BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐH SƯ PHẠM KỸ THUẬT TPHCM**

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN**

**XÂY DỰNG HỆ THỐNG PHÁT HIỆN XÂM NHẬP MẠNG IDS SỬ DỤNG MACHINE LEARNING**

**SV2024-190**

**Chủ nhiệm đề tài: Nguyễn Thắng Lợi**

**TP Hồ Chí Minh, 11/2024**

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐH SƯ PHẠM KỸ THUẬT TPHCM**

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN**

**XÂY DỰNG HỆ THỐNG PHÁT HIỆN XÂM NHẬP MẠNG IDS SỬ DỤNG MACHINE LEARNING**

**SV2024-190**

Thuộc nhóm ngành khoa học: an ninh mạng

SV thực hiện: Nguyễn Thắng Lợi Nam, Nữ: Nam

Dân tộc: Kinh

Lớp, khoa: 22162A, Công Nghệ Thông Tin Năm thứ: 2/4

Ngành học: An Toàn Thông Tin

Người hướng dẫn: ThS. Nguyễn Thị Thanh Vân

**TP Hồ Chí Minh, 11/2024**

# **MỤC LỤC**

[MỤC LỤC I](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc6316)

[DANH MỤC HÌNH ẢNH II](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc25590)

[DANH MỤC BẢNG III](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc14031)

[PHẦN MỘT: MỞ ĐẦU 1](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc31223)

[1. Lý do chọn đề tài 1](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc17726)

[2. Mục tiêu đề tài 1](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc19082)

[3. Phương pháp nghiên cứu 2](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc7406)

[4. Đối tượng và phạm vi nghiên cứu 2](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc30679)

[PHẦN 2: NỘI DUNG 3](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc17937)

[CHƯƠNG 1: TỔNG QUAN VỀ TẤN CÔNG MẠNG VÀ CÁC PHƯƠNG PHÁP PHÁT HIỆN PHỔ BIẾN 3](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc32499)

[1.1. Giới thiệu về tấn công mạng 3](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc26701)

[1.2. Các dạng tấn công phổ biến 3](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc13359)

[1.3. Cách nhận diện một cuộc tấn công mạng 6](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc11499)

[1.3.1. Dựa vào các gói tin (packets) 6](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc4353)

[1.3.2. Phát hiện dựa trên dòng dữ liệu bất thường 7](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc3985)

[1.4. Các phương pháp phát hiện tấn công mạng 7](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc4696)

[1.4.1. Phát hiện dựa trên sự bất thường (Anomaly-based Detection) 8](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc20943)

[1.4.2. Phát hiện dựa trên dấu hiệu tấn công (Signature-based Detection) 8](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc12919)

[1.4.3. Phát hiện dựa trên hành vi (Behavioral Detection):  9](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc4097)

[CHƯƠNG 2: CÁC CÔNG TRÌNH NGHIÊN CỨU LIÊN QUAN 11](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc8911)

[CHƯƠNG 3: HỆ THỐNG PHÁT HIỆN XÂM NHẬP MẠNG 14](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc10835)

[3.1. Giới thiệu về hệ thống phát hiện xâm nhập mạng 14](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc19817)

[3.2. Cấu trúc của IDS 16](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc2804)

[3.3. Phân loại IDS 17](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc8926)

[3.4. Các thuật toán học máy được áp dụng phổ biến trong hệ thống IDS 19](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc23215)

[3.4.1 Random Forest 19](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc8444)

[3.4.2 K-nearest neighbors (KNN) 20](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc32648)

[3.4.3 Support vector machine (SVM) 21](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc8057)

[3.4.4 K-mean clustering 22](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc23828)

[3.4.5 Artificial neural network 23](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc7021)

[3.4.6 Voting and stacking model 24](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc19607)

[CHƯƠNG 4: XÂY DỰNG HỆ THỐNG PHÁT HIỆN XÂM NHẬP MẠNG SỬ DỤNG MACHINE LEARNING 25](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc16231)

[4.1. Kiến trúc hệ thống IDS đề xuất 25](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc13119)

[4.2. Xử lý dữ liệu từ tập dữ liệu CIC-IDS 2018 26](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc5461)

[4.2.1. Tổng quan về tập dữ liệu và các tùy chỉnh 26](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc3815)

[4.2.2 Tiền xử lý và chuẩn hóa dữ liệu 31](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc30882)

[4.3. Xây dựng và huấn luyện mô hình 33](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc22093)

[4.3.1 Thiết kế mô hình 33](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc21035)

[4.3.2. Huấn luyện mô hình và tinh chỉnh tham số 35](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc12193)

[4.4 Xây dựng hệ thống IDS 36](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc13567)

[4.4.1 Data Gathering 36](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc22737)

[4.4.2. Backend API 39](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc27146)

[4.4.3 Giao diện Quản lý (Management Interface) 41](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc14085)

[CHƯƠNG 5: THỰC NGHIỆM VÀ ĐÁNH GIÁ 43](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc10769)

[5.1. Đánh giá mô hình học máy 43](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc28452)

[5.2. Đánh giá hiệu suất hệ thống IDS 46](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc180)

[5.3. Ưu và nhược điểm của hệ thống: 47](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc14171)

[5.3.1. Ưu điểm 47](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc26699)

[5.3.2. Nhược điểm 47](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc20988)

[PHẦN 3: KẾT LUẬN 48](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc28120)

[1. Kết quả nghiên cứu của đề tài 48](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc25645)

[2. Hạn chế của sản phẩm 48](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc25958)

[3. Hướng phát triển 49](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc12096)

[TÀI LIỆU THAM KHẢO 50](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc14992)

[PHỤ LỤC 54](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc10675)

[1. Mã nguồn dùng để tùy chỉnh dữ liệu gốc 54](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc1538)

[2. Mã nguồn tiền xử lý và chuẩn hóa dữ liệu 55](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc7451)

[3. Mã nguồn thiết kế mô hình học máy 55](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc6190)

[4. Mã nguồn huấn luyện và tinh chỉnh tham số mô hình 56](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc22671)

[5. Mã nguồn Data Gathering 57](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc27280)

[6. Mã nguồn Backend API 59](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc17232)

[7. Mã nguồn Giao diện Quản lý 60](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc13368)

# **DANH MỤC HÌNH ẢNH**

# 

[Hình 1 : Nguyễn lý cơ bản của IDS [20] 14](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc5317)

[Hình 2 : Cấu trúc của IDS [19] 16](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc9067)

[Hình 3 : Nguyên lý hoạt động của SIDS [21] 18](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc7210)

[Hình 4 : Nguyên lý hoạt động AIDS [21] 19](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc19350)

[Hình 5 : Mô tả hoạt động Random Forest 20](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc25864)

[Hình 6 : Mô tả KNN bằng đồ thị. 21](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc27267)

[Hình 7 : Thuật toán SVM. 22](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc24724)

[Hình 8 : Phân cụm trong thuật toán K-Means 23](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc5688)

[Hình 9 : Mô Hình Artificial neural network. 24](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc18719)

[Hình 10 : Sơ đồ kiến trúc hệ thống IDS 25](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc28857)

[Hình 11 : Mô hình mạng mà tập dữ liệu CIC-IDS được thu thập [27] 27](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc3822)

[Hình 12 : Các thuật toán trong mô hình học máy 33](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc7036)

[Hình 13 : Sơ đồ hoạt động của hệ thống Backend API - ML 39](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc1862)

# **DANH MỤC BẢNG**

[Bảng 1 : Danh sách nghiên cứu về thuật toán Decision Tree 11](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc17421)

[Bảng 2 : Danh sách nghiên cứu về thuật toán SVM 12](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc16782)

[Bảng 3 : Danh sách nghiên cứu về thuật toán ANN 12](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc160)

[Bảng 4 : Các thuộc tính được trích xuất 37](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc27744)

[Bảng 5 : Kết quả train từ bộ dữ liệu CIC-IDS-2018 43](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc20930)

[Bảng 6 : So sánh giữa các nhóm nghiên cứu 44](file:///C:\Users\84845\Downloads\NCKH-IDS%20(3).docx#_Toc21851)

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐH SƯ PHẠM KỸ THUẬT TPHCM**

**THÔNG TIN KẾT QUẢ NGHIÊN CỨU CỦA ĐỀ TÀI**

**1. Thông tin chung:**

**-** Tên đề tài: Xây dựng hệ thống phát hiện xâm nhập mạng IDS sử dụng Machine Learning

- Chủ nhiệm đề tài: Nguyễn Thắng Lợi Mã số SV: 22162023

- Lớp: 22162A Khoa: Công Nghệ Thông Tin

- Thành viên đề tài:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Lớp** | **Khoa** |
| 1 | Nguyễn Văn Trường | 22162052 | 22162B | CNTT |
| 2 | Lê Anh Khoa | 22162016 | 22162A | CNTT |
| 3 | Nguyễn Thành Lộc | 22162022 | 22162B | CNTT |
| 4 | Nguyễn Trí Dũng | 22162009 | 22162A | CNTT |

- Người hướng dẫn: ThS. Nguyễn Thị Thanh Vân

**2. Mục tiêu đề tài:** Phân tích và thiết kế kiến trúc cho một hệ thống IDS tích hợp mô hình học máy, cách thức khai phá dữ liệu và huấn luyện cho mô hình học máy để tối ưu hóa khả năng nhận diện xâm nhập.

**3. Tính mới và sáng tạo:** Trên thế giới đã có nhiều nghiên cứu về việc áp dụng các công nghệ tiên tiến, đặc biệt là Machine Learning vào việc xây dựng các hệ thống phát hiện xâm nhập mạng. Tuy nhiên, việc nghiên cứu và ứng dụng Machine Learning vào hệ thống phát hiện xâm nhập mạng ở Việt Nam vẫn còn hạn chế chưa đáp ứng được nhu cầu bảo mật ngày càng cao. Do đó, nhóm đã tiến hành nghiên cứu và phát triển một hệ thống phát hiện xâm nhập mạng sử dụng Machine Learning trên bộ dữ liệu CIC-IDS 2018 để có thể góp phần nâng cao bảo mật cho các hệ thống bảo mật.

**4. Kết quả nghiên cứu:** Xây dựng hệ thống phát hiện xâm nhập mạng bao gồm các thành phần như: Bộ phận thu thập gói tin, bộ phận backend API-ML, bộ phận quản lý. Hoạt động kết hợp thu thập dữ liệu, xử lý đặc trưng, phân tích với Machine Learing và quản lý các phản hồi để đảm bảo an ninh mạng hiệu quả. Kết hợp các thuật toán Decision Tree, Random Forest, Extra Trees vào một mô hình Voting Classifier để kết hợp dự đoán từ nhiều mô hình con, tạo ra một kết quả cuối cùng bằng cách lấy ý kiến đa số giúp tận dụng được điểm mạnh của từng mô hình, giảm thiểu nhược điểm riêng lẻ của chúng. Kết quả đạt được sau khi sử dụng mô hình học máy với bộ dữ liệu CIC-IDS-2018 đạt được độ chính xác trung bình 97,95%.

**5. Đóng góp về mặt giáo dục và đào tạo, kinh tế - xã hội,** **an ninh, quốc phòng và khả năng áp dụng của đề tài:** Đề tài khi được áp dụng sẽ giúp tăng cường hiệu quả trong việc bảo vệ an ninh mạng trong việc phát hiện và xử lý kịp thời các cuộc tấn công, giúp cải thiện độ an toàn cho hệ thống thông tin tại các tổ chức, doanh nghiệp với tình hình tấn công mạng ngày càng gia tăng. Bên cạnh đó, hệ thống này cũng mang ý nghĩa thiết thực trong việc thúc đẩy ứng dụng công nghệ trí tuệ nhân tạo vào bảo mật, tạo nên môi trường bảo mật an toàn và bền vững. Và bài nghiên cứu này cũng có thể tài liệu cho sinh viên ngành an toàn thông tin nói riêng, công nghệ thông tin nói chung để góp phần nâng cao nhận thức, kiến thức về an toàn mạng.

**6.** **Công bố khoa học của SV từ kết quả nghiên cứu của đề tài** *(ghi rõ tên tạp chí nếu có)* hoặc nhận xét, đánh giá của cơ sở đã áp dụng các kết quả nghiên cứu *(nếu có)*:

Ngày tháng năm

|  |  |
| --- | --- |
|  | **SV chịu trách nhiệm chính**  **thực hiện đề tài**  *(kí, họ và tên)* |

**Nhận xét của người hướng dẫn về những đóng góp khoa học của SV thực hiện đề tài** *(phần này do người hướng dẫn ghi):*

Ngày tháng năm

|  |  |
| --- | --- |
|  | **Người hướng dẫn**  *(kí, họ và tên)* |

# **PHẦN MỘT: MỞ ĐẦU**

# 1. Lý do chọn đề tài

Hiện nay, các hành vi tấn công của tội phạm mạng ngày càng tinh vi về kỹ thuật và gia tăng cả về tần suất lẫn số lượng. Để bảo vệ hệ thống mạng trước các cuộc tấn công này, nhiều giải pháp đã và đang được triển khai, trong đó có Hệ thống phát hiện xâm nhập (IDS - Intrusion Detection System). IDS là phần mềm hoặc thiết bị được thiết kế để giám sát lưu lượng truy cập mạng, phát hiện và cảnh báo quản trị viên về những bất thường hoặc hành vi xâm nhập trái phép.

Gần đây, các mô hình Machine Learning đang được ứng dụng rộng rãi trong nhiều lĩnh vực công nghệ, bao gồm cả an ninh mạng. Một hệ thống IDS ứng dụng Machine Learning sẽ là giải pháp tuyệt vời để nâng cao tính bảo mật cho hệ thống mạng của các cơ quan, tổ chức.

Nhận thấy tầm quan trọng của việc ứng dụng IDS trong hệ thống mạng tại các cơ quan, tổ chức và việc áp dụng các mô hình học máy để xây dựng hệ thống IDS hiệu quả, chúng tôi đã chọn thực hiện đề tài “Tìm hiểu và xây dựng hệ thống phát hiện xâm nhập mạng IDS sử dụng Machine Learning”.

Việc nghiên cứu và phát triển hệ thống IDS sử dụng Machine Learning không chỉ giúp nâng cao khả năng phát hiện các mối đe dọa mạng mà còn cung cấp giải pháp bảo mật hiệu quả cho các tổ chức và doanh nghiệp trong việc bảo vệ hệ thống mạng khỏi các mối đe dọa ngày càng phức tạp.

# 2. Mục tiêu đề tài

Đề tài tập trung vào việc phân tích và thiết kế kiến trúc cho một hệ thống IDS tích hợp mô hình học máy, cách thức khai phá dữ liệu và huấn luyện cho mô hình học máy để tối ưu hóa khả năng nhận diện xâm nhập. Cụ thể gồm có các mục tiêu như sau:

* Trình bày các kiến thức tổng quan về xâm nhập mạng và các phương pháp phát hiện xâm nhập mạng phổ biến hiện nay
* Phân tích và đúc kết về tình hình nghiên cứu cả trong lẫn ngoài nước về các nội dung có liên quan đến đề tài
* Nghiên cứu về kết cấu và nguyên lý hoạt động của IDS, cách thức tích hợp học máy và các thuật toán học máy phù hợp
* Đề xuất kiến trúc và xây dựng hệ thống IDS sử dụng học máy
* Thử nghiệm và đánh giá về hiệu suất của kiến trúc đề xuất, từ đó chỉ ra các ưu nhược điểm và phương hướng phát triển phù hợp.

# 3. Phương pháp nghiên cứu

* Tìm hiểu các tài liệu về IDS và đặc điểm các cuộc xâm nhập mạng
* Tìm hiểu và phân tích các công trình liên quan
* Xử lý dữ liệu và huấn luyện mô hình học máy
* Thiết kế và xây dựng hệ thống IDS
* Tiến hành kiểm nghiệm hệ thống

# 4. Đối tượng và phạm vi nghiên cứu

Đề tài thực hiện đưa ra cái nhìn tổng quan về đặc điểm các cuộc xâm nhập và cách thức nhận diện thông qua hệ thống IDS. Sau đó tiến hành huấn luyện mô hình học máy và tích hợp để tạo ra hệ thống IDS. Đề tài được nghiên cứu trên phạm vi các mạng LAN vừa và nhỏ, có tích hợp trên cả các server để tăng cường khả năng bảo vệ các điểm trọng yếu trong hệ thống mạng.

# **PHẦN 2: NỘI DUNG**

# **CHƯƠNG 1: TỔNG QUAN VỀ TẤN CÔNG MẠNG VÀ CÁC PHƯƠNG PHÁP PHÁT HIỆN PHỔ BIẾN**

## 1.1. Giới thiệu về tấn công mạng

Tấn công mạng là các hành động trái phép đối với các tài sản số bên trong mạng của một tổ chức; nhằm vào hệ thống thông tin máy tính, cơ sở hạ tầng, mạng máy tính hoặc thiết bị máy tính cá nhân. Các bên tấn công thường thực hiện các cuộc xâm nhập mạng này nhằm thay đổi, phá hủy hoặc đánh cắp dữ liệu cá nhân.

## 1.2. Các dạng tấn công phổ biến

Trong lĩnh vực an ninh mạng, tấn công mạng được chia thành 2 dạng chính:

*Tấn công thụ động (Passive Attack)*

Là dạng tấn công nhằm thu thập hoặc nghe lén dữ liệu mà không làm thay đổi hoặc phá hủy dữ liệu. Mục tiêu của những cuộc tấn công này là thu thập thông tin nhạy cảm hoặc theo dõi các hoạt động trong mạng mà không gây sự chú ý.[1] Hậu quả mặc dù không gây thiệt hại trực tiếp đến dữ liệu, tấn công thụ động có thể dẫn đến việc rò rỉ thông tin nhạy cảm, gây nguy hiểm cho bảo mật lâu dài của hệ thống.

*Tấn công chủ động (Active attack)*

Là các hành động có mục đích làm thay đổi, sửa đổi hoặc phá hủy dữ liệu trong hệ thống mạng. Những cuộc tấn công này trực tiếp gây ra gián đoạn hoặc thiệt hại cho hệ thống và dữ liệu.[1] Tấn công chủ động thường dễ phát hiện hơn so với tấn công thụ động vì chúng tạo ra những thay đổi có thể quan sát được trên hệ thống. Hậu quả là chúng gây ra thiệt hại về dữ liệu, làm gián đoạn dịch vụ, tổn thất tài chính và làm ảnh hưởng đến uy tín của tổ chức

Hai dạng tấn công trên không tồn tại độc lập mà thường được triển khai thông qua các kỹ thuật tấn công cụ thể. Những kỹ thuật này có thể được thiết kế để phá hoại hệ thống (tấn công chủ động) hoặc theo dõi thông tin (tấn công thụ động). Dưới đây là một số kỹ thuật tấn công mạng phổ biến.

#### 1.2.1 Nhóm kỹ thuật tấn công nghe lén và thu thập thông tin (Tấn công thụ động)

*Sniffing (Nghe lén dữ liệu)*

Sniffing là hành vi chặn và phân tích dữ liệu được truyền qua mạng. Kẻ tấn công sử dụng các công cụ đặc biệt để theo dõi các gói tin, nhằm thu thập thông tin nhạy cảm như mật khẩu, email, hoặc thông tin cá nhân.

*Man-in-the-Middle (MitM)*

Man-in-the-Middle (MitM) là một dạng tấn công trong đó kẻ tấn công đặt mình vào vị trí trung gian giữa hai bên giao tiếp (thường là hai thiết bị hoặc người dùng) để nghe lén và thu thập dữ liệu mà không bị phát hiện.

