bao gồm cả network flows và host logs, được lấy từ các bộ dữ liệu mô phỏng tấn công có chứa các hoạt động cả ở mức mạng và mức hệ thống. Dữ liệu host logs do cấu trúc chưa xác định rõ ràng nên cần phải xử lý bằng cách phân tích nội dung của logs và tách thành dạng có cấu trúc, các feature như type_log, msg_log, exe_log, v.v.

- Tiền xử lý dữ liệu: Trước tiên, hai loại dữ liệu được kết hợp với nhau thông qua việc ghép nối từng record theo địa chỉ IP và thời gian gần nhất, với độ chính xác tính đến từng giây. Bản ghi host logs sẽ được nối vào vector đặc trưng của network flows, tạo thành một mẫu dữ liệu giàu thông tin hơn. Sau khi kết hợp, dữ liệu tiếp tục được xử lý tương tự như với chỉ network flows: cân bằng nhãn, chuẩn hóa dữ liệu, và bổ sung thông tin phân vùng mạng để tăng khả năng phân loại theo ngữ cảnh hoạt động.
- Lựa chọn mô hình và đánh giá kết quả: Quy trình lựa chọn và đánh giá mô hình được tiến hành tương tự như với trường hợp chỉ sử dụng network flows. Tuy nhiên, việc tích hợp thêm dữ liệu host logs giúp có đầy đủ thông tin về dữ liệu từ các góc nhìn khác nhau và cải thiện độ chính xác tổng thể.

5.1.3 Kết quả thực hiện trên bộ dữ liệu Unraveled

a) Phân tích các giai đoạn tấn công

Do các bộ dữ liệu khác nhau có các nhãn về giai đoạn tấn công khác nhau nên cần ánh xạ các giai đoạn tấn công. Đới với bộ dữ liệu Unraveled có xuất hiện 6 giai đoạn tấn công trong hệ thống. Ở đây sẽ cần thực hiện ánh xạ 6 giai đoạn này ứng với 5 giai đoạn tấn công đã được trình bày như trong máy trạng thái của hệ thống. Cụ thể các giai đoạn ánh xạ sẽ được trình bày trong hình 5.2.



Hình 5.2: Ánh xạ các giai đoạn tấn công trong bộ dữ liệu Unraveled với máy trạng thái

Với bộ dữ liệu Unraveled sẽ có 2 kịch bản huấn luyện mô hình: huấn luyện chỉ sử dụng network flows và huấn luyện sử dụng kết hợp network flows và host

logs. Dưới đây sẽ trình bày về kết quả của các phương pháp huấn luyện mô hình khác nhau.

Với kịch bản đầu tiên là cách huấn luyện mô hình chỉ dựa vào netowrk flows có bổ sung thông tin về vùng của hệ thống. Kết quả của mô hình học máy áp dụng phương pháp bổ sung thông tin về vùng của hệ thống so với không sử dụng được trình bày ở bảng dưới. Kết quả được so sánh với nhiều mô hình khác nhau để đưa ra được mô hình có kết quả tốt nhất.

	Sau khi cải thiện			**Trước khi cải thiện**		
	XGBoost	LogisticRegression	RandomForest	XGBoost	LogisticRegression	RandomForest
Begin	**1.00**	0.72	0.99	**0.99**	0.71	0.99
Reconnaissance	**1.00**	0.90	1.00	**1.00**	0.9	0.99
Establish Foothold	**0.96**	0.53	0.96	**0.5**	0.53	0.47
Lateral Movement	**1.00**	0.26	1.00	**0.38**	0.29	0.27
Data Exfiltration & Cover up	**0.99**	0.75	0.99	**0.82**	0.73	0.79
Weight average F1	**0.99**	0.62	0.99	**0.74**	0.61	0.7
Macro average F1	**0.99**	0.61	0.99	**0.74**	0.61	0.7

Bảng 5.1: So sánh hiệu suất các thuật toán trước và sau khi cải thiện với bộ dữ liệu Unraveled

Với kịch bản thứ hai sẽ thực hiện so sánh giữa kết quả của việc áp dụng mô hình có sử dụng kết hợp thông tin của network logs và host logs so với việc chỉ sử dụng dữ liệu của network logs để huấn luyện mô hình. Mô hình sử dụng ở đây là XGBoots để đánh giá:

	Chỉ sử dụng network flows	**Kết hợp cả network flows và host logs**
Begin	**1.00**	1.00
Reconnaissance	**1.00**	x
Establish Foothold	**0.96**	1.00
Lateral Movement	**1.00**	0.65
Data exfiltration & Cover up	**0.99**	0.98
Weight average F1	**0.99**	1.00
Macro average F1	**0.99**	0.86

Bảng 5.2: So sánh hiệu quả phát hiện giữa các phương pháp

Đánh giá chung về kết quả trước và sau khi kết hợp nhiều phương pháp cho thấy được rằng so với việc sau khi cải thiện về việc bổ sung thêm thông tin và kết hợp các loại dữ liệu thì hiệu suất của mô hình được cải thiện rất đáng kể.

Trước khi áp dụng thì kết quả không được cao do bị nhầm lẫn giữa các nhãn. và sau khi cải thiện thì kết quả giữa sử dụng 1 loại dữ liệu so với cả 2 loại mặc dù không khác nhau nhiều về độ chính xác nhưng do dữ liệu kết hợp đang bị hạn chế do thiếu nhãn Reconnaissance ở các thiết bị khi kết hợp 2 loại dữ liệu.

b) Phân tích mức độ phòng thủ

Mức độ phòng thủ trong hệ thống sẽ được chia thành 3 mức đã được nhắc đến trong hình về máy trạng thái của hệ thống gồm có: (i) Detect attack, (ii) Retric attack and apply countermeasures, (iii) Restore and config. Để phân biệt được các mức độ phòng thủ của hệ thống thì ở đây thực hiện bằng cách dựa trên mức thay đổi phần trăm số lượng sự kiện tấn công so với ngày trước đó.

Cho:

- C_t: Số lượng sự kiện tấn công tại thời điểm hiện tại t
- C_{t-1}: Số lượng sự kiện tấn công tại thời điểm trước đó $t-1$

Khi đó, **phần trăm thay đổi** được tính theo công thức:

$$\Delta\% = \frac{C_t - C_{t-1}}{C_{t-1}} \times 100$$

Dựa trên giá trị của $\Delta\%$, trạng thái được xác định như sau:

$$\text{Label}_t = \begin{cases} \text{"Detect attack"} & \text{nếu } \Delta\% \geq 20 \\ \text{"Retric attack and apply countermeasures"} & \text{nếu } \Delta\% \leq -20 \\ \text{Label}_{t-1} & \text{trong các trường hợp khác} \end{cases}$$

c) Phân loại đối tượng tấn công

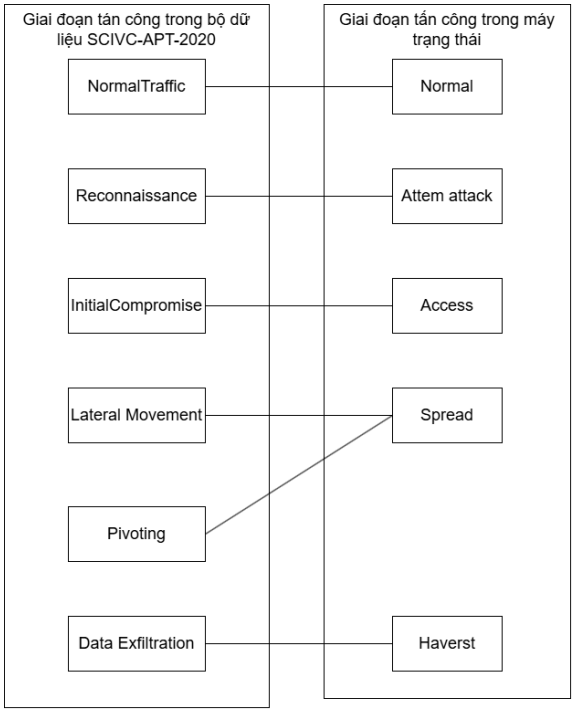
Trong bộ dữ liệu Unraveled thì các loại đối tượng tấn công sẽ gồm có: (i) AA, (ii) SH, (iii) APT đã được gán nhãn ở cột "Signature". Để phân loại được các phần dữ liệu nào là do đối tượng tấn công nào gây ra thì ở đây sẽ xây dựng mô hình học máy để phân biệt. Do có bổ sung thêm thông tin về các phân vùng public, middle, private nên giúp cho phân vùng các đối tượng được dễ dàng hơn. Giới hạn của các đối tượng sẽ bị hạn chế bởi các vùng có mức độ bảo mật cao. Bảng 5.5 là kết quả của các mô hình học máy được sử dụng trong việc phân loại các đối tượng tấn công.

	XGBoost	RandomForest	Logistic Regression
AA	**0.99**	0.99	0.92
APT	**1.00**	1.00	1.00
Weight average F1	**0.99**	0.99	0.97
Macro average F1	**0.99**	0.99	0.96

Bảng 5.3: Kết quả phân loại các đối tượng tấn công

5.1.4 Kết quả thực hiện trên bộ dữ liệu SCIVC-APT-2020

Hình 5.3 ánh xạ các giai đoạn tấn công của bộ dữ liệu SCIVC-APT-2020 với các giai đoạn tấn công xuất hiện trong máy trạng thái. Trong bộ dữ liệu SCIVC-APT-2020 gồm có 6 giai đoạn liên quan đến cuộc tần công. Để đưa ra được sự quát giữa các bộ dữ liệu các giai đoạn tấn công đó sẽ được ánh xạ sang 5 giai đoạn tấn công mà đã được trình bày giống như trong máy trạng thái của hệ thống.



Hình 5.3: Ánh xạ các giai đoạn tấn công trong bộ dữ liệu SCIVC-APT-2020 với máy trạng thái

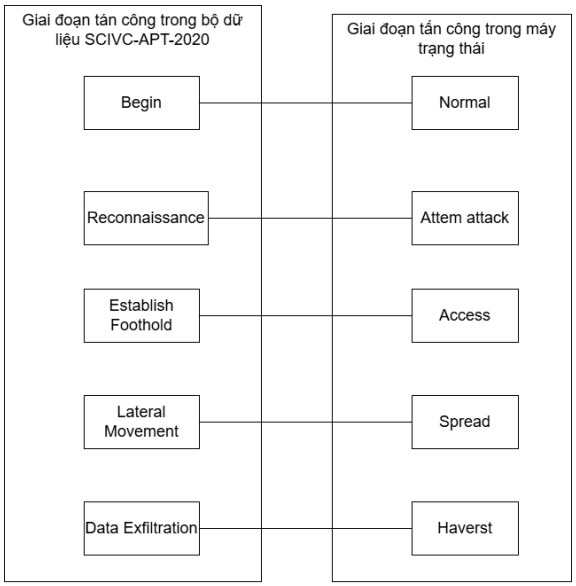
Dưới đây là kết quả khi mà thực hiện so sánh các kết quả của mô hình học máy XGBoost và RandomForest sau khi mà áp dụng phương pháp xử lý, bổ sung thêm thông tin dữ liệu với không sử dụng phương pháp cải tiến. Có thể thấy rằng kết quả đạt được khá cao nhưng do ở nhãn Data Exfiltration & Cover up thì kết quả có bị kém do là số lượng mẫu quá ít chỉ khoảng 600 mẫu dữ liệu.