*Traffic Analysis (Phân tích lưu lượng mạng)*

Traffic Analysis là quá trình phân tích lưu lượng mạng để suy luận thông tin nhạy cảm từ các mẫu lưu lượng, chẳng hạn như tần suất liên lạc, kích thước gói tin, hoặc địa chỉ IP nguồn và đích. Mặc dù không truy cập trực tiếp vào nội dung dữ liệu, nhưng nó có thể tiết lộ thông tin quan trọng về hệ thống hoặc người dùng.

#### 1.2.2 Nhóm kỹ thuật tấn công vào ứng dụng web (Tấn công chủ động)

*Tấn công SQL injection*

Là hình thức tấn công nhằm chèn các câu lệnh SQL độc hại vào các trường nhập liệu để truy cập, chỉnh sửa hoặc xóa dữ liệu trong cơ sở dữ liệu. Nó xảy ra khi một người dùng thực thi một truy vấn SQL đến cơ sở dữ liệu thông qua dữ liệu nhập từ máy client đến máy chủ.

*Tấn công Cross-site scripting (XSS)*

Tấn công XSS sử dụng tài nguyên web của bên thứ ba để chạy các tập lệnh trong trình duyệt browser web hoặc ứng dụng liên quan tập lệnh. Cụ thể, kẻ tấn công tiêm nhiễm một payload chứa JavaScript độc hại vào cơ sở dữ liệu của trang web. Khi người dùng yêu cầu một trang từ trang web, trang web sẽ truyền trang, với payload của tin tặc là một phần của nội dung HTML, đến trình duyệt của người dùng, nơi thực thi tập lệnh độc hại.

#### 1.2.3 Nhóm kỹ thuật tấn công vào hệ thống và dịch vụ (Tấn công chủ động)

*Denial-of-service (DoS) và distributed denial-of-service (DDoS)*

Một cuộc tấn công DDos sẽ chiếm đoạt tài nguyên (resource) của hệ thống khiến nó không thể phản hồi các yêu cầu dịch vụ. DDoS cũng là một cuộc tấn công vào tài nguyên của hệ thống, nhưng nó được thực hiện từ một số lượng lớn các host khác mà bị nhiễm phần mềm độc hại do tin tặc kiểm soát.

*Buffer Overflow*

Buffer Overflow xảy ra khi một ứng dụng hoặc chương trình bị gửi dữ liệu vượt quá khả năng lưu trữ của bộ nhớ đệm (buffer), dẫn đến ghi đè dữ liệu lên các vùng nhớ khác. Kẻ tấn công có thể lợi dụng lỗ hổng này để thực thi mã độc trên hệ thống.

*Privilege Escalation*

Privilege Escalation là kỹ thuật tấn công trong đó kẻ tấn công khai thác lỗ hổng trong hệ thống để nâng quyền từ người dùng thường lên quyền quản trị hoặc thậm chí quyền root (siêu người dùng).

#### 1.2.4 Nhóm kỹ thuật tấn công vào mạng (Tấn công chủ động)

*ARP Spoofing*

ARP Spoofing xảy ra khi kẻ tấn công gửi các gói tin ARP giả mạo trên mạng LAN để đánh lừa thiết bị mạng, thay đổi bảng ánh xạ giữa địa chỉ IP và địa chỉ MAC. Điều này cho phép kẻ tấn công chặn, thay đổi hoặc thậm chí làm gián đoạn lưu lượng mạng.

*DNS Spoofing*

DNS Spoofing xảy ra khi kẻ tấn công thay đổi bản ghi DNS hoặc trả lời truy vấn DNS bằng thông tin giả mạo, chuyển hướng người dùng đến các trang web giả mạo hoặc máy chủ không mong muốn.

*Replay Attack*

Đây là cuộc tấn công khi kẻ tấn công ghi lại các gói tin hoặc thông tin truyền tải giữa hai bên trong mạng và phát lại chúng sau đó để thực hiện các hành động trái phép. Replay Attack thường được thực hiện trên các giao thức mạng không được mã hóa hoặc có lỗ hổng bảo mật.

#### 1.2.5 Nhóm kỹ thuật tấn công vào lỗi người dùng (Tấn công chủ động)

*Spear-Phishing*

Spear-Phishing là hình thức tấn công giả mạo được cá nhân hóa, trong đó kẻ tấn công gửi email hoặc tin nhắn giả mạo từ các nguồn đáng tin cậy nhằm lừa người dùng thực hiện các hành động như tải phần mềm độc hại, cung cấp thông tin đăng nhập, hoặc chuyển tiền.

*Drive-by Download (Tấn công tải mã độc ngầm)*

Drive-by Download xảy ra khi người dùng vô tình tải xuống mã độc từ một trang web bị nhiễm mà không nhận thức được điều này. Tấn công này thường lợi dụng các lỗ hổng trình duyệt hoặc hệ điều hành.

*Social Engineering (Kỹ thuật xã hội)*

Social Engineering là một phương pháp tấn công sử dụng các thủ đoạn tâm lý để lừa người dùng cung cấp thông tin nhạy cảm hoặc thực hiện các hành động không mong muốn, như cung cấp mật khẩu, tài khoản, hoặc thông tin cá nhân.

## 1.3. Cách nhận diện một cuộc tấn công mạng

### 1.3.1. Dựa vào các gói tin (packets)

Nguồn gốc và đặc điểm của một gói tin có thể cung cấp thông tin hữu ích để dự đoán mục đích hoặc nguy cơ tiềm ẩn mà gói tin đó mang lại. Một gói tin riêng lẻ có thể không biểu hiện là phần mềm độc hại, nhưng khi được ghép nối với các gói tin khác, nó có thể trở thành một phần của một chương trình độc hại hoàn chỉnh. Ngoài ra, một gói tin cũng có thể chứa dữ liệu nhằm đột nhập hệ thống thông qua việc giả mạo ID, thường không bị phát hiện ngay mà có thể kéo dài hàng tuần hoặc thậm chí vài tháng. Các tác nhân độc hại thường sử dụng nhiều kỹ thuật tinh vi để tránh sự chú ý của các hệ thống bảo mật.

Bằng cách phân tích lưu lượng gói tin, chúng ta có thể theo dõi hành vi của chúng. Điều này giúp nhận diện những gói tin có hành vi đáng ngờ, chẳng hạn như những gói tin cố gắng "quét" các khu vực mạng cụ thể trong một khoảng thời gian định kỳ (vài phút mỗi ngày hoặc vài giờ mỗi tuần). Những hoạt động này có thể là dấu hiệu sớm của một cuộc tấn công mạng đang được chuẩn bị. Ngoài ra, các gói tin giao tiếp với máy chủ proxy ẩn danh hoặc với các máy chủ đặt tại những khu vực thường liên quan đến hành vi giả mạo cũng nên được giám sát chặt chẽ.

Phân tích hành vi gói tin, một lĩnh vực quan trọng trong an ninh mạng, đang được đẩy mạnh bởi sự gia tăng của các thiết bị được kết nối qua Internet of Things (IoT). Không chỉ máy tính, các bộ định tuyến, thiết bị gia đình thông minh, và các thiết bị IoT khác cũng có thể trở thành mục tiêu của các cuộc tấn công. Tin tặc có thể khai thác lỗ hổng trong các thiết bị này trước cả khi nhà sản xuất nhận ra vấn đề.

Trong một số trường hợp, người dùng vô tình trở thành đồng phạm trong các cuộc tấn công mạng mà không hề hay biết. Ví dụ, một người dùng có thể đang trò chuyện qua FaceTime hoặc Skype mà không nhận ra rằng thiết bị của họ đã bị biến thành một bot tấn công. Những dấu hiệu như hình ảnh hoặc video bị giật, mờ, âm thanh bị ngắt quãng có thể là biểu hiện của một cuộc tấn công đang diễn ra.

### 1.3.2. Phát hiện dựa trên dòng dữ liệu bất thường

Trong hệ thống , những bất thường trên dòng dữ liệu có thể xảy ra do vi phạm các giao thức và tải trọng ứng dụng, có thể kể đến như:

*Dựa trên giao thức bất thường:*

Ở đây đề cập đến sự bất thường trong định dạng giao thức và hành vi giao thức liên quan đến các tiêu chuẩn và thông số kỹ thuật của Internet. Có nhiều khía cạnh trong giao thức TCP và IP cần được theo dõi, ví dụ, các cờ khác nhau, SYN, ACK, FIN, sự kết hợp của chúng và các cờ dự trữ. Cách phân mảnh IP khi lắp ráp lại được thực hiện theo tiêu chuẩn

*Dựa trên bất thường thống kê:*

Từ chối dịch vụ (DoS) và từ chối dịch vụ phân tán (DDoS) dẫn đến một loạt lưu lượng truy cập trên mạng diễn ra bất bình. Để khắc phục loại tấn công này, các cấu hình cơ bản được tạo ra trên lưu lượng di chuyển bình thường như được mô tả trước đó là dựa trên mô hình thống kê như Naïve Bayes để xác định các gói bất thường trên mạng. Trong khi học hành vi của lưu lượng mạng, chức năng của mô hình thống kê là tính toán điểm xác suất cho từng gói dữ liệu được coi là lưu lượng bình thường. Điểm số được tính toán dựa trên dữ liệu được lấy mẫu trong một khoảng thời gian và được lưu trữ trong một hồ sơ cơ sở. Dựa vào điểm số này ta có thể phát hiện được bất thường trong hệ thống để kịp thời cảnh báo và ngăn chặn[2].

## 1.4. Các phương pháp phát hiện tấn công mạng

Để nhận biết một cuộc tấn công mạng, hệ thống cần dựa vào các dấu hiệu và mô hình hoạt động bất thường trong mạng. Những đặc điểm nhận biết này có thể được xác định thông qua các phương pháp phát hiện khác nhau. Dưới đây là chi tiết về các phương pháp này

### 1.4.1. Phát hiện dựa trên sự bất thường (Anomaly-based Detection)

Phương pháp phát hiện dựa trên bất thường hoạt động bằng cách huấn luyện hệ thống với một đường chuẩn đại diện cho hành vi hoạt động bình thường của hệ thống. Sau đó, khi xảy ra một sự kiện bất thường, hệ thống sẽ so sánh sự kiện đó với đường chuẩn và nhận diện nó là bất thường [2].

Khi một sự kiện được xác định là bất thường, hệ thống sẽ kích hoạt cảnh báo, thông báo cho người dùng mạng và hệ thống máy tính rằng có dấu hiệu bất thường. Các kỹ thuật phát hiện dựa trên sự bất thường:

* Không giám sát (Unsupervised): Hệ thống phát hiện bất thường không giám sát nhận diện các sự kiện bất thường trong tập dữ liệu chưa gán nhãn. Nó thực hiện việc này dựa trên các đặc điểm nội tại của dữ liệu và hoạt động theo giả định rằng phần lớn các hoạt động trong tập dữ liệu là bình thường.
* Bán giám sát (Semi-supervised): Hệ thống phát hiện bất thường bán giám sát sử dụng tập dữ liệu bình thường đã được gán nhãn để tạo ra đường chuẩn. Sau đó, hệ thống sử dụng đường chuẩn này để so sánh các sự kiện và quyết định xem chúng có bình thường hay không.

Giám sát (Supervised): Hệ thống phát hiện bất thường có giám sát sử dụng các tập dữ liệu đã được gán nhãn là “bình thường” và “bất thường”.

### 1.4.2. Phát hiện dựa trên dấu hiệu tấn công (Signature-based Detection)

Phương pháp phát hiện dựa trên dấu hiệu tấn công là một cơ chế an ninh mạng nhằm nhận diện các mối đe dọa bằng cách so sánh dữ liệu đầu vào với cơ sở dữ liệu chứa các dấu hiệu của các mối đe dọa đã biết. Những dấu hiệu này là các chuỗi dữ liệu hoặc đặc trưng liên quan đến các hoạt động độc hại đã biết, chẳng hạn như virus, phần mềm độc hại, và các cuộc xâm nhập mạng [3]. Có các loại chữ ký như sau:

- Dấu hiệu tĩnh (Static Signatures): Đây là các chuỗi dữ liệu cố định hoặc trùng khớp với các chuỗi chính xác trong các tệp tin độc hại. Dấu hiệu tĩnh hiệu quả đối với các mối đe dọa đã biết nhưng có thể bị vượt qua nếu phần mềm độc hại có các biến thể nhỏ.

- Dấu hiệu động (Dynamic Signatures): Các dấu hiệu này dựa trên hành vi của phần mềm độc hại thay vì các chuỗi dữ liệu tĩnh. Chúng theo dõi các hành động được thực hiện bởi chương trình để phát hiện hoạt động đáng ngờ.

Về cách thức hoạt động: Các hệ thống bảo mật duy trì một cơ sở dữ liệu lớn về các dấu hiệu tấn công của các mối đe dọa đã biết. Các dấu hiệu này liên tục được cập nhật bởi các chuyên gia và tổ chức an ninh mạng. Khi dữ liệu vào hệ thống, phần mềm bảo mật sẽ quét dữ liệu để tìm kiếm các kết quả khớp với cơ sở dữ liệu dấu hiệu. Nếu phát hiện sự khớp giữa dữ liệu đầu vào và một dấu hiệu đã biết, hệ thống sẽ gắn cờ dữ liệu đó như một mối đe dọa tiềm ẩn.

### 1.4.3. Phát hiện dựa trên hành vi (Behavioral Detection):

Phương pháp phát hiện dựa trên kỹ thuật phân tích các hành động và tương tác của một chương trình hoặc tiến trình thay vì chỉ dựa vào mã hoặc chữ ký của nó. Phương pháp này có thể nhận diện phần mềm độc hại bằng cách tìm kiếm các mẫu hành vi đáng ngờ hoặc bất thường như :

* Chỉnh sửa tệp hệ thống
* Truy cập tài nguyên mạng
* Tạo tiến trình mới
* Chèn mã

Phát hiện dựa trên hành vi cũng có thể theo dõi hiệu suất hệ thống như mức sử dụng CPU, dung lượng bộ nhớ, hoạt động ổ đĩa, hoặc lưu lượng mạng để phát hiện các dấu hiệu bất thường do phần mềm độc hại gây ra. Có khả năng nhận diện các mối đe dọa mới và chưa biết, ngay cả khi chúng thay đổi hoặc né tránh chữ ký. Phát hiện dựa trên hành vi sử dụng nhiều kỹ thuật và công cụ khác nhau để thu thập và phân tích dữ liệu hành vi từ hệ thống.

* Kỹ thuật suy đoán (Heuristics): So sánh hành vi của một chương trình hoặc tiến trình với một đường chuẩn hành vi bình thường hoặc các hành vi độc hại đã biết, và gắn cờ bất kỳ sự khác biệt hoặc tương đồng nào làm dấu hiệu tiềm ẩn của phần mềm độc hại.
* Sandboxing: Cô lập chương trình hoặc tiến trình trong một môi trường an toàn và kiểm soát để phân tích hành vi.
* Học máy (Machine Learning): Ứng dụng một nhánh của trí tuệ nhân tạo có khả năng tự động và liên tục cập nhật, cải thiện các mô hình và quy tắc dựa trên phản hồi và kết quả phân tích. Điều này giúp phát hiện dựa trên hành vi thích ứng với sự thay đổi và tiến hóa liên tục của phần mềm độc hại[2].

# **CHƯƠNG 2: CÁC CÔNG TRÌNH NGHIÊN CỨU LIÊN QUAN**

Xâm nhập mạng là vấn đề đang được xem trọng ở khắp nơi trên thế giới, bất kỳ nơi nào cũng có thể bị tấn công mạng do đó đây cũng là chủ đề phổ biến thu hút nhiều sự đầu tư của nhiều tổ chức, công ty lớn quan tâm. Trong suốt những năm gần đây nhiều bài báo khoa học, nghiên cứu đã được phát triển với chủ đề phát triển xâm nhập mạng kết hợp với machine learning. Có thể kể đến một số nghiên cứu về hệ thống phát hiện xâm nhập ứng dụng các loại học máy như:

1. *Decision Tree*

Decision Tree là một mô hình học máy phổ biến trong lĩnh vực machine learning bằng việc xây dựng một mô hình dạng cây, trong đó mỗi nút đại diện cho một điều kiện dựa trên đặc trưng dữ liệu và các nhánh biểu diễn cho các kết quả có thể có. Đây là một mô hình phổ biến được ứng dụng trong hệ thống phát hiện xâm nhập, có nhiều nghiên cứu về mô hình này như:

Bảng 1: Danh sách nghiên cứu về thuật toán Decision Tree

|  |  |  |  |
| --- | --- | --- | --- |
| Tác giả | Mô hình sử dụng | Bộ dữ liệu | Kết quả đạt được |
| Shailendra Sahu, B. M. Mehtre [29] | Decision Tree (J48) | Kyoto 2006+ dataset | Đạt độ chính xác cao nhất là 97,23% |
| Kajal Rai, M. Syamala Devi, Ajay Guleria [30] | Decision Tree Split (DTS) algorithm | NSL-KDD | Đạt độ chính xác cao nhất là 79,52% |
| Siva S. Sivatha Sindhu, S. Geetha, A. Kannan [31] | Decision Stump, C4.5, Naive Baye’s Tree,  Random Forest, Random Tree and Representative Tree | KDD’99 CUP | Mô hình REP Tree đạt độ chính xác cao nhất với 89,11% |
| Alikhanov Jumabe, SeungSam Yan [32] | CatBoost | CIC-IDS-2018 | Đạt độ chính xác cao nhất là 92,41% |
| Bùi Điền Phong [2] | Decision Tree | NSL-KDD | Đạt kết quả cao nhất là 98,05% |

1. *Support vector machine*

Là một mô hình học máy được thiết kế để tìm ra một siêu phẳng tối ưu để phân chia các điểm dữ liệu thành các nhóm tương ứng với các lớp khác nhau, sao cho khoảng cách giữa siêu phẳng và các điểm dữ liệu gần nhất có thể. Các công trình nghiên cứu liên quan như:

Bảng 2: Danh sách nghiên cứu về thuật toán SVM

|  |  |  |  |
| --- | --- | --- | --- |
| Tác giả | Mô hình sử dụng | Bộ dữ liệu | Kết quả đạt được |
| Majjed Al-Qatf, Yu Lasheng, Mohammed Al-Habib, Kamal Al-Sabahi [18] | Kết hợp SAE và SVM | NSL-KDD | Độ chính xác cao nhất là 99,31% |
| Amin Dastanpour, Raja Azlina Raja Mahmood [33] | Genetic algorithm kết hợp SVM | KDD’99 CUP | Độ chính xác cao nhất là 99,5% |
| Jayshree Jha, Leena Ragha [34] | SVM | NSL-KDD | Độ chính xác cao nhất là 99,37% |

1. *Artificial Neural Network*

*Mạng nơ-ron nhân tạo* (Artificial Neural Network) là mô hình toán học được thiết kế để mô phỏng cách thức hoạt động của hệ thần kinh sinh học trong não người. ANN bao gồm nhiều nơ-ron nhân tạo kết nối với nhau , cho phép hệ thống học hỏi xử lý thông tin một cách hiệu quả. Các công trình liên quan như:

Bảng 3: Danh sách nghiên cứu về thuật toán ANN

|  |  |  |  |
| --- | --- | --- | --- |
| Tác giả | Mô hình sử dụng | Bộ dữ liệu | Kết quả đạt được |
| Bhupendra Ingre, Anamika Yadav [35] | ANN | NSL-KDD | Kết quả cao nhất đạt 81,2% |
| Marwan Ali Albahar, Muhammad Binsawad [36] | ANN | Open Stack dataset | Kết quả cao nhất đạt 97,87% |
| L.P. Dias, J. J. F. Cerqueira [37] | ANN | KDD’99 CUP | Kết quả cao nhất đạt 99,8% |

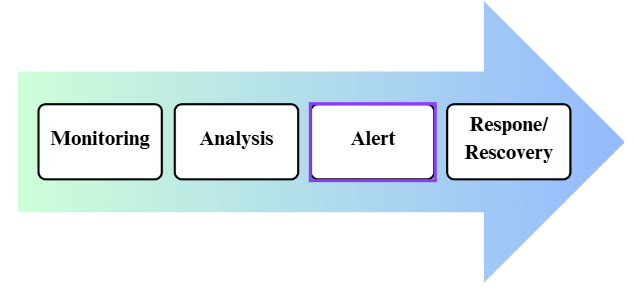
# 

# **CHƯƠNG 3: HỆ THỐNG PHÁT HIỆN XÂM NHẬP MẠNG**

## 3.1. Giới thiệu về hệ thống phát hiện xâm nhập mạng

Hệ thống phát hiện xâm nhập (Intrusion Detection System - IDS) là một hệ thống có nhiệm vụ giám sát các hoạt động trong mạng hoặc trên một thiết bị cụ thể để phát hiện các hành vi bất thường hoặc có khả năng đe dọa đến an ninh. Mục tiêu chính của IDS là xác định các mối đe dọa tiềm ẩn, giúp ngăn ngừa các cuộc tấn công hoặc xâm nhập trái phép vào hệ thống mạng. IDS hoạt động bằng cách phân tích lưu lượng mạng hoặc hành vi của hệ thống, so sánh chúng với các mẫu và quy tắc đã được định nghĩa trước, từ đó phát hiện ra những hành vi bất thường có thể là dấu hiệu của một cuộc tấn công.

Hệ thống phát hiện xâm nhập (IDS) kiểm tra lưu lượng truy cập mạng để phát hiện bất kỳ hoạt động đáng ngờ hoặc bất thường nào và đưa ra cảnh báo đến các người quản trị mạng. Một IDS cơ bản có nguyên lý hoạt động gồm 4 thành phần chính bao gồm: Giám sát, phân tích, cảnh báo, phản hồi hoặc phục hồi [20]:



Hình 1: Nguyễn lý cơ bản của IDS [20]

* Thành phần giám sát (Monitoring): Thành phần này có nhiệm vụ thu thập dữ liệu từ các nguồn khác nhau như lưu lượng mạng, nhật ký hệ thống,… bằng cách thu thập toàn bộ các gói tin thông qua các công cụ phổ biến như Wireshark, Snort[11], các giải pháp giám sát log như ELK Stack,… Các gói tin sau khi thu thập đều được sao lưu, xử lý và phân tích đến từng trường thông tin nhằm xác định gói tin thuộc kiểu gói tin nào, dịch vụ nào, kích cỡ bao nhiêu,.. Việc giám sát có thể được thực hiện theo thời gian thực.
* Thành phần phân tích (Analysis): Đây là thành phần quan trọng nhất trong IDS. Thành phần này dựa trên dữ liệu được thu thập để xác định xem có dấu hiệu đáng ngờ hay hay vi bất thường của các cuộc tấn công hay không. Để có thể phát hiện được dấu hiệu đáng ngờ, hệ thống có thể dựa trên:

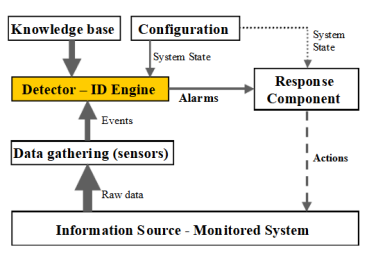
+ Dựa trên chữ ký (Signature-based Detection): IDS sẽ so sánh dữ liệu thu thập được với các mẫu tấn công đã được thu thập từ các cuộc tấn công đã xảy ra. Phương pháp này giúp phát hiện nhanh và hiệu quả các mối đe dọa đã biết còn với các cuộc tấn công chưa ghi nhận thì phương pháp này không có hiệu quả

+ Dựa trên dấu hiệu bất thường (Anomaly-based Detection): IDS sẽ so sánh hoạt động với một dữ liệu cơ sở được xác định là “hoạt động bình thường”. Nếu phát hiện có sai lệch với dữ liệu cơ sở thì hệ thống sẽ đưa ra cảnh báo. Phương pháp này hiệu quả trong việc phát hiện các cuộc tấn công chưa xảy ra nhưng cũng có thể đưa ra các cảnh báo giả (false negative).

* Thành phần cảnh báo (Alert): Khi có bất kỳ hoạt động hay hành vi đáng ngờ xảy ra thì hệ thống sẽ thông báo cho các người quản trị mạng về các hành vi có khả năng đe doạ. Việc cảnh báo có thể thông báo cho quản trị viên qua email hay bảng điều khiển (Dashboard) để người quản trị có thể nhanh chóng ứng phó với các tình huống xấu.
* Thành phần phản hồi/phục hồi (Response/Recovery): Sau khi gửi cảnh báo tới người quản trị thì hệ thống có thể xử lý các mối đe doạ được phát hiện như tự động chặn các IP đáng ngờ, cập nhật các quy tắc tường lửa,… Sau khi các mối đe doạ đã được giải quyết thì hệ thống cần được khôi phục, các lỗ hổng bảo mật cần được vá và đánh giá lại các chính sách bảo mật. Cuối cùng hệ thống sẽ ghi nhận chi tiết về cuộc tấn công, phân tích kỹ và cải thiện hệ thống.

## 3.2. Cấu trúc của IDS

Cấu trúc chính của IDS bao gồm các thành phần: Nguồn thông tin, thu thập dữ liệu, công cụ phát hiện xâm nhập, cơ sở tri thức, cấu hình, thành phần phản hồi, luồng dữ liệu và cảnh báo [19].



Hình 2: Cấu trúc của IDS [19]

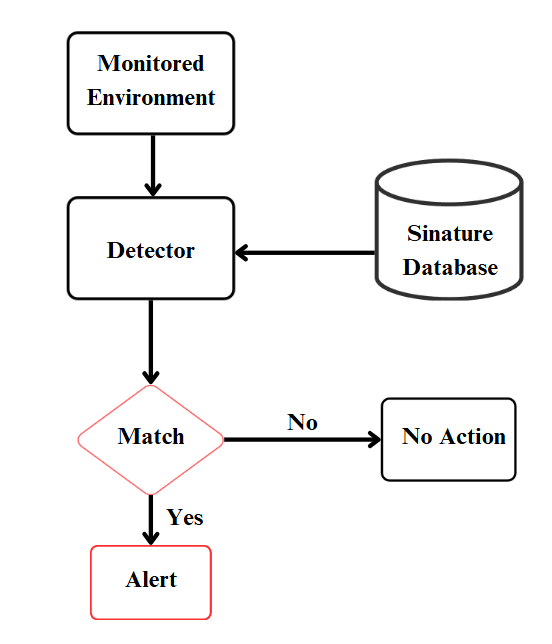
* Nguồn thông tin, hệ thống giám sát (Information Source - Monitored System): Đây là hệ thống giám sát lưu lượng mạng, các dữ liệu từ nguồn này sẽ được thu thập và phân tích.
* Thu thập dữ liệu (Data Gathering - Sensors): Có nhiệm vụ thu thập dữ liệu từ hệ thống giám sát. Dữ liệu có thể là lưu lượng mạng, nhật ký hệ thống,…
* Bộ phát hiện - Công cụ phát hiện xâm nhập (Detector - ID Engine): Có nhiệm vụ xử lý dữ liệu từ Sensors để phát hiện các hoạt động bất thường. Việc phát hiện có thể dựa trên chữ ký (Signature) hay các hoạt động bất thường.
* Cơ sở tri thức (Knowledge Base): Có tác dụng lưu trữ dữ liệu đã được thu thập từ sensors nhưng được lưu dưới dạng đã được xử lý. Các dữ liệu này được các chuyên gia mạng và bảo mật biên soạn nhằm duy trì hệ thống có thể phát hiện được các cuộc tấn công mới nhất
* Cấu hình (Configuration): Có chức năng cung cấp thông tin hiện tại về IDS bao gồm các thiết lập, ngưỡng và tham số hoạt động. Để cho phép tuỳ chỉnh nhằm tối ưu hoá IDS phù hợp với các môi trường mạng hay các yêu cầu bảo mật khác nhau.
* Thành phần phản hồi (Response Component): Có nhiệm vụ thực hiện các cảnh báo khi phát hiện các hành vi bất thường. Các phản hồi có thể được thực hiện tự động hay do con người tự xử lý. Nhằm đảm bảo các mối đe doạ sẽ được phát hiện kịp thời, giảm thiểu các thiệt hại có thể xảy ra.

## 3.3. Phân loại IDS

Hệ thống phát hiện xâm nhập có thể xác định các hoạt động, hành vi đáng ngờ gây ra thiệt hại cho hệ thống thông tin. Bất kỳ hoạt động nào có thể gây ra đe doạ đến tính bảo mật, toàn vẹn hoặc tính sẵn sàng của hệ thống thông tin sẽ được xem xét là một sự xâm nhập. Như các hoạt động làm cho máy tính không phản hồi với các dịch vụ hợp pháp sẽ được xem là một đe doạ. IDS là một phần mềm hay một hệ thống máy tính nhằm bảo vệ hệ thống thông tin. Mục tiêu của IDS là xác định được các lưu lượng truy cập mạng độc hại mà hệ thống tường lửa truyền thống không thể xác định được. IDS có thể phân thành hai nhóm chính là SIDS và AIDS [21].

*a) Hệ thống phát hiện xâm nhập dựa trên việc so khớp mẫu (Signature-based Intrusion Detection Systems)*

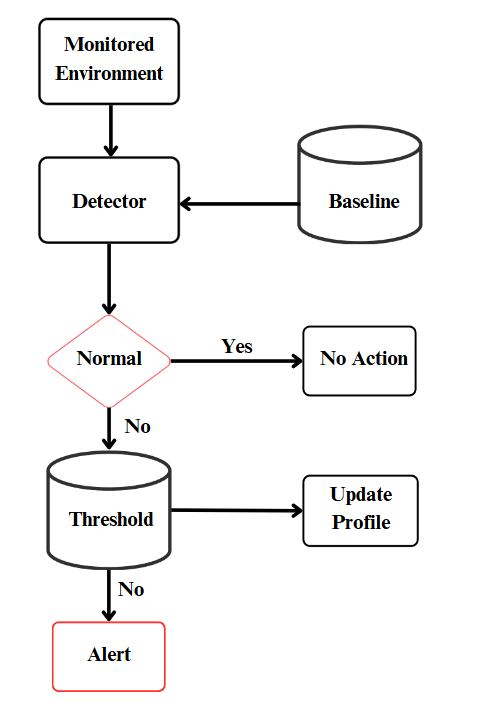
Hệ thống sẽ so sánh hoạt động hiện tại với cơ sở dữ liệu chứa các chữ ký của các cuộc tấn công trước đó đã được ghi nhận. Sau đó thực hiện so khớp nếu có dấu hiệu nào giống với chữ ký thì hệ thống sẽ ghi nhận là tấn công và đưa ra cảnh báo. Quy tắc của SIDS có dạng “nếu: điều kiện - thì: kết quả”. SIDS thường được triển khai trong các công cụ phổ biến như Snort, NetSTAT. SIDS mang lại khả năng nhận diện chính xác và tốc độ nhanh do so khớp với các cuộc tấn công đã xảy ra. Nhưng hạn chế của SIDS cũng nằm trong việc chỉ so khớp nên sẽ không hiệu quả đối với việc phát hiện ra các cuộc tấn công chưa từng xảy ra. Các phương pháp xây dựng chữ ký có thể kể đến như máy trạng thái (State machines),…



Hình 3: Nguyên lý hoạt động của SIDS [21]

*b) Hệ thống phát hiện xâm nhập dựa trên phát hiện bất thường Anomaly-based Intrusion Detection System (AIDS)*

Hệ thống sẽ mô hình hóa hành vi bình thường của hệ thống máy tính, được tạo ra bằng cách sử dụng các phương pháp như học máy, thống kê. Bất kỳ sai lệch đáng kể nào giữa hành vi quan sát được và mô hình học máy sẽ được ghi nhận là một hành vi bất thường có thể dẫn đến cuộc tấn công. AIDS có thể được phân loại dựa vào phương pháp dùng để xây dựng nó như dựa trên kiến thức, dựa trên thống kê, dựa trên học máy. Ưu điểm chính của AIDS là phát hiện được các cuộc tấn công chưa từng biết vì không phụ thuộc vào cơ sở dữ liệu chữ ký. AIDS cũng có thể nhận diện được các hoạt động độc hại từ bên trong hệ thống do các hành vi bên trong hệ thống sẽ gây bất thường nên hệ thống có thể đưa ra cảnh báo. Do đó những kẻ tấn công khó mà xác định được hành vi nào là bất thường để không gây ra kích hoạt cảnh báo. Nhưng một trong những nhược điểm lớn nhất của AIDS là có thể đưa nhiều cảnh báo giả vì những bất thường có thể là những hoạt động bình thường mới xuất hiện.



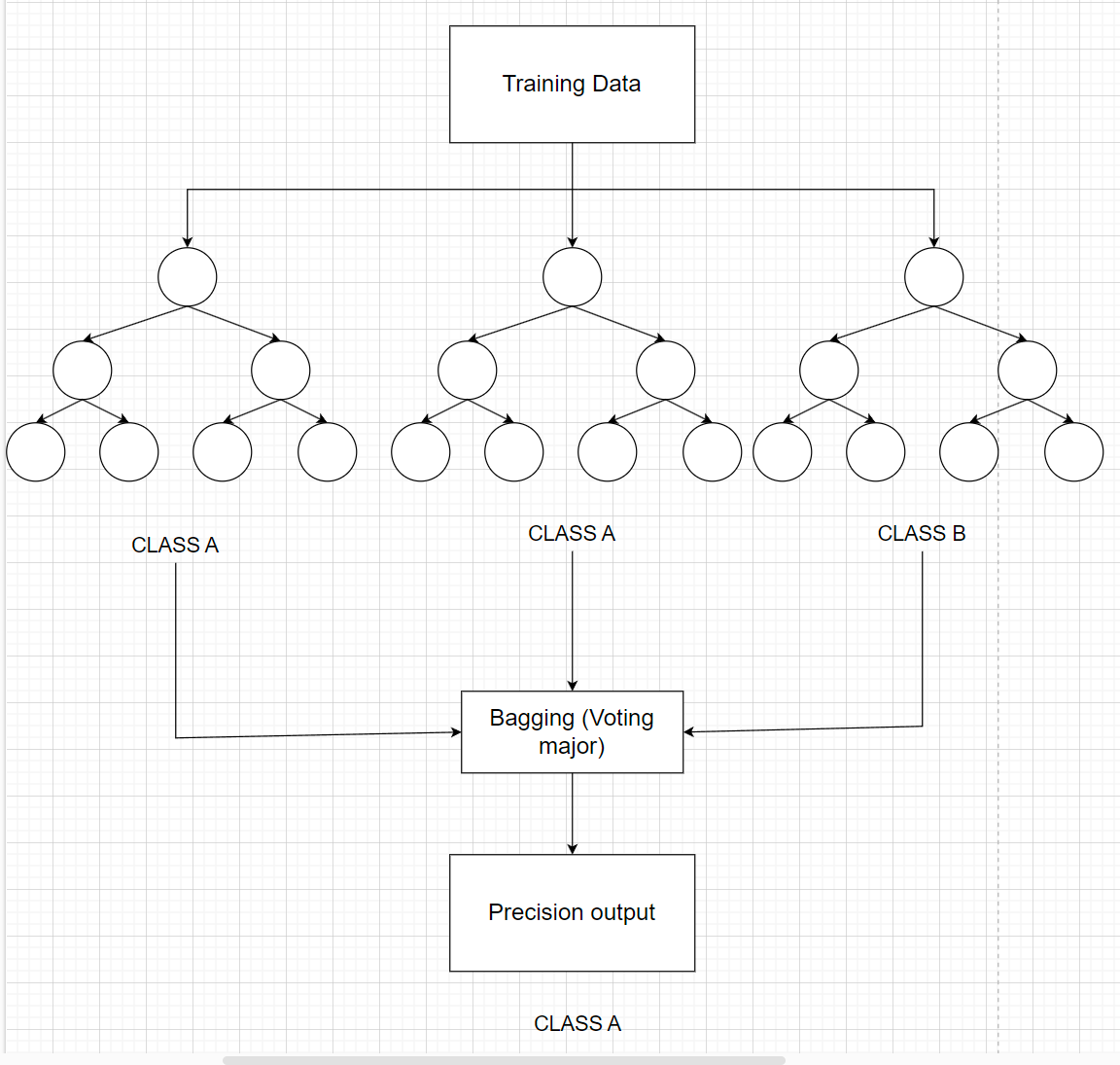
Hình 4: Nguyên lý hoạt động AIDS [21]

## 3.4. Các thuật toán học máy được áp dụng phổ biến trong hệ thống IDS

### 3.4.1 Random Forest

Thuật toán Random Forest là một kỹ thuật mạnh mẽ trong học máy. Nó hoạt động bằng cách tạo ra một số cây quyết định trong giai đoạn đào tạo. Mỗi cây được xây dựng bằng cách sử dụng một tập hợp con ngẫu nhiên của tập dữ liệu để đo một tập hợp con ngẫu nhiên các tính năng trong mỗi phân vùng. Tính ngẫu nhiên này tạo ra sự thay đổi giữa các cây riêng lẻ, giảm nguy cơ overfitting và cải thiện hiệu suất dự đoán tổng thể.

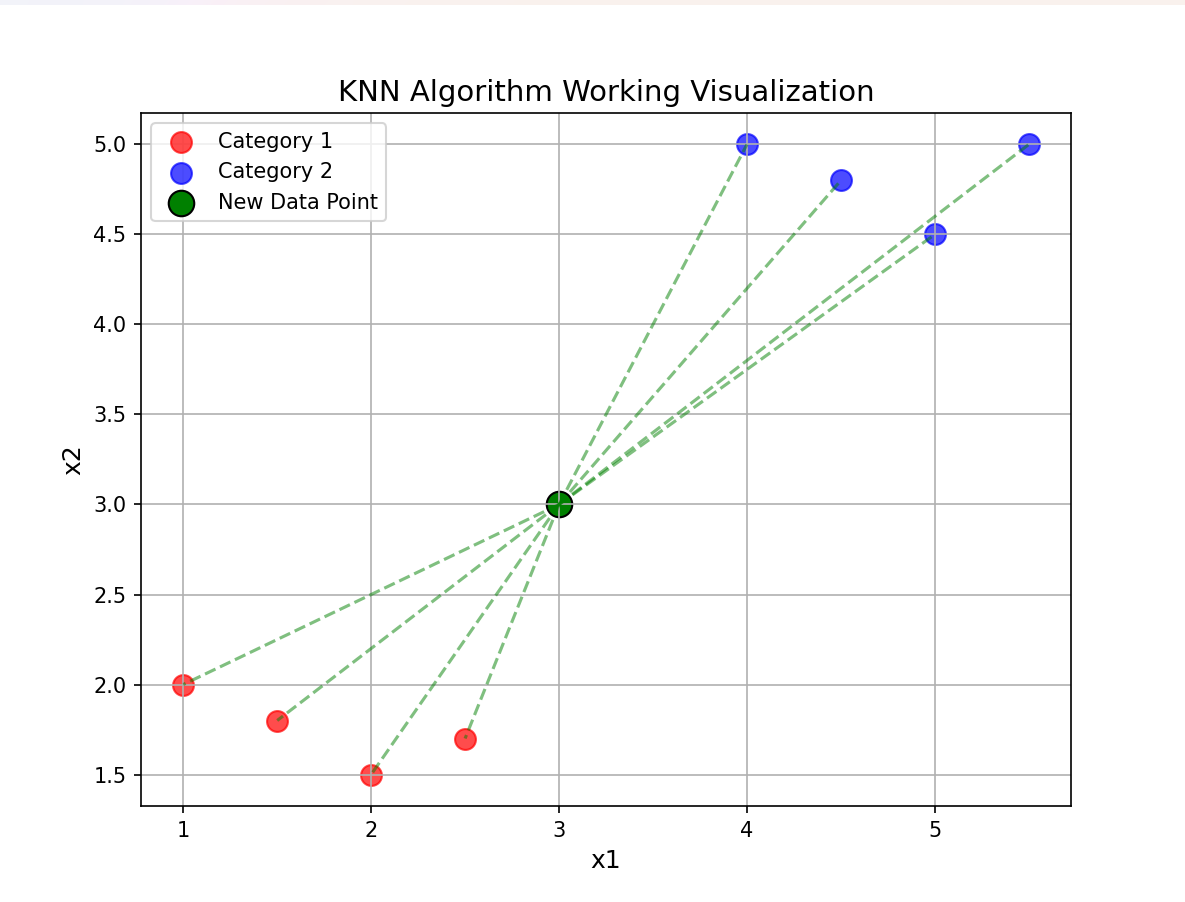
Trong dự đoán, thuật toán tổng hợp kết quả của tất cả các cây, bằng cách bỏ phiếu (cho các tác vụ phân loại) hoặc bằng cách tính trung bình (cho các tác vụ hồi quy). Quá trình ra quyết định hợp tác này, được hỗ trợ bởi nhiều cây với các hiểu biết sâu sắc của chúng, cung cấp một ví dụ về kết quả ổn định và chính xác. Random Forest được sử dụng rộng rãi cho các hàm phân loại và hồi quy, được biết đến với khả năng xử lý dữ liệu phức tạp, giảm tình trạng quá khớp và cung cấp dự báo đáng tin cậy trong các môi trường khác nhau [22].