	Sau khi cải thiện		**Trước khi cải thiện**	
	XGBoost	RandomForest	XGBoost	RandomForest
Benign	**1.00**	1.00	**1.00**	1.00
Reconnaissance	**0.85**	0.85	**0.85**	0.85
InitialCompromise	**1.00**	0.96	**1.00**	0.96
Lateral Movement & Pivoting	**0.92**	0.91	**0.92**	0.91
Data Exfiltration	**0.65**	0.6	**0.65**	0.6
Weight average F1	**0.87**	0.84	**0.87**	0.84
Macro average F1	**1.00**	1.00	**1.00**	1.00

Bảng 5.4: So sánh hiệu suất các thuật toán trước và sau khi cải thiện đối với bộ dữ liệu SCIVC-APT-2020

5.1.5 Kết quả thực hiện trên bộ dữ liệu DAPT-2020

Hình 5.3 ánh xạ các giai đoạn tấn công của bộ dữ liệu DAPT-2020 với các giai đoạn tấn công xuất hiện trong máy trạng thái. Trong bộ dữ liệu DAPT-2020 gồm có 5 giai đoạn liên quan đến cuộc tần công. Để đưa ra được sự quát giữa các bộ dữ liệu các giai đoạn tấn công đó sẽ được ánh xạ sang 5 giai đoạn tấn công mà đã được trình bày giống như trong máy trạng thái của hệ thống.



Hình 5.4: Ánh xạ các giai đoạn tấn công trong bộ dữ liệu DAPT-2020 với máy trạng thái

Dưới đây là kết quả khi mà thực hiện so sánh các kết quả của mô hình học máy XGBoost và RandomForest sau khi mà áp dụng phương pháp xử lý, bổ sung thêm thông tin dữ liệu với không sử dụng phương pháp cải tiến. Có thể thấy mặc dù kết quả có kém hơn so với 2 bộ dữ liệu trên nhưng cũng khá cao trên 80% đối với mô hình học máy XGBoots.

Loại tấn công	Sau khi cải thiện		Trước khi cải thiện	
	XGBoost	RandomForest	XGBoost	RandomForest
Benign	**0.98**	0.97	**0.96**	0.95
Reconnaissance	**0.85**	0.82	**0.80**	0.80
Establish Foothold	**0.92**	0.90	**0.88**	0.88
Lateral Movement	**0.87**	0.78	**0.83**	0.71
Data Exfiltration & Cover up	**0.79**	0.86	**0.75**	0.83
Weight average F1	**0.84**	0.85	**0.78**	0.80
Macro average F1	**0.88**	0.87	**0.84**	0.83

Bảng 5.5: So sánh hiệu suất các thuật toán trước và sau khi cải thiện với bộ dữ liệu DAPT-2020

5.2 Chức năng trực quan hoá và phân tích dữ liệu tấn công

5.2.1 Vấn đề

Chức năng trực quan hoá và phân tích dữ liệu tấn công được tích hợp vào hệ thống RIDX sẽ giúp cho người sử dụng có các góc nhìn chi tiết, đầy đủ về hệ thống, các thành phần của cuộc tấn công. Các thông tin về hệ thống cần thể hiện như về thiết kế hệ thống, cấu hình phân tích cho hệ thống. Ngoài ra, với các mô hình học máy sử dụng cho việc phát hiện các thông tin tấn công APT từ dữ liệu network flows thì cũng cần phải trực quan hoá thông tin đã phân tích được. Các thông tin sẽ cần phải thể hiện diễn biến của cuộc tấn công tại các thời điểm hoặc trong từng khoảng thời gian.

5.2.2 Giải pháp

Để có thể phân tích và trực quan hoá các thông tin thì dữ liệu sẽ được thể hiện quá các biểu đồ trực quan hoá. Với các thông tin về hệ thống sẽ hiển thị bao gồm: cấu trúc hệ thống, các phân vùng của hệ thống được phân chia dựa vào mức độ bảo mật, quyền truy cập vào hệ thống đối với các tài sản thiết kế trong hệ thống.

Các kết quả phân tích được từ dữ liệu mạng do các mô hình học máy dự đoán ra sẽ được thể hiện thông qua các biểu đồ. Các dữ liệu phân tích ra sẽ thể hiện sự biến đổi về các thành phần của cuộc tấn công: bên tấn công, bên phòng thủ, đối tượng tấn công. Ngoài ra để thể hiện rõ hơn về quá trình biến đổi thì chức năng hiện thị kết quả theo các khoảng thời gian trong quá khứ. Chi tiết sẽ được thể hiện như dưới đây.

a) Xây dựng Socket API để xử lý truyền dữ liệu với kích thước lớn Do dữ liệu cho quá trình phân tích có kích thước lớn và gồm nhiều các thông tin khác nhau như: kịch bản triển khai, dữ liệu mạng, v.v. Nên để tránh việc phải truyền nhiều

lần gây tốn thời gian thì sẽ xây dựng Socket API để truyền dữ liệu với các ưu điểm chính:

- Tối ưu về thời gian: Do kiến trúc hệ thống hiện tại làm microservices nên các API khi gọi sẽ đều thông qua API gateway xử lý rồi truyền đến các thành phần khác. Việc xây dựng Socket API ở đây sẽ kết nối trực tiếp giữa người dùng và phần xử lý chính ở core service giúp cho giảm thiểu bước xử lý.
- Tăng kích thước dữ liệu được truyền: Các dữ liệu về kịch bản triển khai gồm nhiều các thông tin về tài sản, mối quan hệ tài sản có kích thước lớn mà khi truyền qua API thông thường cần rất nhiều thời gian xử lý và có thể là không truyền được do kích thước vượt quá giới hạn. Để xử lý việc này thì Socket API sẽ chia gói tin thành các phần nhỏ và truyền đi để xử lý. Khi nhận được các gói tin sẽ được ghép lại như ban đầu.
- Nâng cao hiệu suất: Việc phải gọi API nhiều lần để truyền các thông tin sẽ cần nhiều thời gian để có thể xử lý và truyền đi qua nhiều các thành phần. Sử dụng Socket API sẽ có tốc độ nhanh và đơn giản hơn trong trường hợp dữ liệu có kích thước lớn.