Hình : Mô tả hoạt động Random Forest

### 3.4.2 K-nearest neighbors (KNN)

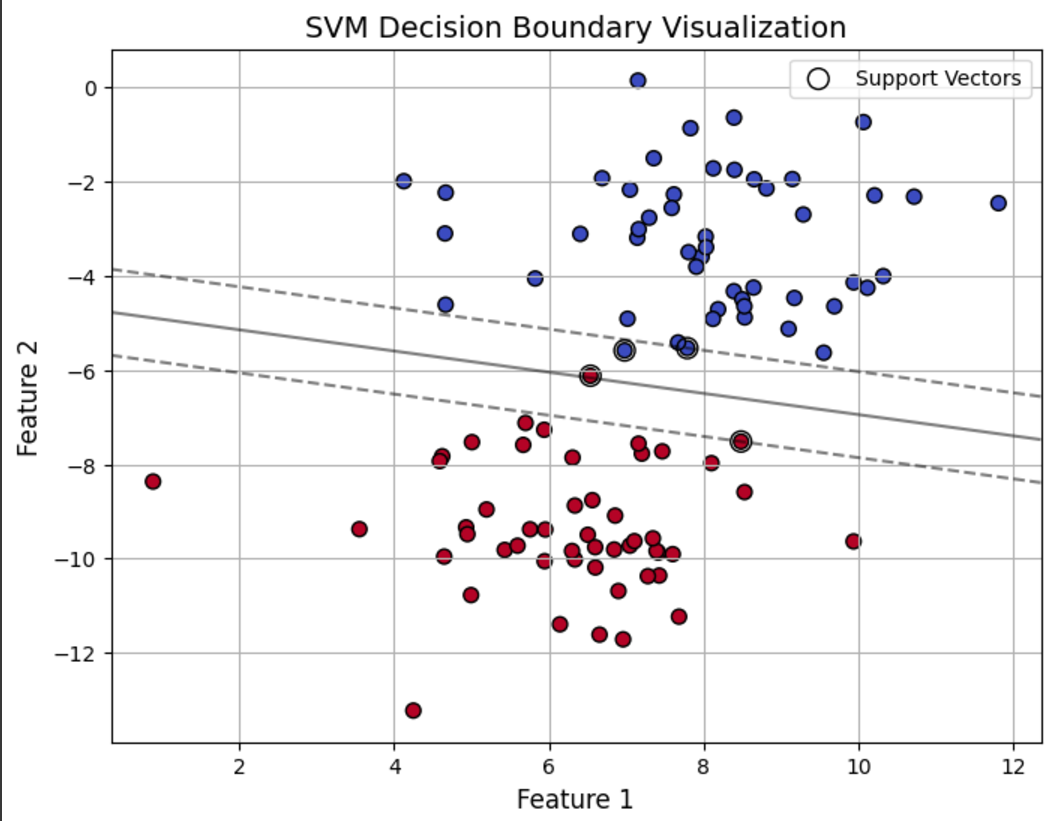
Việc phân cụm là một ý tưởng phân chia dữ liệu thành các cụm có ý nghĩa (hoặc nhóm), bằng cách đặt dữ liệu có tính chất giống nhau vào cùng một cụm. KNN là một trong những thuật toán ML không giám sát phổ biến dựa trên centroid (tâm của cụm dữ liệu). K đại diện cho số lượng centroid trong một tập dữ liệu. Khoảng cách từ mỗi điểm dữ liệu tới tâm được tính toán để chỉ định một số điểm dữ liệu nhất định sẽ vào cùng một cụm nào. Ý tưởng chính của thuật toán là làm giảm tổng khoảng cách giữa các điểm dữ liệu và centroid tương ứng của chúng trong một cụm [10].



Hình : Mô tả KNN bằng đồ thị.

### 3.4.3 Support vector machine (SVM)

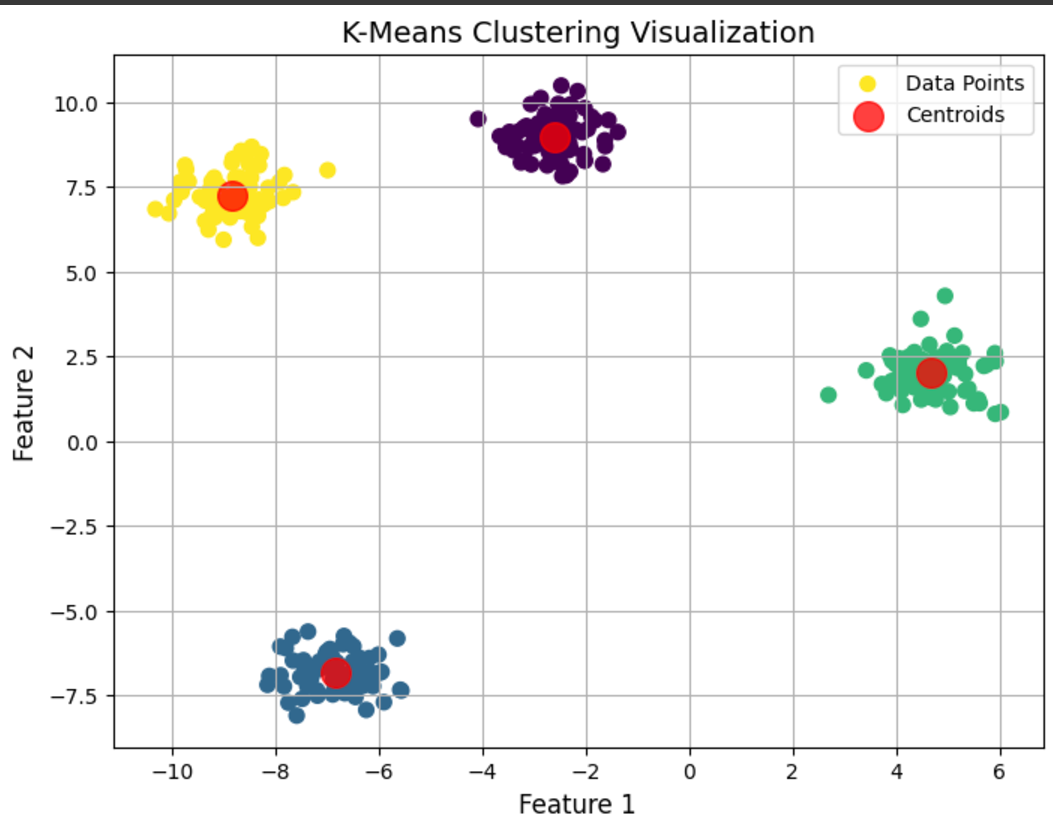
Support Vector Machine (SVM) là một thuật toán học máy có giám sát được sử dụng cho cả nhiệm vụ phân loại và hồi quy . Mặc dù có thể áp dụng cho các bài toán hồi quy, nhưng SVM phù hợp nhất cho các nhiệm vụ phân loại . Mục tiêu chính của thuật toán SVM là xác định siêu phẳng tối ưu trong không gian N chiều có thể phân tách hiệu quả các điểm dữ liệu thành các lớp khác nhau trong không gian đặc điểm. Thuật toán đảm bảo rằng biên độ giữa các điểm gần nhất của các lớp khác nhau, được gọi là vectơ hỗ trợ , được tối đa hóa [23].



Hình : Thuật toán SVM.

### 3.4.4 K-mean clustering

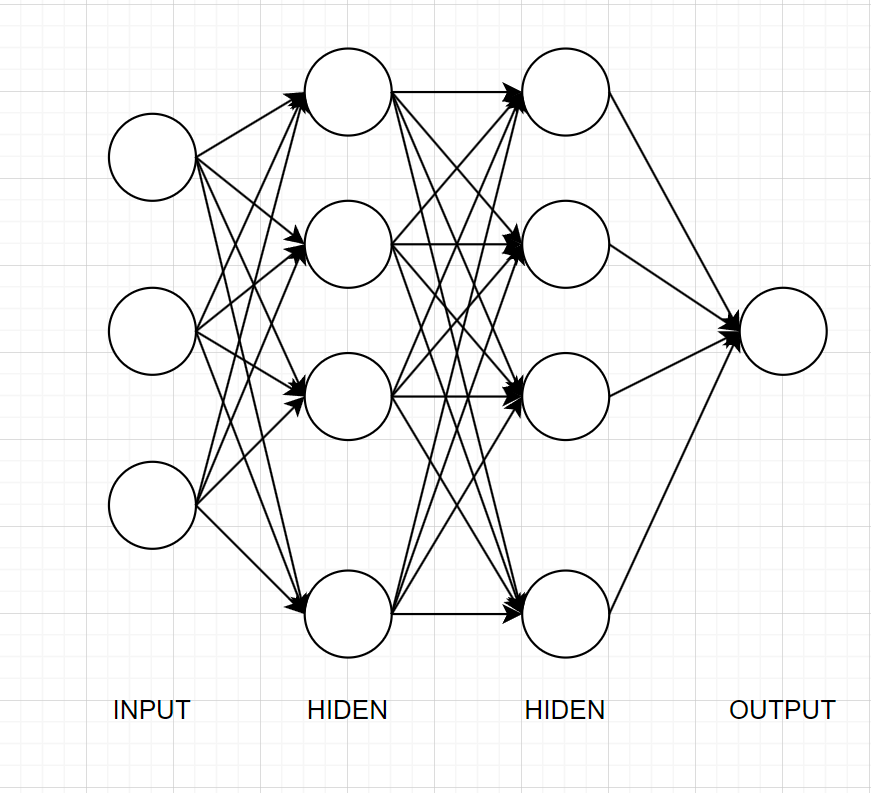
K-mean clustering là một phương pháp lượng tử hóa vectơ, ban đầu từ xử lý tín hiệu, nhằm mục đích phân chia n quan sát thành k cụm trong đó mỗi quan sát thuộc về cụm có giá trị trung bình gần nhất (trung tâm cụm hoặc trọng tâm cụm), đóng vai trò là nguyên mẫu của cụm. Điều này dẫn đến việc phân chia không gian dữ liệu thành các ô Voronoi. Phân cụm k-means giảm thiểu các phương sai trong cụm (khoảng cách Euclid bình phương), nhưng không phải khoảng cách Euclid đều, đây sẽ là bài toán Weber khó hơn: giá trị trung bình tối ưu hóa các lỗi bình phương, trong khi chỉ có trung vị hình học mới giảm thiểu khoảng cách Euclid. Ví dụ, có thể tìm thấy các giải pháp Euclid tốt hơn bằng cách sử dụng k-median và k-medoid [24].



Hình : Phân cụm trong thuật toán K-Means

### 3.4.5 Artificial neural network

Mạng nơ-ron, còn được gọi là artificial neural networks (ANN) hoặc artificially generated neural networks (SNN) là một tập hợp con của máy học cung cấp nền tảng cho các kỹ thuật học sâu. Tên và hình thức của chúng được lấy cảm hứng từ bộ não con người và chúng mô phỏng cách các nơron thực sự giao tiếp với nhau. Mạng nơ-ron nhân tạo (ANN) là các hệ thống song song lớn, bao gồm một số lượng lớn các bộ xử lý cơ bản được kết nối với nhau. Bài báo này thảo luận về mạng nơ-ron nhân tạo và các loại cơ bản của nó. Bài báo này giải thích về ANN và các phác thảo cơ bản về nơron cơ bản và mô hình máy tính nhân tạo. Bài báo mô tả các cấu trúc mạng và phương pháp học, cũng như một số ANN phổ biến nhất [25].



Hình : Mô Hình Artificial neural network.

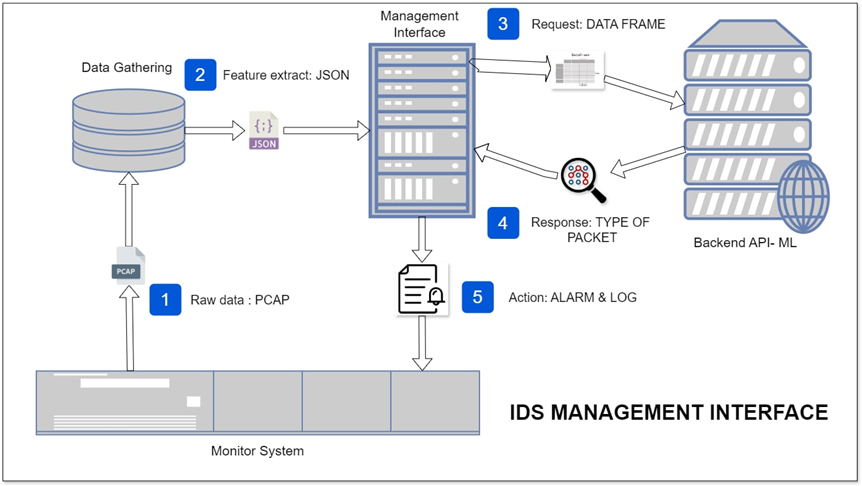
### 3.4.6 Voting and stacking model

Stacking là một kỹ thuật tổng hợp phức tạp hơn liên quan đến việc kết hợp các loại mô hình khác nhau (thường được gọi là trình học cơ sở) để cải thiện hiệu suất. Ý tưởng đằng sau stacking là tận dụng thế mạnh của một số mô hình bằng cách đào tạo một siêu mô hình (thường được gọi là mô hình cấp hai) học cách đưa ra dự đoán dựa trên đầu ra của các mô hình cơ sở.

Trong Voting, nhiều mô hình được đào tạo độc lập trên cùng một tập dữ liệu và các dự đoán của chúng được kết hợp bằng cách bỏ phiếu trong trường hợp nhiệm vụ phân loại hoặc bằng cách tính trung bình trong trường hợp nhiệm vụ hồi quy. Đây là một trong những phương pháp tập hợp đơn giản nhất và có thể được phân loại thành hai loại: bỏ phiếu cứng và bỏ phiếu mềm [26].

# **CHƯƠNG 4: XÂY DỰNG HỆ THỐNG PHÁT HIỆN XÂM NHẬP MẠNG SỬ DỤNG MACHINE LEARNING**

## 4.1. Kiến trúc hệ thống IDS đề xuất



Hình : Sơ đồ kiến trúc hệ thống IDS

Nhóm nghiên cứu đề xuất kiến trúc cho hệ thống IDS như hình trên, theo đó sẽ gồm có ba thành phần cốt lõi tương tác với hệ thống mạng được IDS bảo vệ. Ba thành phần bao gồm:

1. *Bộ phận thu thập gói tin (Data gathering):* chịu trách nhiệm thu nhập các gói tin trong hệ thống mạng ở dạng PCAP, xử lý các dữ liệu thô để trích xuất ra đặc trưng cần thiết cho mô hình học máy đã phát triển và gửi đi ở dạng JSON.
2. *Bộ phận backend API – ML:* đây là nơi tải mô hình học máy đã huấn luyện, tiếp nhận dữ liệu ở dạng data-frame để tạo đầu vào cho mô hình đưa ra dự đoán, kết quả dự đoán sẽ được trả về để cho biết luồng dữ liệu đó có phải là một hành vi xâm nhập không và nếu có thì thuộc phân lớp nào.
3. *Bộ phận quản lý (Management Interface):* đây là trung tâm điều khiển của IDS, nó tiếp nhận các JSON chứa đặc trưng và sau đó chuyển sang dạng dataframe để đưa vào backend đồng thời nhận dữ liệu API trả về để phát cảnh báo cho hệ thống. Đây là nơi tạo ra giao diện tương tác cho người dùng, mọi hoạt động điều khiển và cảnh báo sẽ được hiển thị tại đây.

Hệ thống IDS hoạt động phối hợp giữa ba thành phần với nhau và tương tác với hệ thống được bảo vệ. Chu trình diễn ra như sau:

- *1. Dữ liệu thô (PCAP) từ Hệ thống Giám sát:* IDS thu thập dữ liệu thô từ hệ thống giám sát dưới dạng các gói tin PCAP (Packet Capture).

- *2. Trích xuất Đặc trưng (JSON):* Dữ liệu thô sau khi được thu thập sẽ trải qua bước trích xuất đặc trưng và chuyển đổi thành định dạng JSON để chuẩn bị cho bước xử lý tiếp theo.

- *3. Yêu cầu Dữ liệu (Data Frame):* Giao diện Quản lý gửi yêu cầu dữ liệu dưới dạng Data Frame đến một hệ thống Backend, nơi có API và các mô hình học máy (Machine Learning - ML) để phân tích dữ liệu.

- *4. Phản hồi Loại Gói Tin:* Hệ thống Backend sẽ xử lý và trả về phản hồi cho Giao diện Quản lý, bao gồm việc nhận diện loại gói tin dựa trên mô hình học máy.

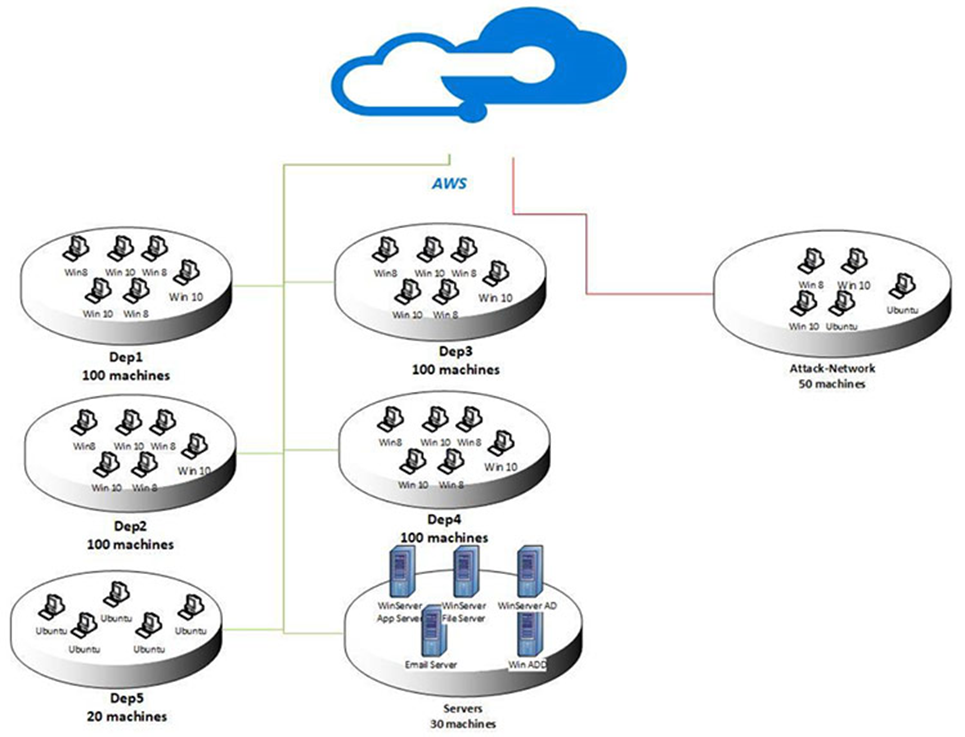
- *5. Hành động (Cảnh báo và Ghi log):* Dựa trên phản hồi từ hệ thống Backend, Giao diện Quản lý sẽ thực hiện hành động, như kích hoạt cảnh báo (Alarm) và ghi log để lưu trữ thông tin về các gói tin đáng ngờ.

## 4.2. Xử lý dữ liệu từ tập dữ liệu CIC-IDS 2018

### 4.2.1. Tổng quan về tập dữ liệu và các tùy chỉnh

Bộ dữ liệu CIC-IDS 2018 được phát triển và cung cấp bởi Trung tâm An ninh mạng Canada (Canadian Institute for Cybersecurity - CIC) thuộc Đại học New Brunswick [27]. Đây là một trong những tập dữ liệu tiêu chuẩn được sử dụng phổ biến trong nghiên cứu phát hiện xâm nhập mạng, với mục đích đánh giá hiệu suất của các hệ thống phát hiện xâm nhập dựa trên các kỹ thuật học máy và học sâu. Bộ dữ liệu CIC-IDS 2018 được xây dựng nhằm mô phỏng các kịch bản tấn công mạng phổ biến trong thực tế và cung cấp thông tin chi tiết về các hành vi của các loại tấn công, từ đó giúp phát triển và kiểm thử các thuật toán phát hiện xâm nhập.

CIC-IDS 2018 bao gồm dữ liệu thu thập từ một hệ thống mạng trong thời gian từ ngày 3 tháng 7 đến ngày 6 tháng 7 và từ ngày 9 tháng 7 đến ngày 12 tháng 7 năm 2018. Tập dữ liệu này được tạo ra từ lưu lượng truy cập mạng với sự tham gia của 25 thiết bị kết nối, bao gồm các máy chủ và thiết bị đầu cuối, được thiết lập trong môi trường mạng mô phỏng. Mỗi ngày thu thập dữ liệu được thiết kế để ghi nhận các hành vi tấn công khác nhau, nhằm đảm bảo rằng dữ liệu chứa đựng đầy đủ các kịch bản tấn công điển hình mà các hệ thống IDS có thể phải xử lý. Bộ dữ liệu này có thể được chia thành hai loại dữ liệu chính: dữ liệu tấn công và dữ liệu thông thường. Dữ liệu tấn công bao gồm nhiều loại tấn công phổ biến như DoS, DDoS, Brute Force, Web Attack, và Botnet, trong khi dữ liệu thông thường (benign data) đại diện cho lưu lượng truy cập mạng hợp lệ. [27]



Hình : Mô hình mạng mà tập dữ liệu CIC-IDS được thu thập [27]

Trong bộ dữ liệu CIC-IDS 2018, quá trình trích xuất đặc trưng được thực hiện thông qua công cụ CICFlowMeter. CICFlowMeter là trình tạo luồng lưu lượng mạng được viết bằng Java và cung cấp tính linh hoạt hơn về việc lựa chọn các tính năng bạn muốn tính toán, thêm các tính năng mới và kiểm soát tốt hơn thời lượng thời gian chờ của luồng [28]. Nó tạo ra các Luồng song hướng (Biflow), trong đó gói tin đầu tiên xác định hướng đi tới (nguồn tới đích) và hướng đi lui (đích tới nguồn), do đó 83 tính năng thống kê như Thời lượng, Số lượng gói tin, Số lượng byte, Độ dài gói tin, v.v. cũng được tính riêng theo hướng đi tới và hướng đi lui [28]. Bước trích xuất đặc trưng đóng vai trò quan trọng trong việc biến đổi dữ liệu ban đầu thành các thông tin đại diện cho các mẫu hành vi mạng, giúp cải thiện độ chính xác và hiệu suất của các thuật toán phát hiện xâm nhập.

Bộ dữ liệu CIC-IDS 2018 bao gồm hơn 80 đặc trưng được trích xuất từ các gói tin và luồng dữ liệu mạng (network flows), từ đó cung cấp các thuộc tính phân biệt giữa các loại kết nối bình thường và tấn công [27]. Những đặc trưng này bao gồm:

* *Thông tin kết nối cơ bản:*

Địa chỉ IP nguồn và đích: Đóng vai trò xác định thiết bị gửi và nhận, giúp nhận diện các kết nối đáng ngờ giữa các thiết bị mạng.

Cổng nguồn và cổng đích: Giúp xác định dịch vụ hoặc ứng dụng được sử dụng trong kết nối, từ đó cung cấp manh mối về mục tiêu của cuộc tấn công (như Port Scan).

Giao thức (Protocol): Chỉ ra loại giao thức truyền tải, chẳng hạn như TCP, UDP, hoặc ICMP, từ đó giúp phân tích các kiểu tấn công điển hình liên quan đến từng loại giao thức.

* *Đặc trưng về luồng dữ liệu (Flow Statistics):*

Thời gian bắt đầu và kết thúc luồng (Start and End Time): Đặc trưng về thời gian giúp phát hiện các mẫu lưu lượng truy cập, chẳng hạn như các tấn công từ chối dịch vụ (DoS) thường diễn ra liên tục trong một khoảng thời gian ngắn.

Tổng số gói tin (Total Packets) và tổng số byte (Total Bytes): Những đặc trưng này giúp đo lường khối lượng và tần suất của các luồng dữ liệu. Một khối lượng lớn dữ liệu gửi đến trong một thời gian ngắn có thể là dấu hiệu của tấn công DDoS hoặc các loại tấn công khác.

Tốc độ dữ liệu (Data Rate): Đặc trưng này giúp phát hiện các cuộc tấn công khi tốc độ dữ liệu gửi đến tăng đột biến bất thường.

* *Đặc trưng thống kê trong luồng dữ liệu:*

Độ dài trung bình của gói tin (Average Packet Length) và độ lệch chuẩn của độ dài gói tin (Packet Length Variance): Đặc trưng này cung cấp thông tin về các thay đổi trong luồng dữ liệu. Các luồng dữ liệu với độ dài gói tin ổn định thường chỉ ra kết nối thông thường, trong khi sự thay đổi lớn có thể là dấu hiệu của hoạt động xâm nhập.

Thời gian giữa các gói tin (Time Between Packets): Những thay đổi lớn trong thời gian giữa các gói tin có thể là dấu hiệu của hành vi bất thường trong mạng, chẳng hạn như tấn công Brute Force.

* *Đặc trưng về hành vi mạng nâng cao:*

Tần suất gói tin gửi và nhận (Packet Frequency): Giúp xác định các mẫu hành vi bất thường khi tần suất dữ liệu tăng đột ngột.

Tỷ lệ giữa các gói tin đi và về (Forward/Backward Ratio): Đây là tỷ lệ giữa các gói tin gửi đi và nhận lại. Các tấn công như Ping Flood có thể có tỷ lệ này khác thường, vì tấn công gửi nhiều gói tin đi nhưng không nhận lại. Cấu trúc này không chỉ hỗ trợ cho việc phân tích mạng mà còn tạo điều kiện để các thuật toán học máy áp dụng trong phát hiện xâm nhập mạng có đủ thông tin cần thiết để nhận diện các mẫu hành vi mạng bất thường.

Sự phong phú và tính đa dạng trong dữ liệu của CIC-IDS 2018 giúp tăng cường khả năng ứng dụng của các hệ thống IDS. Nhờ vào việc bao gồm các mẫu tấn công phức tạp như Infiltration (xâm nhập), Heartbleed, và các kiểu tấn công mạng tinh vi khác, CIC-IDS 2018 cung cấp một nền tảng phong phú cho việc nghiên cứu và phát triển các thuật toán phát hiện tấn công nâng cao. Tuy nhiên, một thách thức của bộ dữ liệu này là khối lượng dữ liệu lớn, đòi hỏi khả năng xử lý và lưu trữ cao, đặc biệt trong giai đoạn tiền xử lý và huấn luyện mô hình học máy. Các nhà nghiên cứu cần áp dụng các kỹ thuật tiền xử lý phù hợp để lọc bỏ các dữ liệu dư thừa và chuẩn hóa dữ liệu, từ đó tối ưu hóa hiệu quả và tốc độ của các mô hình học máy.

Bộ dữ liệu CIC-IDS 2018 đã trở thành một trong những tiêu chuẩn thử nghiệm phổ biến nhất trong nghiên cứu an ninh mạng, giúp các nhà nghiên cứu phát triển và đánh giá hiệu suất của các mô hình học máy và hệ thống IDS. Việc sử dụng CIC-IDS 2018 không chỉ cung cấp một nguồn dữ liệu phong phú mà còn tạo điều kiện so sánh kết quả giữa các nghiên cứu khác nhau trong cùng một lĩnh vực. Sự đa dạng về loại hình tấn công và dữ liệu mạng của CIC-IDS 2018 đã giúp các nghiên cứu đạt được những tiến bộ vượt bậc trong phát hiện tấn công mạng và củng cố khả năng bảo vệ của hệ thống IDS.

Dữ liệu từ tập CIC-IDS 2018 đã được tùy chỉnh và tiền xử lý nhằm phục vụ cho mục tiêu nghiên cứu phát hiện xâm nhập mạng một cách hiệu quả hơn. Các thao tác chính bao gồm việc kết hợp nhiều tệp dữ liệu khác nhau từ CIC-IDS 2018, thực hiện gán nhãn lại cho các loại tấn công theo nhóm, và lưu kết quả vào tệp dữ liệu mới.

Dữ liệu từ các ngày khác nhau trong tập CIC-IDS 2018 được chia thành nhiều tệp CSV, mỗi tệp mô phỏng một ngày làm việc với các hoạt động tấn công và lưu lượng truy cập bình thường. Các tệp dữ liệu bao gồm những ngày chứa các cuộc tấn công như DDoS, PortScan, Infiltration, và WebAttacks, cùng với các tệp chứa lưu lượng mạng thông thường. Mã nguồn trên sử dụng thư viện pandas để đọc các tệp CSV này, sau đó kết hợp chúng thành một tệp dữ liệu lớn chứa toàn bộ các mẫu (samples) để thuận tiện cho các bước xử lý tiếp theo. [Phụ lục 1]

Dữ liệu CIC-IDS 2018 chứa nhiều loại tấn công khác nhau với các nhãn chi tiết, nhưng để giảm độ phức tạp và tăng hiệu quả phát hiện, các loại tấn công này được gộp lại thành các nhóm lớn. Cụ thể [Phụ lục 1]:

* *DoS/DDoS:* Các nhãn như 'DoS', 'DDoS', 'DoS GoldenEye', 'DoS Hulk', 'DoS Slowhttptest', và 'DoS slowloris' được gộp thành một nhãn chung là 'DoS/DDoS'. Điều này giúp giảm thiểu sự phân tán của các nhãn và tăng cường khả năng phân loại của mô hình.
* *BruteForce:* Các nhãn tấn công bẻ khóa SSH và FTP ('SSH-Patator', 'FTP-Patator') được gộp lại thành nhóm 'BruteForce', đại diện cho các kiểu tấn công brute-force nhắm vào các dịch vụ khác nhau.
* *WebAttacks:* Các nhãn liên quan đến tấn công web ('Web Attack Brute Force', 'Web Attack Sql Injection', 'Web Attack XSS') được gộp vào nhóm 'WebAttacks', nhằm tạo điều kiện cho mô hình nhận diện các cuộc tấn công vào lớp ứng dụng.
* *OtherAttacks:* Một số tấn công hiếm gặp hoặc ít phổ biến hơn như 'Infiltration' và 'Heartbleed' được gộp chung vào nhóm 'OtherAttacks'.
* *Botnet:* Nhóm dữ liệu này giữ nguyên nhãn 'Bot', đại diện cho hoạt động của các botnet.
* *Dữ liệu thông thường (benign):* giữ nguyên nhãn 'BENIGN'.

Sau khi gán lại nhãn, các nhóm dữ liệu tấn công và dữ liệu thông thường được kết hợp vào một tập dữ liệu hoàn chỉnh. Tập dữ liệu này được sắp xếp lại theo chỉ số ban đầu để đảm bảo tính nhất quán. Kết quả của quá trình tiền xử lý và tùy chỉnh được lưu trữ vào tệp 'data\_processed.csv' để sử dụng trong quá trình huấn luyện mô hình và đánh giá hệ thống phát hiện xâm nhập mạng (IDS) [Phụ lục 1].

Việc nhóm lại các loại tấn công khác nhau giúp đơn giản hóa bài toán phân loại và tăng cường khả năng phát hiện của mô hình học máy, vì dữ liệu được tổ chức thành các lớp rõ ràng hơn. Bên cạnh đó, các loại tấn công khác nhau được nhóm thành các lớp lớn giúp giảm thiểu tình trạng mất cân bằng dữ liệu, giúp mô hình học máy dễ dàng phân biệt các loại tấn công phổ biến và hiếm gặp, từ đó cải thiện hiệu quả tổng thể của hệ thống phát hiện xâm nhập mạng.

### 4.2.2 Tiền xử lý và chuẩn hóa dữ liệu

Dữ liệu từ tập CIC-IDS 2018 đã trải qua quy trình tiền xử lý và chuẩn hóa nhằm đảm bảo tính toàn vẹn, loại bỏ nhiễu và chuẩn bị cho quá trình huấn luyện mô hình phát hiện xâm nhập mạng. Các bước xử lý này gồm ba phần chính: làm sạch dữ liệu, chuẩn hóa đặc trưng, và cân bằng lớp dữ liệu.

Làm sạch dữ liệu: hàm load\_and\_clean\_data() đảm nhiệm vai trò làm sạch dữ liệu thô từ tệp đầu vào, bao gồm [Phụ lục 2]:

* Loại bỏ giá trị thiếu và dữ liệu trùng lặp: Các hàng chứa giá trị thiếu hoặc trùng lặp được xóa bỏ nhằm tăng tính nhất quán cho tập dữ liệu.
* Xử lý các giá trị vô hạn và giá trị ngoại lai: Các đặc trưng số học của dữ liệu được kiểm tra và loại bỏ những giá trị vô hạn (dạng ±∞) và những giá trị lớn bất thường, đảm bảo rằng tập dữ liệu không chứa các ngoại lệ có thể ảnh hưởng tiêu cực đến mô hình học máy.

Sau bước này, chỉ những hàng dữ liệu sạch và không chứa giá trị bất thường mới được giữ lại để tiếp tục quá trình tiền xử lý. Tiếp theo là bước chuẩn hóa và mã hóa dữ liệu. Hàm preprocess\_and\_split\_data() thực hiện các bước chuẩn hóa và mã hóa nhãn cho dữ liệu [Phụ lục 2]:

* Chuẩn hóa đặc trưng số: Sử dụng bộ chuẩn hóa StandardScaler, các cột số trong tập dữ liệu được chuyển đổi thành dạng chuẩn hóa với trung bình bằng 0 và độ lệch chuẩn bằng 1. Điều này giúp các đặc trưng có độ lớn khác nhau trở nên đồng nhất, làm giảm độ nhạy của mô hình đối với sự khác biệt về đơn vị và thang đo, giúp mô hình học hiệu quả hơn.
* Mã hóa nhãn dữ liệu: Cột nhãn 'Label' được mã hóa bằng LabelEncoder, chuyển đổi các nhãn dạng chuỗi (như 'BENIGN', 'DoS/DDoS', 'PortScan') thành các giá trị số. Mã hóa này giúp mô hình học máy dễ dàng xử lý và phân biệt các lớp tấn công khác nhau.

Bước cuối ở giai đoạn này là chia tách dữ liệu và cân bằng lớp. Sau khi chuẩn hóa và mã hóa, tập dữ liệu được chia thành các tập huấn luyện, kiểm định, và kiểm tra, với tỷ lệ chia cố định [Phụ lục 2]:

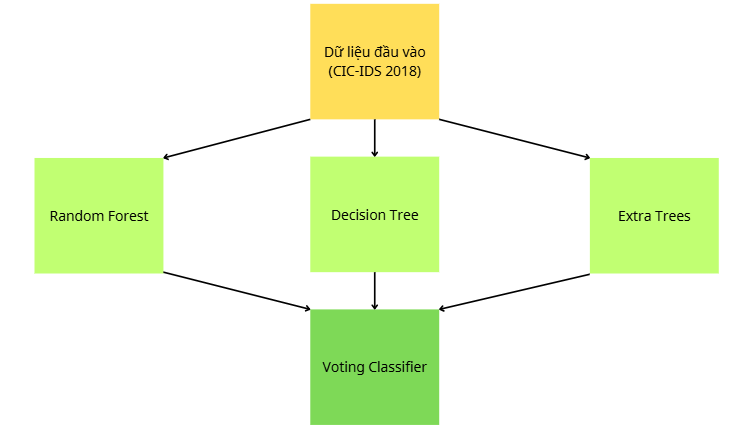
* Tập dữ liệu đầu tiên được chia thành train (80%) và test (20%).
* Tiếp đó, tập huấn luyện được tiếp tục chia thành train (90%) và validation (10%), nhằm tạo điều kiện để tối ưu hóa và kiểm tra hiệu suất của mô hình.
* Để khắc phục tình trạng mất cân bằng lớp, phương pháp Synthetic Minority Over-sampling Technique (SMOTE) được áp dụng lên tập huấn luyện. SMOTE giúp tạo ra các mẫu dữ liệu mới cho các lớp có ít dữ liệu bằng cách tổng hợp từ các điểm gần nhau trong không gian đặc trưng của các lớp này. Việc cân bằng lớp giúp mô hình học máy đạt hiệu suất cao và không bị thiên vị bởi các lớp dữ liệu chiếm ưu thế.

Qua quá trình tiền xử lý, dữ liệu từ tập CIC-IDS 2018 được làm sạch, chuẩn hóa và cân bằng, giúp tăng cường hiệu quả của mô hình học máy trong phát hiện xâm nhập. Việc áp dụng SMOTE giúp giảm thiểu các vấn đề liên quan đến mất cân bằng dữ liệu, trong khi chuẩn hóa đặc trưng và mã hóa nhãn giúp tăng tính đồng nhất và đảm bảo dữ liệu đầu vào cho mô hình đạt yêu cầu về chất lượng và cấu trúc.

## 4.3. Xây dựng và huấn luyện mô hình

### 4.3.1 Thiết kế mô hình

Trong phần này, mô hình phát hiện xâm nhập mạng được thiết kế với đội ngũ các thuật toán học máy phức tạp, bao gồm Random Forest, Decision Tree, Extra Trees, kết hợp trong mô hình Voting Classifier. Việc kết hợp nhiều thuật toán giúp tận dụng được điểm mạnh của từng mô hình để tăng cường tính độ chính xác và tính ổn định khi phát hiện các cuộc tấn công mạng.