b) Trực quan hoá kết quả phân tích cuộc tấn công

Màn hình trực quan hoá kết quả phân tích sẽ bao gồm:

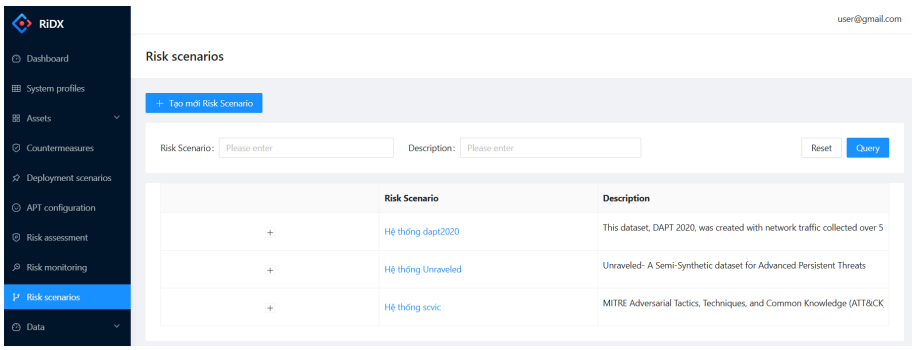
- Thông tin về cấu hình hệ thống triển khai của ứng với bộ dữ liệu: Thông tin về sơ đồ cấu trúc hệ thống, thông tin về các tài sản và các phân vùng được chia (public, middle, private) từ cấu trúc hệ thống.
- Thông tin rủi ro trên các tài sản: Chưa thông tin về địa chỉ IP, các kết quả từ việc đánh giá rủi ro như serverity, likelihood, risk level mà đã đánh giá được từ dữ liệu mạng.
- Các biểu đồ thống kê kết quả phân tích theo các mốc thời gian: CÁc biểu đồ về tỷ lệ giai đoạn tấn công phát hiện ra và thay đổi của mức độ phòng thủ trên các mốc thời gian, thông tin phân loại đối tượng tấn công.
- Biểu đồ phân tích kết quả theo windows time: Cho phép người dùng nhập dữ liệu đầu vào là windows time và khoảng cách từ đó xây dựng biểu đồ thống kê tỷ lệ phần trăm của các nhãn giai đoạn tấn công trong khoảng thời gian và bước tiếp theo các bước nhảy.

5.2.3 Kết quả

Dưới đây là các màn hình xây dựng cho chức năng trực quan hoá và phân tích dữ liệu tấn công.

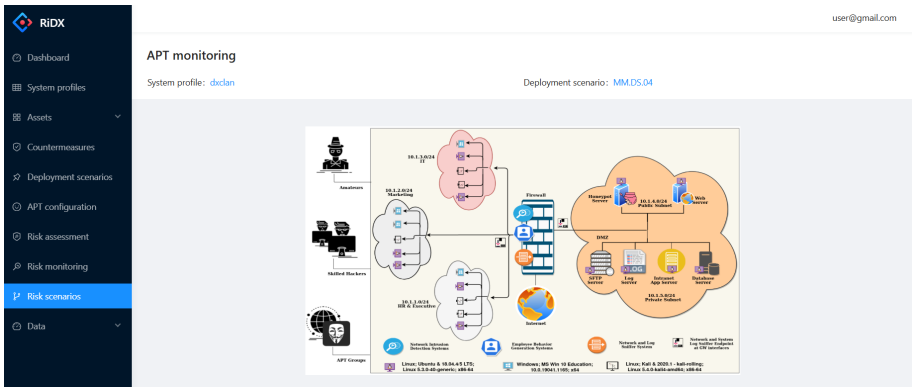
a) Các màn hình hiển thị Dashboard

Hình 5.5 hiển thị sẽ bao gồm các danh sách của kịch bản rủi ro gồm tên và các mô tả về hệ thống. Ngoài ra cũng có chức năng tạo mới kịch bản rủi ro khi ấn vào nút tạo mới kịch bản rủi ro.



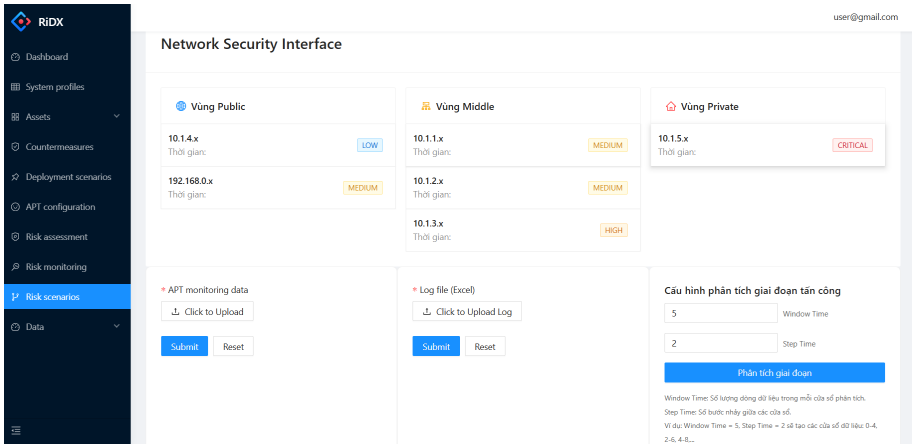
Hình 5.5: Màn hình danh sách kịch bản rủi ro

Hình 5.6 chứa thông tin về thiết kế của hệ thống đang xây dựng cho kịch bản rủi ro. Các hệ thống này có kiến trúc khá tổng quát có thể làm tài liệu tham khảo cho các hệ thống khác tương tự. Ngoài ra cũng chứa thông tin về tên system profile gồm các thông tin về tài sản của hệ thống đã xây dựng, tên của kịch bản triển khai.