Hình : Các thuật toán trong mô hình học máy

Random Forest là một mô hình học máy thuộc nhóm ensemble learning, sử dụng nhiều cây quyết định (Decision Trees) để dự đoán và kết hợp kết quả của chúng. Đặc điểm nổi bật của Random Forest là khả năng giảm thiểu độ lệch, giúp giảm thiểu độ nhạy với dữ liệu nhiễu nhiễu và tăng tính ổn định của mô hình. Trong dự án này, RandomForestClassifier được cấu hình với 500 cây quyết định (được xác định bởi tham số N\_ESTIMATORS), từ đó tăng độ chính xác và tính ổn định trong phân loại các loại tấn công khác nhau [Phụ lục 3].

Decision Tree là một trong những mô hình cơ bản và dễ hiểu nhất trong học máy, được sử dụng để phân loại bằng cách tìm hiểu các quy tắc ra quyết định từ dữ liệu. Thuật toán DecisionTreeClassifier được tích hợp và có cân nhắc đến cân bằng lớp (class\_weight=balanced), để đảm bảo khả năng xử lý các dữ liệu có sự mất cân bằng giữa các lớp [Phụ lục 3].

Extra Trees (Extremely Randomized Trees) là một mở rộng của Random Forest, được sử dụng để tăng cường tính đa dạng và giảm thiểu overfitting. Extra Trees khác biệt với Random Forest ở chỗ nó sinh ra ngẫu nhiên các điểm chia (splits) thay vì tìm điểm tốt nhất, giúp giảm tính phụ thuộc và tăng độ chính xác cho mô hình. Tương tự Random Forest, Extra Trees cũng được cấu hình với 500 cây quyết định, giúp cung cấp sự đa dạng và ổn định.

Trong dự án này, các thuật toán Random Forest, Decision Tree, và Extra Trees được kết hợp vào một mô hình Voting Classifier. Voting Classifier được thiết kế để kết hợp kết quả dự đoán từ nhiều mô hình con, tạo ra một kết quả cuối cùng bằng cách lấy ý kiến đa số (‘voting='hard'’) [Phụ lục 3]. Cách tiếp cận này tận dụng được điểm mạnh của từng mô hình để giảm thiểu nhược điểm riêng lẻ của chúng, tạo ra một mô hình cuối cùng ổn định hơn và có khả năng tích hợp các quyết định từ nhiều góc nhìn khác nhau. Việc sử dụng Voting Classifier giúp tăng độ chính xác trong phát hiện tấn công mạng, đặc biệt là đối phó với các loại tấn công phức tạp. Sự kết hợp nhiều thuật toán khác nhau giúp tạo nên sự đa dạng về khả năng phân loại, giảm thiểu nguy cơ độ lệch trong mô hình và đảm bảo tính khái quát hóa cao khi đối phó với các loại dữ liệu khác nhau.

### 4.3.2. Huấn luyện mô hình và tinh chỉnh tham số

Quá trình huấn luyện và tối ưu hóa mô hình phát hiện xâm nhập trong hệ thống Intrusion Detection System (IDS) bao gồm hai bước quan trọng: chọn lọc đặc trưng và huấn luyện mô hình kết hợp với tinh chỉnh tham số. Mục tiêu của quá trình này là tăng độ chính xác và hiệu suất của mô hình trong phát hiện xâm nhập mạng.

Chọn lọc đặc trưng giúp loại bỏ các đặc trưng không cần thiết, tối ưu hóa tốc độ tính toán và giảm thiểu nhiễu. Mô hình dùng phương pháp chọn lọc đặc trưng dựa trên độ quan trọng, được tính từ feature\_importance\_optimized [Phụ lục 4]:

* Đo lường độ quan trọng của đặc trưng*:* “feature\_importance\_optimized()” sử dụng một mô hình Voting Classifier, tính độ quan trọng của từng đặc trưng bằng cách lấy trung bình các trọng số đặc trưng từ các mô hình thành phần. Điều này cung cấp một đánh giá toàn diện về mức độ ảnh hưởng của các đặc trưng trong việc phát hiện các loại xâm nhập.
* Chọn đặc trưng quan trọng nhất: Sau khi tính toán độ quan trọng, các đặc trưng được sắp xếp theo thứ tự giảm dần của giá trị quan trọng. Các đặc trưng có tổng độ quan trọng cộng dồn đạt tới 90% được chọn để xây dựng mô hình, nhằm giữ lại các đặc trưng có giá trị cao nhất và loại bỏ các đặc trưng ít ảnh hưởng.
* Lưu trữ danh sách đặc trưng: Danh sách các đặc trưng đã chọn được lưu vào một tệp, cho phép quá trình huấn luyện có thể tái sử dụng và dễ dàng theo dõi. Việc này giúp tối ưu hóa việc xử lý dữ liệu và đảm bảo tính nhất quán trong các lần huấn luyện tiếp theo.

Sau khi xác định các đặc trưng cần thiết, bước tiếp theo là huấn luyện mô hình và tinh chỉnh các tham số [Phụ lục 4]:

* Chuẩn bị dữ liệu: Hàm “preprocess\_and\_split\_data()” sử dụng các đặc trưng đã chọn để chuẩn bị và phân chia dữ liệu thành các tập huấn luyện, kiểm định, và kiểm tra.
* Huấn luyện mô hình Voting Classifier: Hàm “train\_model()” thực hiện huấn luyện mô hình “VotingClassifier”, một mô hình kết hợp các mô hình thành phần khác như Random Forest, Decision Tree, và Support Vector Machine. Voting Classifier giúp cải thiện hiệu suất nhờ tính ổn định và sự đa dạng của các mô hình thành phần, tối ưu hóa khả năng tổng quát của mô hình trên dữ liệu chưa biết.
* Đánh giá mô hình và tinh chỉnh: Mô hình ban đầu được đánh giá trên tập kiểm định để xác định hiệu suất. Nếu độ chính xác chưa đạt yêu cầu, quá trình chọn lọc đặc trưng sẽ được lặp lại với bộ đặc trưng đã tối ưu hóa. Khi đạt được hiệu suất tốt trên tập kiểm định, mô hình sẽ được kiểm tra lần cuối trên tập kiểm tra để đánh giá khả năng tổng quát.

Sau khi huấn luyện hoàn tất, mô hình cùng với các bộ chuẩn hóa “scaler” và “label encoder” được lưu trữ bằng hàm “save\_models()” để có thể dễ dàng tải lại và sử dụng trong hệ thống IDS thực tế mà không cần phải tái huấn luyện [Phụ lục 4].

Bằng cách chọn lọc đặc trưng, huấn luyện Voting Classifier, và tinh chỉnh tham số, mô hình được tối ưu hóa cả về hiệu suất và độ chính xác. Quy trình này giúp mô hình học máy phát hiện xâm nhập có khả năng nhận diện nhanh chóng và chính xác các hành vi bất thường trên mạng, góp phần nâng cao hiệu quả và độ tin cậy của hệ thống IDS trong phát hiện các mối đe dọa tiềm ẩn.

## 4.4 Xây dựng hệ thống IDS

Dựa trên kiến trúc đã thiết kế và trình bày ở phần 4.1, nhóm nghiên cứu tiến hành xây dựng từng thành phần trong hệ thống IDS được đề xuất để đánh giá hiệu quả hoạt động của hệ thống. Theo đó sẽ bao gồm ba thành cốt lõi dưới đây:

### 4.4.1 Data Gathering

Để xây dựng một hệ thống IDS hiệu quả, bước đầu tiên và quan trọng là thu thập dữ liệu từ mạng nhằm phát hiện các hành vi bất thường. Quy trình thu thập dữ liệu từ mạng bao gồm các bước dưới đây.

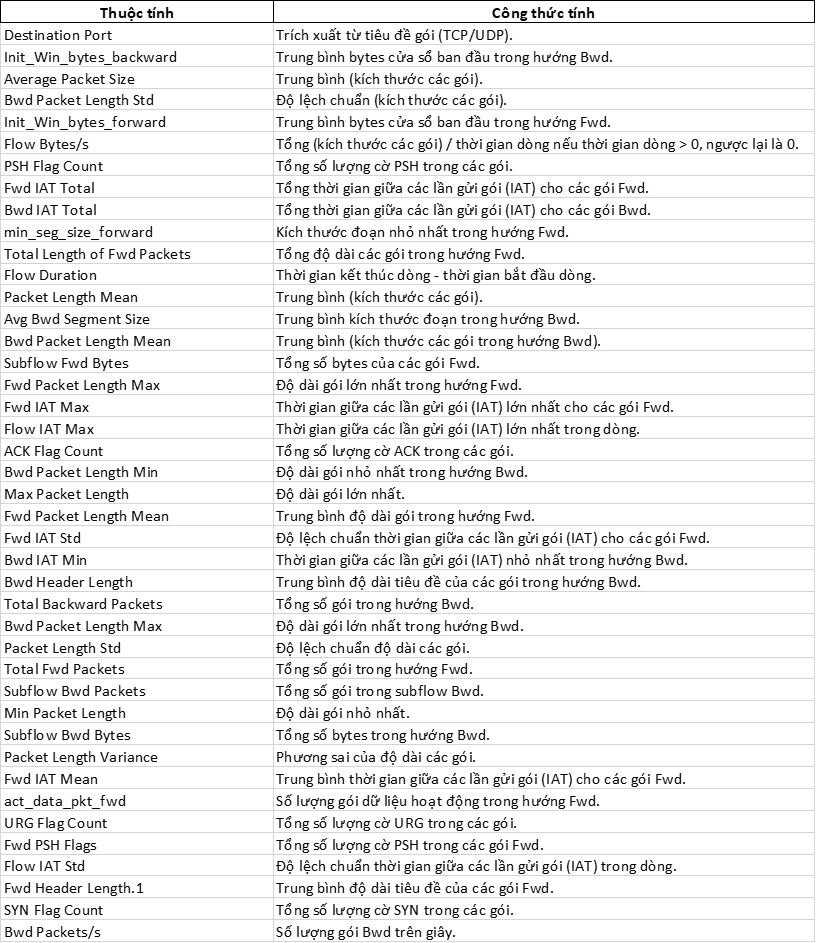
Hàm capture\_packets() sử dụng thư viện scapy để bắt gói tin trong thời gian xác định, cụ thể ở đây là 60 giây trên giao diện mạng "Wi-Fi" [Phụ lục 5]. Mỗi gói tin bắt được chứa nhiều thông tin, bao gồm địa chỉ IP nguồn và đích, các cổng nguồn và đích, và các thuộc tính giao thức (TCP hoặc UDP). Các gói tin được lưu trữ để xử lý và tính toán đặc trưng, làm cơ sở để phát hiện các hành vi xâm nhập bất thường.

Mỗi gói tin được phân loại và nhóm thành các "luồng" dựa trên các yếu tố như địa chỉ IP nguồn, địa chỉ IP đích, cổng nguồn, và cổng đích. Hàm extract\_flow\_key() tạo ra một khóa định danh cho từng luồng mạng, giúp phân biệt các luồng khác nhau trong dữ liệu mạng [Phụ lục 5].

Bằng cách nhóm các gói tin lại thành các luồng mạng, ta có thể tính toán các đặc trưng quan trọng, ví dụ như thời gian giữa các gói tin, số lượng gói tin trong mỗi luồng, và tổng số byte được truyền tải.

Sau khi phân loại thành các luồng, hàm process\_packets() tính toán các đặc trưng của từng luồng để phục vụ cho việc huấn luyện mô hình học máy [Phụ lục 5]. Các đặc trưng này được chọn dựa trên sự liên quan của chúng trong phát hiện các hành vi bất thường và các cuộc tấn công trong mạng, bao gồm:

Bảng 4: Các thuộc tính được trích xuất



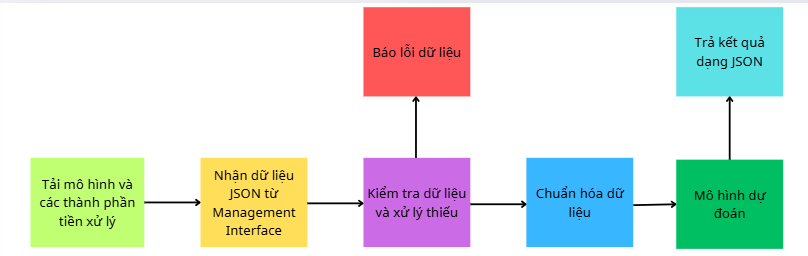
Trong quá trình tính toán, các giá trị đặc trưng có thể không hợp lệ, chẳng hạn như giá trị NaN hoặc vô cùng (infinity). Do đó, mã nguồn thay thế các giá trị không hợp lệ này bằng giá trị mặc định (0) để đảm bảo rằng các đặc trưng là hoàn chỉnh và có thể sử dụng trực tiếp trong các bước tiền xử lý và huấn luyện mô hình tiếp theo.

Sau khi tính toán, các đặc trưng của mỗi luồng được lưu vào pandas DataFrame, giúp hệ thống IDS dễ dàng xử lý và phân tích dữ liệu trong các bước tiếp theo như tiền xử lý, chuẩn hóa và dự đoán [Phụ lục 5].

Nhờ quy trình thu thập dữ liệu và tính toán đặc trưng này, hệ thống IDS có thể liên tục theo dõi và phân tích các luồng mạng theo thời gian thực. Điều này tạo nền tảng quan trọng cho quá trình phát hiện xâm nhập tự động, cho phép phát hiện các hành vi bất thường và các cuộc tấn công trong mạng một cách hiệu quả.

### 4.4.2. Backend API

Hệ thống IDS cần một API để nhận dữ liệu mạng, thực hiện dự đoán và trả về kết quả cho người dùng hoặc hệ thống giám sát. Đoạn mã trên xây dựng một dịch vụ RESTful API sử dụng Flask, cung cấp khả năng xử lý yêu cầu với dữ liệu về lưu lượng mạng và dự đoán trạng thái xâm nhập theo thời gian thực. Về cấu trúc của API [Phụ lục 6]:



Hình : Sơ đồ hoạt động của hệ thống Backend API - ML

* *Tải mô hình và các thành phần hỗ trợ:* Các thành phần như “voting\_model” (mô hình phát hiện xâm nhập), “scaler” (bộ chuẩn hóa dữ liệu), và “label\_encoder” (mã hóa nhãn) được tải từ các tệp lưu trước đó bằng “joblib”. Điều này giúp tái sử dụng mô hình đã huấn luyện, đảm bảo nhất quán khi áp dụng vào dữ liệu mới. Trong trường hợp không thể tải được các thành phần, API sẽ ghi lại lỗi và dừng hoạt động.
* *Thiết kế - “/api/evaluate”:* End point “evaluate()” nhận các yêu cầu “POST”, xử lý dữ liệu trong “request.json” từ phía người dùng. Yêu cầu phải bao gồm dữ liệu các đặc trưng mạng trong định dạng JSON. API kiểm tra xem dữ liệu đầu vào có đầy đủ các đặc trưng cần thiết, được định nghĩa trong danh sách “EXPECTED\_COLUMNS”, nhằm đảm bảo mô hình có đủ thông tin để thực hiện dự đoán chính xác. Nếu thiếu bất kỳ đặc trưng nào, API sẽ trả về lỗi và danh sách các cột còn thiếu.
* *Tiền xử lý và dự đoán:* Sau khi xác minh cấu trúc dữ liệu, API chuyển đổi các đặc trưng về định dạng chuẩn đã sử dụng trong quá trình huấn luyện mô hình. Dữ liệu đầu vào được chuẩn hóa bằng “scaler” trước khi đưa vào mô hình phát hiện. Sau đó, mô hình dự đoán trạng thái của từng luồng mạng trong dữ liệu. Các nhãn dự đoán là các giá trị số, vì vậy “label\_encoder” được sử dụng để chuyển đổi các nhãn này về định dạng ký tự (chẳng hạn như "normal" hoặc "attack"), giúp người dùng dễ hiểu hơn.
* *Trả về kết quả:* API trả về các nhãn dự đoán của mô hình dưới dạng một danh sách JSON, bao gồm từng nhãn cho mỗi luồng mạng. Điều này giúp hệ thống giám sát có thể xem chi tiết trạng thái của từng luồng.

Mặc dù API hiện tại đã thực hiện chức năng nhận dữ liệu mạng và dự đoán trạng thái xâm nhập, hệ thống vẫn cần nhiều cải tiến để có thể sử dụng hiệu quả trong môi trường thực tế. Một số hướng phát triển bao gồm:

* *Tăng cường bảo mật cho API:* Hệ thống IDS yêu cầu bảo mật cao để tránh tình trạng API bị tấn công hoặc lợi dụng. Có thể bổ sung các cơ chế xác thực (authentication), mã hóa SSL và giới hạn tốc độ truy cập.
* *Tối ưu hóa hiệu năng:* Ở môi trường thực tế, IDS phải xử lý lượng lớn dữ liệu trong thời gian thực. Có thể cải thiện hiệu năng bằng cách tối ưu hóa mô hình và triển khai cơ chế xử lý đồng thời, nhằm đảm bảo độ trễ thấp.
* *Cập nhật mô hình tự động:* Để mô hình có thể nhận diện các kiểu tấn công mới, hệ thống nên có khả năng cập nhật mô hình định kỳ hoặc liên tục học từ dữ liệu mới. Điều này giúp cải thiện khả năng phát hiện xâm nhập của IDS theo thời gian.
* *Khả năng mở rộng (scalability):* Khi triển khai trong mạng lớn, cần đảm bảo rằng API có thể xử lý lưu lượng tăng cao bằng cách sử dụng các công nghệ như Docker hoặc Kubernetes để triển khai dưới dạng microservice.

Với các cải tiến trên, API sẽ có khả năng đáp ứng yêu cầu vận hành IDS trong môi trường mạng thực tế, giúp phát hiện và ngăn chặn các cuộc tấn công một cách hiệu quả và kịp thời.

### 4.4.3 Giao diện Quản lý (Management Interface)

Nhóm nghiên cứu đã phát triển một giao diện quản lý cơ bản để kiểm nghiệm tính năng phát hiện xâm nhập của hệ thống. Giao diện này được xây dựng dưới dạng ứng dụng dòng lệnh đơn giản, cho phép người dùng thực hiện các tác vụ chính như thu thập gói tin mạng, gửi dữ liệu tới API phân tích và nhận kết quả phát hiện. Dù giao diện hiện tại chỉ ở mức thử nghiệm, hệ thống này đóng vai trò quan trọng trong việc xác minh các chức năng cơ bản của IDS và đánh giá hiệu quả bước đầu của hệ thống. Các chức năng hiện tại của Giao diện Quản lý [Phụ lục 7]:

* *Chức năng Thu Thập Gói Tin Mạng:* khi người dùng chọn "Start IDS," hệ thống sẽ khởi động quá trình thu thập gói tin từ một giao diện mạng nhất định, ví dụ như “Wi-Fi”. Dữ liệu này sau đó được chuyển đổi thành các đặc trưng có ý nghĩa trong việc phát hiện xâm nhập. Module “packet\_capture” chịu trách nhiệm thu thập và xử lý các gói tin này theo thời gian thực, từ đó phân loại chúng vào các luồng dữ liệu khác nhau để phục vụ cho quá trình đánh giá.
* *Gửi Dữ liệu Tới Máy Chủ Phân Tích (API):* dữ liệu được ghi lại dưới dạng tệp JSON đi kèm với timestamp, sau đó được gửi tới API “/api/evaluate” trên server để phân tích. Việc lưu trữ dữ liệu với timestamp giúp nhóm nghiên cứu dễ dàng quản lý và kiểm tra lại dữ liệu đã thu thập cho từng phiên thử nghiệm. API này sẽ sử dụng mô hình học máy để đưa ra các nhãn phân loại, từ đó giúp nhận diện các hoạt động bất thường trong mạng.
* *Hiển Thị Kết Quả dưới dạng Bảng:* kết quả phân tích từ server, bao gồm nhãn phân loại cho từng luồng dữ liệu, được hiển thị dưới dạng bảng cho người dùng. Nhờ đó, người quản trị có thể dễ dàng theo dõi và phân tích trạng thái an ninh của từng luồng mạng, đồng thời nhận diện nhanh chóng các hoạt động nghi ngờ.