Hình 5.6: Màn hình thông tin thiết kế của hệ thống

Hình 5.7 chứa thông tin địa chỉ IP đã được phân vào các vùng của hệ thống đã cấu hình dựa vào quyền truy nhập, bảo mật dữ liệu. Bên dưới là phần upload dữ liệu mạng, dữ liệu log, lựa chọn windows time và step để đánh giá kết quả phân tích.

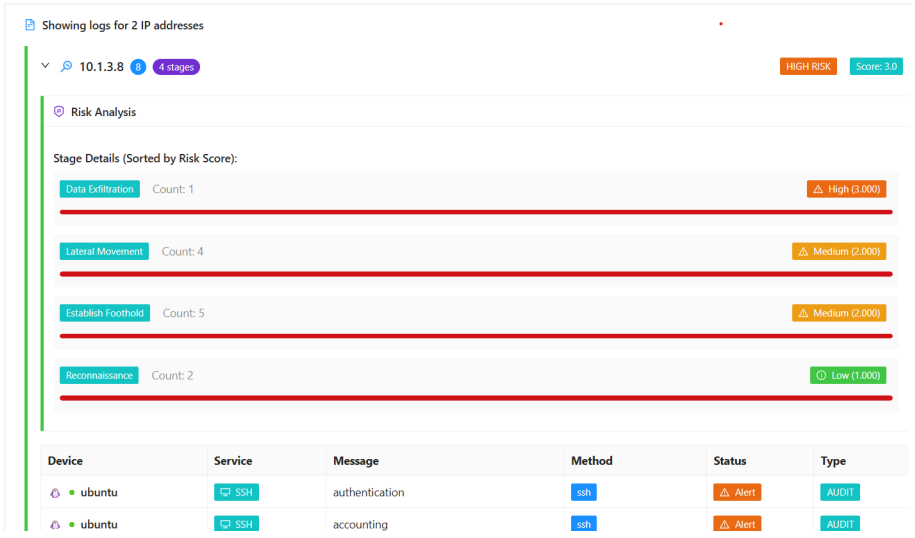


Hình 5.7: Màn hình thông tin thiết kế của hệ thống

b) Màn hình trực quan hoá kết quả phân tích

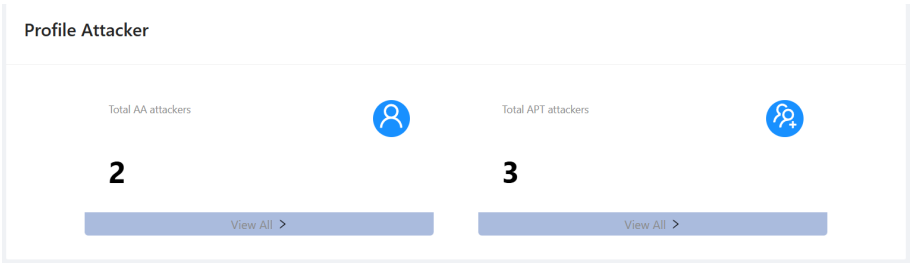
Tiếp theo sau khi đã khởi tạo kịch bản rủi ro và upload các dữ liệu thì dưới đây là phần trực quan hoá các kết quả phân tích.

Hình 5.8 trình bày kết quả đánh giá rủi ro theo từng tài sản. Khi người dùng tải lên dữ liệu host logs, hệ thống sẽ dựa trên kết quả đánh giá rủi ro đã được phân tích từ dữ liệu network flows trước đó để xác định và hiển thị mức độ rủi ro tương ứng cho các tài sản xuất hiện trong logs.



Hình 5.8: Màn hình rủi ro theo tài sản

Hình 5.9 cho biết các đối tượng phân loại được trong quá tình tấn công. Các đối tượng ở đây gồm có amature attacker và apt attacker. Các thông tin hiển thị bao gồm các lần mà đối tượng có thay đổi về giai đoạn tấn công. Khi ấn vào từng đối tượng sẽ cho phép hiển thị chi tiết về đánh giá theo từng đối tượng.

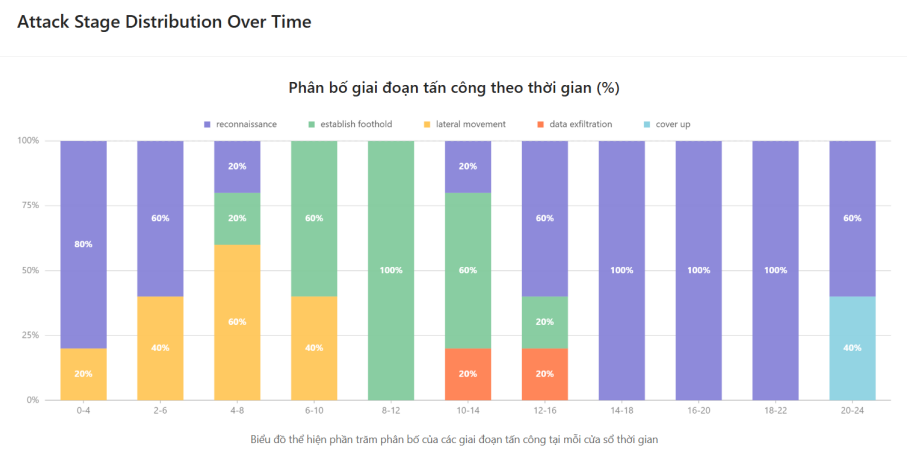


Hình 5.9: Màn hình thông tin về đối tượng tấn công

Hình 5.10 chứa 2 biểu đồ về kết quả phân tích về tỷ lệ các giai đoạn tấn công và mức độ phòng thủ theo các mốc thời gian. Biểu đồ thứ nhất là tỷ lệ phần trăm các giai đoạn tấn công xuất hiện theo các mốc thời gian đã đánh dấu. Biểu đồ thứ hai biểu diễn sự thay đổi về mức độ phòng thủ của hệ thống theo các mốc thời gian.

Hình 5.10: Màn hình kết quả phân tích tấn công và mức độ phòng thủ

Hình 5.11 là biểu đồ biểu diễn tỷ lệ phần trăm của các giai đoạn trong 1 time windows. Với cách đánh giá này sẽ cho góc nhìn rõ ràng về sự biến đổi của cuộc tấn công theo các time windows với khoảng cách từng step do từng time windows có liên quan đến nhau theo tuần tự.



Hình 5.11: Màn hình kết quả phân tích tấn công theo windows time

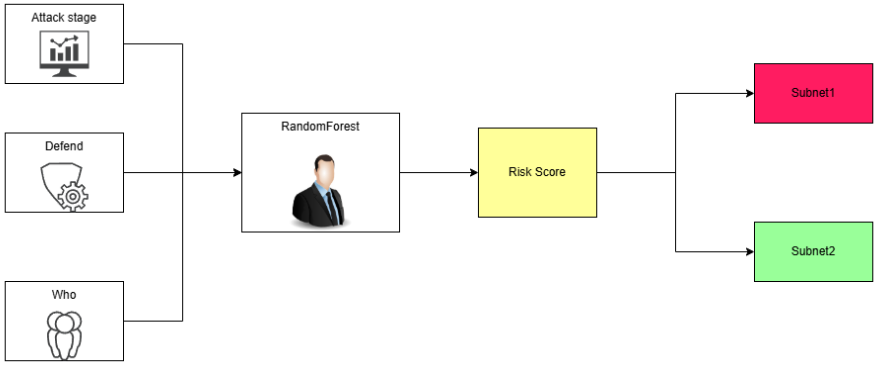
5.3 Xây dựng chức năng đánh giá điểm cho cuộc tấn công

5.3.1 Vấn đề

Hiện tại hệ thống RIDX đã phát triển chức năng đánh giá rủi ro của hệ thống trước khi đưa vào triển khai thông qua sử dụng thông tin về deployment scenario. Khi thực hiện đánh giá thì người dùng sẽ cần phải cung cấp đầu vào cho các mạng subnet1 và subnet2 của hệ thống là điểm số đánh giá về bên tấn công và phòng thủ. Như vậy để đánh giá mỗi lần rất tốn công sức khi đưa vào triển khai thực tế. Vì thế cần phải có cơ chế đánh giá điểm tự động nhằm nâng cao tính chính xác và giảm thiểu công sức khi thực hiện đánh giá rủi ro.

5.3.2 Giải pháp

Để có thể đánh giá điểm về bên tấn công, phòng thủ của hệ thống tự động thì cần phải có cơ chế ánh xạ từ thông tin mà các mô hình học máy đã tổng được về cuộc tấn công sau đó ánh xạ thành các điểm đánh giá. Ở đây sẽ thực hiện bằng cách xây dựng mô hình học máy RandomForest, để đánh giá và đưa ra điểm cụ thể sẽ được trình bày như ở hình dưới.



Hình 5.12: Mô tả đánh giá điểm subnet1 và subnet2

Cụ thể trong hình 5.12 thì gồm có đầu vào là các thông tin về giai đoạn tấn công, mức độ phòng thủ, đối tượng tấn công mà hệ thống phát hiện được bằng sử dụng mô hình học máy đã trình bài ở phần 5.1. Đầu ra là điểm Risk Score sẽ là điểm đánh giá mức độ rủi ro của hệ thống từ các thông tin trên. Và với điểm risk score thì hệ thống sẽ đưa ra điểm đánh giá cho subnet 1 và subnet 2 theo quy luật.

a) Mô hình học máy đánh giá điểm tự động

Sau đây sẽ trình bày về các bước xây dựng mô hình học máy để đánh giá điểm tự động cho kịch bản rủi ro:

- Xây dựng dữ liệu: Để xây dựng dữ liệu huấn luyện thì cần có hai phần là dữ liệu và nhãn của dữ liệu. Với các dữ liệu huấn luyện thì sẽ gồm tất cả các trường hợp xảy ra khi kết hợp các nhãn giai đoạn tấn công, mức độ phòng thủ và đối tượng tấn công. Nhãn ở đây sẽ được đánh giá theo tri thức của chuyên gia với các quy luật được trình bày dưới đây.

  Công thức về mức độ tấn công mà các đối tượng có thể thực hiện:

$$\begin{aligned} \text{AA} &\Leftrightarrow S \in \{0, 1\} \\ \text{SH} &\Leftrightarrow S \in \{0, 1, 2, 3\} \\ \text{APT} &\Leftrightarrow S \in \{0, 1, 2, 3, 4\} \end{aligned}$$

  Công thức về đánh giá điểm A1-A5 từ giai đoạn tấn công:

$$\forall t \in T, \quad S(t) = \begin{cases} \{0, 1\} & \text{nếu } t = \text{AA} \\ \{0, 1, 2, 3\} & \text{nếu } t = \text{SH} \\ \{0, 1, 2, 3, 4\} & \text{nếu } t = \text{APT} \end{cases}$$

$$\begin{aligned} \text{Giai đoạn } 1 \text{ (Attempt)} : &\quad \max(\text{A1–A5}) \leq 3 \\ \text{Giai đoạn } 2 \text{ (Access)} : &\quad 3 \leq \text{A1–A5} \leq 5 \\ \text{Giai đoạn } 3 \text{ (Spread)} : &\quad 3 \leq \text{A1–A5} \leq 5 \\ \text{Giai đoạn } 4 \text{ (Harvest)} : &\quad \min(\text{A1–A5}) \geq 7 \end{aligned}$$