Mặc dù giao diện quản lý hiện tại chủ yếu phục vụ cho mục đích thử nghiệm, nhóm nghiên cứu đã xác định các tính năng và cải tiến cần thiết để đưa hệ thống IDS vào thực tế, nhằm ứng dụng rộng rãi trong các mạng doanh nghiệp và tổ chức. Các tính năng nâng cao sẽ giúp hệ thống đạt được độ tin cậy và khả năng giám sát mạng tối ưu.

Nhóm nghiên cứu dự kiến phát triển một giao diện đồ họa trên nền web hoặc desktop để thay thế giao diện dòng lệnh hiện tại. Giao diện đồ họa này sẽ mang lại trải nghiệm người dùng thân thiện hơn, cung cấp các tính năng trực quan như dashboard hiển thị trạng thái hệ thống theo thời gian thực, bảng điều khiển cấu hình hệ thống, và báo cáo chi tiết. Điều này giúp người quản trị có thể thao tác dễ dàng và giám sát toàn diện các hoạt động của hệ thống. Để nâng cao khả năng phát hiện và xử lý, giao diện quản lý sẽ được tích hợp tính năng cập nhật thời gian thực nhằm cho phép người quản trị giám sát liên tục trạng thái của hệ thống. Khi phát hiện bất kỳ hoạt động nghi ngờ hoặc sự cố an ninh nào, hệ thống sẽ cảnh báo ngay lập tức. Tính năng này rất quan trọng trong các môi trường mạng có yêu cầu cao về an ninh và độ nhạy cảm với thời gian.

Nhóm nghiên cứu nhận thấy rằng các tính năng và cải tiến trên sẽ giúp hệ thống IDS trở nên toàn diện và có khả năng bảo vệ mạnh mẽ hơn trước các mối đe dọa an ninh mạng trong môi trường thực tế. Chúng tôi kỳ vọng hệ thống sẽ đáp ứng được yêu cầu ngày càng cao của các tổ chức và doanh nghiệp trong việc giám sát và bảo vệ an ninh mạng.

# **CHƯƠNG 5: THỰC NGHIỆM VÀ ĐÁNH GIÁ**

## 5.1. Đánh giá mô hình học máy

Bảng 5: Kết quả test của mô hình từ bộ dữ liệu CIC-IDS-2018

|  |  |  |  |
| --- | --- | --- | --- |
| Classification | Precision (%) | Recall (%) | f1-score (%) |
| BENIGN | 99.93 | 99.92 | 99.93 |
| Bot | 87.94 | 76.86 | 82.93 |
| BruteForce | 99.95 | 100 | 99.97 |
| DoS/DDoS | 99.85 | 99.92 | 99.89 |
| OtherAttacks | 100 | 66.67 | 80.00 |
| PortScan | 98.94 | 99.27 | 99.11 |
| WebAttacks | 99.05 | 97.20 | 98.12 |

*Tổng quan hiệu suất mô hình*

Hiệu suất mô hình đạt hiệu suất cao với độ chính xác 99,88% (có xét trọng số) và 97,95% (không xét trọng số), cho thấy rằng mô hình đã dự đoán chính xác gần như toàn bộ tập dữ liệu kiểm tra.

Mô hình đã dự đoán rất tốt các mẫu thuộc lớp BENIGN với cả precision và recall đều đạt mức tốt lần lượt là 99.93% và 99.92%, điều này thể hiện khả năng phân loại chính xác tuyệt đối các trường hợp không phải là tấn công, chứng tỏ mô hình rất hiệu quả trong việc xử lý các mẫu bình thường. Đối với lớp Bot, mặc dù precision và recall lần lượt đạt 87.94 và 76.86%, điều này cho thấy mô hình vẫn có khả năng nhận diện tốt các mẫu thuộc lớp này, nhưng vẫn có một số lỗi trong dự đoán (bao gồm cả false positive và false negative).

Lớp BruteForce có precision, recall, và F1-score đạt lần lượt: 99.95%, 100% và 99.97%, chứng minh rằng mô hình nhận diện loại tấn công này rất đáng tin cậy. Tương tự, với lớp DoS/DDoS, các chỉ số cũng đạt mức tốt: 99.85%, 0.99.92% và 99.89%, thể hiện mô hình có hiệu suất xuất sắc trong việc phân loại các cuộc tấn công từ chối dịch vụ và phân tán.

Tuy nhiên, với lớp OtherAttacks, mặc dù precision là 100%, recall chỉ đạt 66.67, dẫn đến F1-score là 80.00%. Điều này cho thấy mô hình vẫn bỏ sót một số mẫu thực sự thuộc lớp này. Việc bỏ sót này có thể là do số lượng mẫu trong lớp quá ít (chỉ 9 mẫu), dẫn đến khó khăn trong việc huấn luyện mô hình để nhận diện tất cả các trường hợp.

Với lớp PortScan, mô hình đạt precision và recall ở mức 98.94% và 99.27%, thể hiện khả năng phân loại rất tốt với ít lỗi xảy ra. Cuối cùng, lớp WebAttacks có precision là 99.05% và recall là 97.20%, cho thấy hiệu suất cao trong việc phát hiện các cuộc tấn công web, mặc dù vẫn có một số lỗi nhỏ trong dự đoán.

Nhìn chung, mô hình hoạt động rất hiệu quả trên hầu hết các lớp, đặc biệt là các lớp phổ biến như BENIGN, BruteForce, DoS/DDoS, và PortScan, nhưng cần cải thiện khả năng nhận diện các lớp ít dữ liệu như Bot và OtherAttacks để tăng cường hiệu suất tổng thể.

*So sánh với các nghiên cứu khác*

Bảng 6: So sánh giữa các nhóm nghiên cứu

|  |  |  |  |
| --- | --- | --- | --- |
| Tác giả | Mô Hình sử dụng | Bộ dữ liệu | Độ chính xác(%) |
| Shailendra Sahu, B. M. Mehtre | Decision Tree (J48) | Kyoto 2006+ dataset | 97.23 |
| Kajal Rai, M. Syamala Devi, Ajay Guleria | Decision Tree Split (DTS) algorithm | NSL\_KDD | 79.52 |
| Siva S. Sivatha Sindhu, S. Geetha, A. Kannan | Decision Stump, C4.5, Naive Baye’s Tree,  Random Forest, Random Tree and Representative Tree | KDD’99 CUP | 89.11 |
| Alikhanov Jumabe, , SeungSam Yan | CatBoost | CIC\_IDS\_2018 | 92.41 |
| Nhóm tác giả đề tài | Voting | CIC\_IDS\_2018 | 99.88 |

Từ bảng 6, có thể thấy rằng các nghiên cứu trước đó đã sử dụng nhiều bộ dữ liệu khác nhau để đánh giá hiệu quả của các mô hình phát hiện xâm nhập (IDS). Tuy nhiên, một điểm đáng chú ý là các bộ dữ liệu như Kyoto 2006+, NSL\_KDD, và KDD'99 CUP đã xuất hiện từ khá lâu và không còn phản ánh đầy đủ các dạng tấn công hiện đại trong mạng ngày nay. Những bộ dữ liệu này có hạn chế về số lượng mẫu, trường thông tin, và tính đại diện của các loại tấn công mới, dẫn đến độ chính xác của các mô hình trong những nghiên cứu này chỉ đạt ở mức trung bình, từ 79.52% đến 97.23%.

Trong khi đó, bộ dữ liệu CIC\_IDS\_2018 được sử dụng trong nghiên cứu của nhóm tác giả Nguyễn Thắng Lợi và cộng sự mang tính hiện đại hơn. Đây là một bộ dữ liệu lớn, mô phỏng các tình huống mạng thực tế với nhiều dạng tấn công mới, bao gồm cả tấn công từ chối dịch vụ (DoS/DDoS), brute force, và botnet. Điều này giúp đảm bảo rằng các mô hình học máy được huấn luyện và đánh giá trên bộ dữ liệu này sẽ có hiệu suất tốt hơn khi triển khai thực tế.

Mô hình voting được sử dụng trong nghiên cứu của nhóm tác giả kết hợp sức mạnh của ba thuật toán học máy phổ biến: Decision Tree, Random Forest và Extra Tree. Sự kết hợp này mang lại tính ổn định và khả năng tổng quát hóa tốt hơn, tận dụng ưu điểm của từng thuật toán. Kết quả đạt được độ chính xác 99.88%, vượt trội hơn so với tất cả các nghiên cứu trước đó trong bảng 6. Đây là minh chứng cho thấy sự hiệu quả của việc sử dụng mô hình voting cùng với một bộ dữ liệu hiện đại như CIC\_IDS\_2018.

Ngoài ra, việc so sánh giữa các mô hình như Decision Tree (J48), Decision Tree Split (DTS), và CatBoost cho thấy rằng các mô hình hiện đại hơn, hoặc các phương pháp kết hợp như voting, có thể tận dụng tốt hơn các đặc điểm phức tạp của dữ liệu. Độ chính xác của các nghiên cứu trước đó thường bị giới hạn bởi tính chất đơn giản hoặc cũ kỹ của bộ dữ liệu, trong khi nhóm nghiên cứu hiện tại đã khắc phục những hạn chế này.

## 5.2. Đánh giá hiệu suất hệ thống IDS

Hệ thống IDS khi sử dụng mô hình đạt độ chính xác 99.88% trên tập dữ liệu kiểm tra, cho thấy khả năng dự đoán và phát hiện các loại tấn công rất mạnh mẽ. Đặc biệt, các lớp phổ biến như BENIGN, BruteForce, DoS/DDoS và PortScan có precision và recall gần như hoàn hảo. Điều này đồng nghĩa với việc IDS sẽ rất hiệu quả trong việc phát hiện các cuộc tấn công thông dụng mà không bỏ sót nhiều sự kiện bất thường.

1. *Xử lý các mẫu không phải là tấn công*

Khả năng phân loại các mẫu thuộc lớp BENIGN với precision và recall đạt mức tốt tốt lần lượt là 99.93% và 99.92%, đây một ưu điểm lớn của hệ thống IDS này. Việc giảm thiểu các false positive trong các mẫu bình thường giúp IDS giảm thiểu báo động sai, hạn chế làm gián đoạn hệ thống hoặc ảnh hưởng đến việc giám sát an ninh.

1. *Phát hiện các cuộc tấn công ít dữ liệu*

Tuy mô hình hoạt động tốt ở các lớp có nhiều dữ liệu, hệ thống IDS vẫn gặp khó khăn với các lớp ít dữ liệu như Bot và OtherAttacks. Với lớp OtherAttacks, recall đạt 66.67%, cho thấy IDS có thể bỏ sót một số cuộc tấn công thực sự. Đây là một yếu tố quan trọng cần xem xét khi áp dụng IDS trong thực tế, đặc biệt nếu hệ thống cần bảo vệ khỏi nhiều loại tấn công đa dạng.

*c) Hiệu suất trong môi trường thực tế*

Với bộ dữ liệu CIC-IDS 2018 (900MB) và các chỉ số hiệu suất cao, hệ thống IDS có thể hoạt động ổn định khi xử lý lượng lớn dữ liệu. Tuy nhiên, để áp dụng trong môi trường thực tế, cần đánh giá thêm về khả năng xử lý trong thời gian thực và tối ưu hóa hệ thống để đáp ứng khối lượng dữ liệu lớn từ các mạng lớn hơn. Tốc độ và độ trễ của mô hình sẽ là các yếu tố quan trọng để đảm bảo IDS hoạt động hiệu quả và liên tục.

## 5.3. Ưu và nhược điểm của hệ thống:

### 5.3.1. Ưu điểm

Mô hình thể hiện độ chính xác cao với accuracy đạt mức 99.88%, minh chứng cho khả năng phân loại tổng thể rất tốt. Điều này cho thấy mô hình đã dự đoán đúng gần như toàn bộ dữ liệu kiểm tra. Hiệu suất của mô hình ở các lớp chính cũng rất ấn tượng, đặc biệt là các lớp như BENIGN, BruteForce, DoS/DDoS, và PortScan, với precision, recall, và F1-score gần như hoàn hảo (gần 100%). Điều này chứng tỏ mô hình có khả năng phân loại mạnh mẽ đối với các loại dữ liệu phổ biến hoặc các lớp có nhiều mẫu. Ngoài ra, khả năng phát hiện cao được thể hiện rõ qua các lớp tấn công phổ biến như BruteForce và DoS/DDoS, với F1-score đạt 99.97% và 99.89%, chứng minh rằng mô hình có thể phát hiện chính xác và đầy đủ các cuộc tấn công này. Cuối cùng, tỷ lệ báo động sai thấp nhờ độ chính xác và precision cao giúp giảm thiểu các cảnh báo sai, điều này cực kỳ quan trọng để giảm thiểu phiền toái cho người quản trị hệ thống.

### 5.3.2. Nhược điểm

Tuy nhiên, mô hình vẫn có hiệu suất thấp ở các lớp ít dữ liệu. Cụ thể, lớp Bot có precision và recall chỉ ở mức 87.94% và 76.86%, trong khi lớp OtherAttacks chỉ đạt recall 66.67%. Điều này cho thấy mô hình gặp khó khăn trong việc nhận diện các lớp ít xuất hiện, dẫn đến tỷ lệ false negative cao hơn, tức là bỏ sót một số mẫu tấn công thực sự. Ngoài ra, khả năng phân loại không đồng đều giữa các lớp là một nhược điểm đáng lưu ý. Dù mô hình hoạt động rất tốt với các lớp phổ biến, hiệu suất lại giảm đáng kể đối với các lớp ít dữ liệu như Bot và OtherAttacks, điều này có thể gây rủi ro khi bỏ sót các loại tấn công đặc biệt quan trọng nhưng ít phổ biến. Nguy cơ overfitting cũng là một yếu tố cần cân nhắc, vì độ chính xác tổng thể quá cao (99.88%) có thể ám chỉ rằng mô hình đã ghi nhớ dữ liệu huấn luyện thay vì học cách tổng quát hóa. Điều này cần được kiểm tra kỹ hơn để đảm bảo mô hình có thể hoạt động tốt trên các tập dữ liệu mới. Cuối cùng, sự bất cân bằng dữ liệu cũng là một thách thức, khi chỉ số macro avg recall chỉ đạt 0.94, cho thấy sự khác biệt hiệu suất giữa các lớp. Việc cải thiện có thể yêu cầu tăng cường dữ liệu hoặc sử dụng các kỹ thuật xử lý dữ liệu mất cân bằng.

**PHẦN 3: KẾT LUẬN**

## 1. Kết quả nghiên cứu của đề tài

Trong lĩnh vực an toàn và an ninh thông tin, hiện nay có rất nhiều nghiên cứu được thực hiện bởi các tổ chức, công ty, và trường đại học trong và ngoài nước. Các nghiên cứu này ngày càng tập trung vào việc ứng dụng học máy vào các giải pháp an ninh mạng, đặc biệt là trong các hệ thống phát hiện xâm nhập (IDS). Các mô hình phát hiện xâm nhập hiện tại bao gồm: phát hiện dựa trên dấu hiệu (signature-based detection), phát hiện dựa trên quá trình tự học (anomaly-based detection), và phát hiện kết hợp cả dấu hiệu lẫn quá trình tự học (hybrid detection). Mỗi mô hình đều có ưu điểm riêng và được lựa chọn tùy thuộc vào nhu cầu cụ thể của từng tổ chức.

Ứng dụng học máy vào công tác an ninh thông tin đang trở thành một xu thế tất yếu, với sự ra đời của hàng loạt các sản phẩm và công trình nghiên cứu ứng dụng công nghệ này. Các sản phẩm an ninh mạng của những công ty bảo mật lớn như Cisco, Juniper, và Trend Micro đều tích hợp khả năng phát hiện thông minh và có hiệu quả rất cao trong việc bảo vệ hệ thống. Tuy nhiên, những giải pháp này thường đi kèm với chi phí duy trì và cập nhật khá cao, đồng thời có xu hướng làm cho người giám sát trở nên phụ thuộc vào công cụ, từ đó giảm tính linh hoạt và tự chủ trong công tác bảo mật của tổ chức.

Với sản phẩm hoàn thiện từ nghiên cứu là hệ thống phát hiện xâm nhập IDS, nhóm nghiên cứu kỳ vọng rằng sản phẩm sẽ không chỉ được đưa vào thực nghiệm trên các mạng LAN tại Trường Đại học Sư phạm Kỹ thuật TP.HCM, mà còn trở thành một tài liệu tham khảo hữu ích cho sinh viên ngành Công nghệ Thông tin của trường. Việc triển khai và nghiên cứu sản phẩm trong môi trường thực tế sẽ giúp sinh viên hiểu sâu hơn về các nguyên lý và phương pháp phát hiện xâm nhập, đồng thời khuyến khích họ tìm hiểu và phát triển các giải pháp an ninh mạng ứng dụng học máy trong tương lai. Điều này không chỉ góp phần nâng cao chất lượng đào tạo mà còn đáp ứng nhu cầu ngày càng cao về chuyên môn trong lĩnh vực an ninh thông tin.

## 2. Hạn chế của sản phẩm

Mặc dù sản phẩm đã đạt được những kết quả nhất định, vẫn còn tồn tại một số hạn chế cần được khắc phục để nâng cao hiệu quả hoạt động. Đầu tiên, bộ dữ liệu được sử dụng để huấn luyện mô hình là một bộ dữ liệu cũ và chưa hoàn toàn phù hợp với các tình huống tấn công trong môi trường thực tế hiện nay. Điều này dẫn đến hiệu suất của hệ thống chưa đạt được mức tối ưu, đặc biệt khi đối mặt với các loại tấn công mới và phức tạp hơn.

Thứ hai, các mô hình học máy được sử dụng trong sản phẩm thuộc các thuật toán truyền thống, đã xuất hiện từ lâu. Trong khi đó, nhiều hệ thống phát hiện xâm nhập (IDS) tiên tiến hiện nay đã tích hợp các kỹ thuật học sâu (deep learning), giúp nâng cao hiệu suất và độ chính xác trong việc phát hiện các hành vi xâm nhập. Học sâu có khả năng xử lý và nhận diện các mẫu phức tạp hơn, từ đó cải thiện độ nhạy của hệ thống đối với các mối đe dọa mới nổi.

Do đó, để nâng cao hiệu quả của hệ thống, cần có sự cập nhật bộ dữ liệu huấn luyện sao cho sát với các mối đe dọa hiện tại, đồng thời nghiên cứu áp dụng các mô hình học sâu hiện đại. Điều này sẽ giúp hệ thống phát hiện xâm nhập đạt được độ chính xác và hiệu suất cao hơn, đáp ứng tốt hơn các yêu cầu của an ninh mạng trong môi trường ngày càng phức tạp.

## 3. Hướng phát triển

Tối ưu hóa thời gian xử lí các đặc trưng của tấn công, thay đổi mô hình học máy để gia tăng hiệu suất và chính xác. Cập nhật bộ dữ liệu từ các hệ thống IDS hiện nay đang hoạt động để có kết quả khách quan nhất về mức độ hiệu quả.