Công thức về đánh giá điểm D1-D5 từ mức độ phòng thủ:

$$f(R,S,D) = \begin{cases} D \geq 8 & \text{nếu } R = 0 \wedge S = 1 \\ 6 \leq D \leq 8 & \text{nếu } R = 0 \wedge S = 2 \\ 4 \leq D \leq 6 & \text{nếu } R = 0 \wedge S = 3 \\ 2 \leq D \leq 4 & \text{nếu } R = 0 \wedge S = 4 \\ D \geq 7 & \text{nếu } R = 1 \wedge S = 1 \\ 6 \leq D \leq 8 & \text{nếu } R = 1 \wedge S = 2 \\ 4 \leq D \leq 6 & \text{nếu } R = 1 \wedge S = 3 \\ D \leq 4 & \text{nếu } R = 1 \wedge S = 4 \\ D \geq 8 & \text{nếu } R = 2 \end{cases}$$

- Xử lý dữ liệu: Sau khi có được bảng dữ liệu có cấu trúc thì thực hiện chuyển các nhãn thành dạng số nhằm chuẩn bị cho bước tiếp theo huấn luyện mô hình.
- Huấn luyện mô hình: Mô hình được sử dụng ở đây là RandomForest. Dữ liệu sẽ được đưa vào để huấn luyện.
- Lưu mô hình: Sau khi đã huấn luyện mô hình thì sẽ được lưu lại kết quả để tích hợp vào hệ thống.

b) Màn hình đánh giá điểm cho kịch bản rủi ro

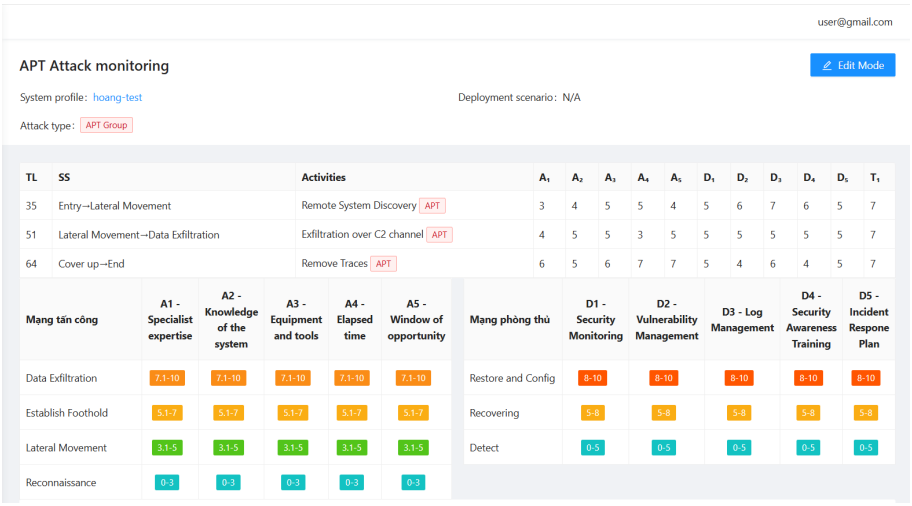
Với màn hình kết quả cho chức năng đánh giá điểm cho kịch bản rủi ro sẽ được xây dựng gồm:

- Kịch bản rủi ro: Gồm có thông tin được tóm gọn từ dữ liệu được phân tích và đánh giá. Các mốc quan trọng mà đối tượng thay đổi trạng thái tấn công, điểm đánh giá cho subnet 1 và 2.
- Kết quả đánh giá rủi ro theo kịch bản rủi ro: Biểu đồ về chỉ số severity, biểu đồ về chỉ số likelihood, biểu đồ về chỉ số risk level.

5.3.3 Kết quả

a) Màn hình hiển thị thông tin điểm đánh giá

Chức năng điểm đánh giá sẽ được trình bày ở trong hệ thống RIDX khi mà người dùng đã upload dữ liệu mạng về cuộc tấn công và kết quả điểm sẽ được trình bày ở trong thông tin của đối tượng tấn công.

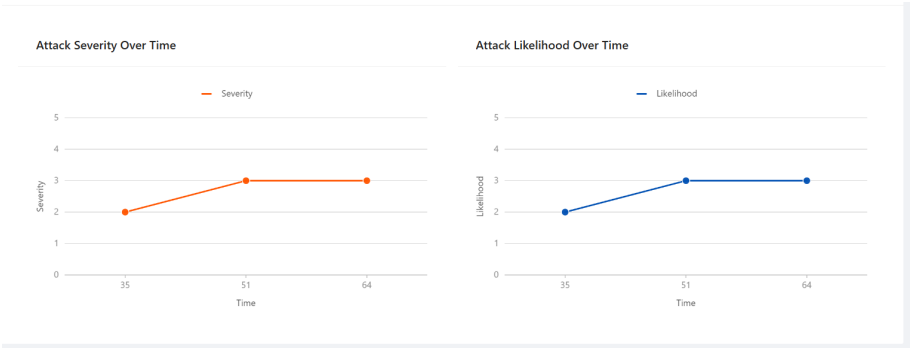


Hình 5.13: Màn hình đánh giá điểm subnet 1 và subnet 2

Đối với điểm đánh giá về các thành phần nếu người dùng có thể thấy điểm đánh giá chưa được tốt thì có thể sửa đổi được điểm đánh giá và gửi lại. Khi đó hệ thống sẽ tính toán lại theo điểm đánh giá mới nhận được và trả về kết quả tương ứng. Bên dưới là các mốc điểm đánh giá giúp cho người dùng tham khảo đối với các giai đoạn mà hệ thống phát hiện ra.

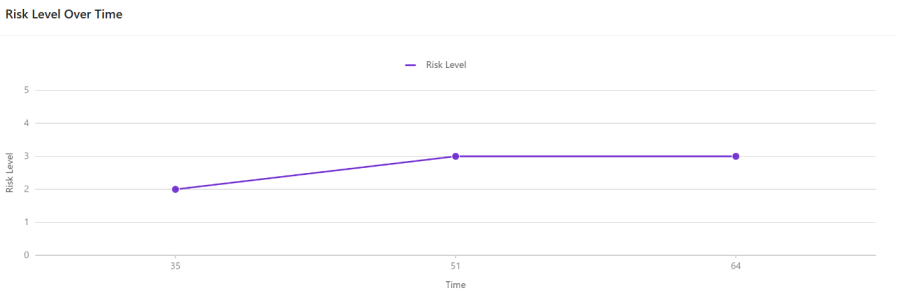
b) Màn hình kết quả quá trình đánh giá

Sau khi đánh giá điểm cho hệ thống thì kết quả đánh giá ở đây sẽ là severity và likelihood thể hiện mức độ rủi ro của hệ thống được biểu diễn như hình 5.14.



Hình 5.14: Màn hình kết quả đánh giá rủi ro severity và likelihood

Hình 5.15 là biểu đồ về kết quả đánh giá risk level sau khi đánh giá rủi ro. Thông số của risk level sẽ được tổng hợp theo thông số severity và likelihood theo công thức bằng tích của chúng.



Hình 5.15: Màn hình kết quả đánh giá rủi ro risk level

CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Kết luận

Đóng góp chính của đồ án bao gồm 2 chức năng chính là: (i) Phân tích thông tin tấn công bằng các mô hình học máy và (ii) Đánh giá điểm kịch bản rủi ro cuộc tấn công cho mạng Bayes đánh giá rủi ro. Về cơ bản các chắc năng đã đạt được hiệu quả cao trên các bộ dữ liệu mô phỏng ở chức năng xây dựng các mô hình học máy. Kết quả của mô hình tốt hơn so với các mô hình được xây dựng trên cùng bộ dữ liệu. Còn một số vấn đề là về việc khi tích hợp dữ liệu host logs và network flows mới thực hiện được trên bộ dữ liệu Unraveled do giới hạn về dữ liệu mà các bộ cung cấp. Về chức năng đánh giá điểm cho cuộc tấn công thì điểm đánh giá chưa được khách quan với đúng ngữ cảnh xảy ra.

Quá trình thực hiện đồ án giúp tôi học được nhiều kiến thức về ATTT, kỹ năng làm việc chuyên nghiệp, v.v. Đây cũng như là sự chuẩn bị trước khi bước vào môi trường doanh nghiệp.

6.2 Hướng phát triển

Các chức năng đã phát triển trong đồ án gồm:

- Phân tích thông tin tấn công bằng các mô hình học máy.
- Trực quan hoá các kết quả phân tích của mô hình học máy.
- Đánh giá điểm cuộc tấn công cho mạng Bayes đánh giá rủi ro và đưa ra kết quả đánh giá.

Với chức năng sử dụng các mô hình học máy để phân tích thông tin tấn công tự động mắc dù cho kết quả rất cao nhưng trong thực tế các dữ liệu về tấn công sẽ khó phát hiện hơn và có nhiều các phương pháp mới hơn để che dấu. Vì thế cần bổ sung quy trình tự động thu thập dữ liệu mới để huấn luyện lại mô hình từ đó giúp phát hiện được tốt hơn. Thêm nữa với chức năng đánh giá điểm cho cuộc tấn công cũng cần phải xây dựng bộ luật chặt chẽ hơn, mức điểm đưa ra được đa dạng hơn để phù hợp với các hành động tấn công.

TÀI LIỆU THAM KHẢO

[1] V. T. H. Giang and N. M. Tuan, "APT Risk Assessment with Hybrid Bayesian Network for DevOps-based Web Systems", *Information Security Journal: A Global Perspective*, 2025. Submission ID: 258478022.

[2] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] ReactJS, "React", 2024. Available: *https://react.dev/*.

[4] Ant Design, "A UI Design Language", 2024. Available: *https://ant.design/*.

[5] UmiJS Team, "UmiJS: Pluggable Enterprise-level React Application Framework", 2024. Available: *https://umijs.org/*.

[6] Node.js Foundation, "Node.js – JavaScript Runtime Built on Chrome's V8 Engine", 2024. Available: *https://nodejs.org/*.

[7] "NestJS – A Progressive Node.js Framework", 2024. Available: *https://nestjs.com/*.

[8] MongoDB Inc., "MongoDB – The Developer Data Platform", 2024. Available: *https://www.mongodb.com/*.

[9] BayesFusion, LLC, "PySMILE – Python Interface to SMILE Bayesian Network Library", 2024. Available: *https://www.bayesfusion.com/docs/pysmile/*.

[10] S. Ramírez, "FastAPI – Modern, Fast (High-performance), Web Framework for Building APIs with Python", 2024. Available: *https://fastapi.tiangolo.com/*.

[11] Scikit-learn, "Scikit-learn: Machine Learning in Python", 2024. Available: *https://scikit-learn.org/*.

[12] S. Myneni, K. Jha, A. Sabur, G. Agrawal, Y. Deng, A. Chowdhary, and D. Huang, "Unraveled: A Semi-Synthetic Dataset for Advanced Persistent Threats", *Computer Networks*, vol. 227, p. 109688, 2023.

[13] H. Ring, K. McLaughlin, S. Sezer, et al., "DAPT2020: Dataset for Advanced Persistent Threat Detection", in *Proceedings of the 7th International Symposium for ICS & SCADA Cyber Security Research*, 2020.

[14] J. Liu, Y. Shen, M. Simsek, B. Kantarci, H. Mouftah, M. Bagheri, and P. Djukic, "A New Realistic Benchmark for Advanced Persistent Threats in Network Traffic", in *Proceedings of the 16th International Conference on Availability, Reliability and Security*, 2021. Available: *https://ieee-dataport.org/documents/scvic-apt-2021*.

[15] National Institute of Standards and Technology (NIST), “National Vulnerability Database”, Apr. 2018. Available: *https://nvd.nist.gov/*.

[16] Python Software Foundation, “Python”, 2024. Available: *https://www.python.org/*.

[17] V. T. H. Giang and N. M. Tuan, “Application of Bayesian Network in Risk Assessment for Website Deployment Scenarios”, *Journal of Science and Technology on Information Security*, vol. 2, no. 14, pp. 3–17, 2021.