# **TÀI LIỆU THAM KHẢO**

[1]: Michael Nieles; Kelley Dempsey; Victoria Yan Pillitteri, “An Introduction to Computer Security: the NIST Handbook”, [Trực tuyến], Đường dẫn:<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-12.pdf>

[2]: Bùi Điền Phong, “XÂY DỰNG CÔNG CỤ PHÁT HIỆN XÂM NHẬP MẠNG MÁY TÍNH”, Ngày 15 tháng 7 năm 2022

[3]: ITU, “What is Signature-Based Detection?”, [Trực tuyến], Đường dẫn:<https://www.ituonline.com/tech-definitions/what-is-signature-based-detection/>

[12] Nguyễn Ngọc Điệp, Nguyễn Thị Thanh Thủy, (2020), Nâng Cao Khả Năng Phát Hiện Xâm Nhập Mạng Sử Dụng Mạng CNN,tạp chí khoa học công nghệ thông tin và truyền thông, Học Viện Công Nghệ Bưu Chính Viễn Thông.

[13] Thạc sĩ Kiều Công Minh, (2023), Ứng Dụng Representation Learning Phát Hiện Tấn Công Botnet, Học Viện Công Nghệ Bưu Chính Viễn Thông.

[14] Thạc sĩ Nguyễn Đức Cường, (2008), Warning and Protection System of Network Attack, Đại Học Bách Khoa Hà Nội

[15] T. Saranya; S. Sridevi; C. Deisy; M. K. A. Ahamed Khan, (2018), “Performance Analysis of Machine Learning Algorithms in Intrusion Detection System: A Review”, 2020, đường dẫn: [https://www.sciencedirect.com/science/article/pii/S187705092031111](https://www.sciencedirect.com/science/article/pii/S1877050920311121)

[16] Majjed Al-Qatf; Yu Lasheng; Mohammed Al-Habib; Kamal Al-Sabahi, Deep Learning Approach Combining Sparse Autoencoder With SVM for Network Intrusion Detection, đường dẫn: [https://ieeexplore. ieee. org/document/8463474](https://ieeexplore.ieee.org/document/8463474)

[17] Nathan Shone; Tran Nguyen Ngoc; Vu Dinh Phai; Qi Shi, (2018), A Deep Learning Approach to Network Intrusion

Detection, đường dẫn: https://ieeexplore. ieee. org/document/8264962

[18] Majjed Al-Qatf; Yu Lasheng; Mohammed Al-Habib; Kamal Al-Sabahi, (2018) Deep Learning Approach Combining Sparse Autoencoder With SVM for Network Intrusion Detection, đường dẫn: [https://ieeexplore. ieee. org/document/8463474](https://ieeexplore.ieee.org/document/8463474)

[19] F. Sabahi; A. Movaghar, (2005), Intrusion Detection: A Survey, đường dẫn:<https://www.researchgate.net/publication/226650646_Intrusion_Detection_A_Survey>

[20] Resmi AMDr. R Manicka chezian, (2017), Intrusion Detection System Techniques and Tools: A Survey,<https://saspublishers.com/media/articles/SJET53122-130.pdf>

[21] Ansam Khraisat; Ammar Alazab , (2021), A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges, <https://www.researchgate.net/publication/349898203_A_critical_review_of_intrusion_detection_systems_in_the_internet_of_things_techniques_deployment_strategy_validation_strategy_attacks_public_datasets_and_challenges>

[22] Susmit\_sekhar\_bhakta,"geeksforgeeks,", (2024). [trực tuyến]. Đường dẫn: <https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/>. [truy cập 10 10 2024].

[23] Aswathisasidharan, "geeksforgeeks,", (2024). [trực tuyến]. Đường dẫn: https://www.geeksforgeeks.org/support-vector-machine-algorithm/. [truy cập 30 10 2024].

[24] GeeksforGeeks, "GeeksforGeeks,", (2024). [Trực tuyến]. Đường dẫn: https://www.geeksforgeeks.org/k-means-clustering-introduction/. [truy cập 10 30 2024].

[25] Roheen Qamar, Baqar Ali Zardari, "Artificial Neural Networks: An Overview," researchgate, Iraq, 2023.

[26] A. Jain, "medium,", (2024). [trực tuyến]. Đường dẫn: https://medium.com/@abhishekjainindore24/different-types-of-ensemble-techniques-bagging-boosting-stacking-voting-blending-b04355a03c93.

[27] Communications Security Establishment (CSE) & the Canadian Institute for Cybersecurity (CIC), “University of New Brunswick”, (2018). Đường dẫn: <https://www.unb.ca/cic/datasets/ids-2018.html>

[28] Arash Habibi Lashkari & Gerard Drapper & Muhammad Saiful Islam, “Github”, (2016). Đường dẫn: <https://github.com/ahlashkari/CICFlowMeter>

[29] Shailendra Sahu; B M Mehtre, (2015), “Network intrusion detection system using J48 Decision Tree”, đường dẫn: <https://www.researchgate.net/publication/308839211_Network_intrusion_detection_system_using_J48_Decision_Tree>

[30] Kajal Rai Saraswat; Mandalika Syamala Devi, (2016), “Decision Tree Based Algorithm for Intrusion Detection”, đường dẫn: <https://www.researchgate.net/publication/298175900_Decision_Tree_Based_Algorithm_for_Intrusion_Detection>

[31] Siva S. Sivatha Sindhu; S. Geetha, (2012), “Decision tree based light weight intrusion detection using a wrapper approach”, đường dẫn: <https://www.sciencedirect.com/science/article/abs/pii/S0957417411009080>

[32] Jumabek Alikhanov; SeungSam Yang, (2021), “CatBoost-Based Network Intrusion Detection on Imbalanced CIC-IDS-2018 Dataset”, đường dẫn: <https://www.researchgate.net/publication/357534483_CatBoost-Based_Network_Intrusion_Detection_on_Imbalanced_CIC-IDS-2018_Dataset>

[33] Amin Dastanpour; Raja Azlina Raja Mahmood, (2013), “Feature Selection Based on Genetic Algorithm and Support Vector Machine for Intrusion Detection System”, đường dẫn: <https://www.researchgate.net/publication/265208874_Feature_Selection_Based_on_Genetic_Algorithm_and_Support_Vector_Machine_for_Intrusion_Detection_System>

[34] Jayshree Jha; Leena Ragha, (2013), “Intrusion Detection System using Support Vector Machine”, đường dẫn: <https://www.ijais.org/proceedings/icwac/number3/495-1342/>

[35] Bhupendra Ingre; Anamika Yadav, (2015), “Performance analysis of NSL-KDD dataset using ANN”, đường dẫn: <https://www.researchgate.net/publication/309698316_Performance_analysis_of_NSL-KDD_dataset_using_ANN>

[36] Marwan Ali Albahar; Muhammad Binsawad, (2020), “Improving Intrusion Detection System using Artificial Neural Network”, đường dẫn: <https://thesai.org/Publications/ViewPaper?Volume=11&Issue=6&Code=IJACSA&SerialNo=70>

[37] L. P. Dias; J. J. F. Cerqueira, (2017), “Using artificial neural network in intrusion detection systems to computer networks”, đường dẫn: https://ieeexplore.ieee.org/document/8101615

# **PHỤ LỤC**

## 1. Mã nguồn dùng để tùy chỉnh dữ liệu gốc

import joblib

import numpy as np

import pandas as pd

''' Đọc dữ liệu từ tệp CSV và kết nối các tệp CSV '''

file\_path1 = /content/drive/MyDrive/MachineLearningCVE/Friday-WorkingHours-Afternoon-DDos.pcap\_ISCX.csv

file\_path2 = /content/drive/MyDrive/MachineLearningCVE/Friday-WorkingHours-Afternoon-PortScan.pcap\_ISCX.csv

file\_path5 = /content/drive/MyDrive/MachineLearningCVE/Thursday-WorkingHours-Afternoon-Infilteration.pcap\_ISCX.csv

file\_path6 = /content/drive/MyDrive/MachineLearningCVE/Thursday-WorkingHours-Morning-WebAttacks.pcap\_ISCX.csv

file\_path3 = /content/drive/MyDrive/MachineLearningCVE/Monday-WorkingHours.pcap\_ISCX.csv

file\_path4 = /content/drive/MyDrive/MachineLearningCVE/Friday-WorkingHours-Morning.pcap\_ISCX.csv

file\_path7 = /content/drive/MyDrive/MachineLearningCVE/Tuesday-WorkingHours.pcap\_ISCX.csv

file\_path8 = /content/drive/MyDrive/MachineLearningCVE/Wednesday-workingHours.pcap\_ISCX.csv

data1 = pd.read\_csv(file\_path1)

data2 = pd.read\_csv(file\_path2)

data5 = pd.read\_csv(file\_path5)

data6 = pd.read\_csv(file\_path6)

data3 = pd.read\_csv(file\_path3)

data4 = pd.read\_csv(file\_path4)

data7 = pd.read\_csv(file\_path7)

data8 = pd.read\_csv(file\_path8)

data1 = pd.concat([data1, data2, data3, data4, data5, data6, data7, data8])

print(Before data processing:)

print(data1.shape)

print(data1[' Label'].value\_counts())

# Re-sampling

data1\_BENIGN = data1[(data1[' Label'] == 'BENIGN')]

data1\_Botnet = data1[(data1[' Label'] == 'Bot')]

data1\_DoS = data1[data1[' Label'].isin(['DoS', 'DDoS', 'DoS GoldenEye', 'DoS Hulk', 'DoS Slowhttptest', 'DoS slowloris'])]

data1\_DoS[' Label'] = 'DoS/DDoS'

data1\_PortScan = data1[(data1[' Label'] == 'PortScan')]

data1\_PortScan[' Label'] = 'PortScan'

data1\_BruteForce = data1[data1[' Label'].isin(['SSH-Patator', 'FTP-Patator'])]

data1\_BruteForce[' Label'] = 'BruteForce'

data1\_WebAttacks = data1[data1[' Label'].isin(['Web Attack � Brute Force', 'Web Attack � Sql Injection', 'Web Attack � XSS'])]

data1\_WebAttacks[' Label'] = 'WebAttacks'

data1\_minor = data1[data1[' Label'].isin(['Infiltration', 'Heartbleed'])]

data1\_minor[' Label'] = 'OtherAttacks'

df = pd.concat([data1\_BENIGN, data1\_DoS, data1\_PortScan, data1\_BruteForce,

data1\_WebAttacks, data1\_Botnet, data1\_minor])

df = df.sort\_index()

print(After data processing:)

print(df.shape)

print(df[' Label'].value\_counts())

df.to\_csv('/content/drive/MyDrive/MachineLearningCVE/data\_processed.csv', index=False)

## 2. Mã nguồn tiền xử lý và chuẩn hóa dữ liệu

def load\_and\_clean\_data(file\_path):

if not os.path.exists(file\_path):

print(fFile not found: {file\_path})

return None, None

df = pd.read\_csv(file\_path)

# Loại bỏ các hàng có giá trị thiếu và trùng lặp

df = df.dropna().drop\_duplicates()

# Lấy danh sách các cột số

numeric\_features = df.select\_dtypes(include=[np.number]).columns.tolist()

# Loại bỏ các hàng có giá trị vô hạn và số quá lớn

df = df[~df[numeric\_features].applymap(np.isinf).any(axis=1)]

df = df[(df[numeric\_features] <= np.finfo(np.float64).max).all(axis=1)]

return df, numeric\_features

def preprocess\_and\_split\_data(df, numeric\_features):

scaler = StandardScaler()

df[numeric\_features] = scaler.fit\_transform(df[numeric\_features])

le = LabelEncoder()

df[' Label'] = le.fit\_transform(df[' Label'])

X = df.drop(columns=[' Label'])

y = df[' Label']

# Chia thành train, validation và test

X\_train\_full, X\_test, y\_train\_full, y\_test = train\_test\_split(X, y, test\_size=TEST\_SIZE, random\_state=RANDOM\_STATE, stratify=y)

X\_train, X\_val, y\_train, y\_val = train\_test\_split(X\_train\_full, y\_train\_full, test\_size=VALIDATION\_SIZE, random\_state=RANDOM\_STATE, stratify=y\_train\_full)

print(fOriginal training set class distribution: {y\_train.value\_counts()})

# Sử dụng SMOTE để cân bằng tập huấn luyện

smote = SMOTE(random\_state=RANDOM\_STATE, sampling\_strategy=SMOTE\_SAMPLING\_STRATEGY)

X\_train, y\_train = smote.fit\_resample(X\_train, y\_train)

print(fResampled training set class distribution: {y\_train.value\_counts()})

return X\_train, X\_val, X\_test, y\_train, y\_val, y\_test, scaler, le

## 3. Mã nguồn thiết kế mô hình học máy

# Hằng số

RANDOM\_STATE = 69

TEST\_SIZE = 0.2

VALIDATION\_SIZE = 0.1

SMOTE\_SAMPLING\_STRATEGY = {4: 1500}

N\_ESTIMATORS = 500

DATA\_FILE\_PATH = '/content/drive/MyDrive/MachineLearningCVE/data\_processed.csv'

SELECTED\_FEATURES\_PATH = 'selected\_features.txt'

def train\_model(X\_train, y\_train):

rf = RandomForestClassifier(random\_state=RANDOM\_STATE, oob\_score=True, n\_jobs=-1, class\_weight=balanced, n\_estimators=N\_ESTIMATORS)

dt = DecisionTreeClassifier(random\_state=RANDOM\_STATE, class\_weight=balanced)

et = ExtraTreesClassifier(random\_state=RANDOM\_STATE, n\_jobs=-1, class\_weight=balanced, n\_estimators=N\_ESTIMATORS)

voting\_model = VotingClassifier(estimators=[('rf', rf), ('dt', dt), ('et', et)], voting='hard', n\_jobs=-1)

voting\_model.fit(X\_train, y\_train)

return voting\_model

## 4. Mã nguồn huấn luyện và tinh chỉnh tham số mô hình

def evaluate\_model(model, X, y, le, dataset\_name=Test):

y\_pred = model.predict(X)

print(fConfusion Matrix ({dataset\_name} set):)

print(confusion\_matrix(y, y\_pred))

print(fClassification Report ({dataset\_name} set):)

print(classification\_report(y, y\_pred, target\_names=le.classes\_, digits=4))

def feature\_importance\_optimized(model, X\_train):

feature\_importances = np.mean([est.feature\_importances\_ for est in model.named\_estimators\_.values()], axis=0)

features\_df = pd.DataFrame({'feature': X\_train.columns, 'importance': feature\_importances})

features\_df.sort\_values(by='importance', ascending=False, inplace=True)

features\_df['cumulative\_importance'] = features\_df['importance'].cumsum()

selected\_features = features\_df[features\_df['cumulative\_importance'] <= 0.9]['feature'].tolist()

print(fTotal number of selected features: {len(selected\_features)})

with open(SELECTED\_FEATURES\_PATH, 'w') as f:

for feature in selected\_features:

f.write(f{feature}\n)

return selected\_features

def save\_models(model, scaler, le):

joblib.dump(model, 'voting\_model.joblib')

joblib.dump(scaler, 'scaler.joblib')

joblib.dump(le, 'label\_encoder.joblib')

def main():

df, numeric\_features = load\_and\_clean\_data(DATA\_FILE\_PATH)

if df is not None:

if not os.path.exists(SELECTED\_FEATURES\_PATH) or open(SELECTED\_FEATURES\_PATH, 'r').read().strip() == '':

First training with all features

X\_train, X\_val, X\_test, y\_train, y\_val, y\_test, scaler, le = preprocess\_and\_split\_data(df, numeric\_features)

voting\_model = train\_model(X\_train, y\_train)

Evaluate on validation set

evaluate\_model(voting\_model, X\_val, y\_val, le, dataset\_name=Validation)

Select features

selected\_features = feature\_importance\_optimized(voting\_model, X\_train)

else:

Training with selected features

with open(SELECTED\_FEATURES\_PATH, 'r') as f:

selected\_features = f.read().splitlines()

X\_train, X\_val, X\_test, y\_train, y\_val, y\_test, scaler, le = preprocess\_and\_split\_data(df[selected\_features + [' Label']], selected\_features)

voting\_model = train\_model(X\_train, y\_train)

Evaluate on validation set

evaluate\_model(voting\_model, X\_val, y\_val, le, dataset\_name=Validation)

Evaluate on test set after optimization

evaluate\_model(voting\_model, X\_test, y\_test, le, dataset\_name=Test)

save\_models(voting\_model, scaler, le)

## 5. Mã nguồn Data Gathering

from collections import defaultdict

import numpy as np

import pandas as pd

from scapy.all import sniff

from scapy.layers.inet import IP, TCP, UDP

# Danh sách các thuộc tính cần tính toán

columns = [

' Destination Port', ' Init\_Win\_bytes\_backward', ' Average Packet Size',

' Bwd Packet Length Std', 'Init\_Win\_bytes\_forward', 'Flow Bytes/s',

' PSH Flag Count', 'Fwd IAT Total', 'Bwd IAT Total', ' min\_seg\_size\_forward',

'Total Length of Fwd Packets', ' Flow Duration', ' Packet Length Mean',

' Avg Bwd Segment Size', ' Bwd Packet Length Mean', ' Subflow Fwd Bytes',

' Fwd Packet Length Max', ' Fwd IAT Max', ' Flow IAT Max', ' ACK Flag Count',

' Bwd Packet Length Min', ' Max Packet Length', ' Fwd Packet Length Mean',

' Fwd IAT Std', ' Bwd IAT Min', ' Bwd Header Length', ' Total Backward Packets',

'Bwd Packet Length Max', ' Packet Length Std', ' Total Fwd Packets',

' Subflow Bwd Packets', ' Min Packet Length', ' Subflow Bwd Bytes',

' Packet Length Variance', ' Fwd IAT Mean', ' act\_data\_pkt\_fwd',

' URG Flag Count', 'Fwd PSH Flags', ' Flow IAT Std', ' Fwd Header Length.1',

' SYN Flag Count', ' Bwd Packets/s'

]

def capture\_packets(interface='Wi-Fi', duration=60):

packets = sniff(iface=interface, timeout=duration, prn=lambda x: x.summary())

return packets

def extract\_flow\_key(packet):

src = packet[IP].src

dst = packet[IP].dst

sport = packet[TCP].sport if TCP in packet else packet[UDP].sport

dport = packet[TCP].dport if TCP in packet else packet[UDP].dport

return (src, dst, sport, dport)

def calculate\_packet\_lengths(packets):

return [len(pkt) for pkt in packets]

def calculate\_iats(packets):

return [packets[i].time - packets[i - 1].time for i in range(1, len(packets))]

def process\_packets(packets):

flows = defaultdict(list)

for packet in packets:

if IP in packet and (TCP in packet or UDP in packet):

flow\_key = extract\_flow\_key(packet)

flows[flow\_key].append(packet)

data = []

for flow\_key, flow\_packets in flows.items():

if not flow\_packets:

continue

try:

packet\_lengths = calculate\_packet\_lengths(flow\_packets)

iats = calculate\_iats(flow\_packets)

flow\_duration = flow\_packets[-1].time - flow\_packets[0].time if flow\_packets else 0

packet\_data = {

' Destination Port': flow\_packets[0][TCP].dport if TCP in flow\_packets[0]

else (

flow\_packets[0][UDP].dport if UDP in flow\_packets[0] else 0),

' Init\_Win\_bytes\_backward': np.mean([pkt[TCP].window for pkt in flow\_packets if TCP in pkt and hasattr(pkt[TCP], 'window')]) if flow\_packets else 0,

' Average Packet Size': np.mean(packet\_lengths),

' Bwd Packet Length Std': np.std(packet\_lengths),

'Init\_Win\_bytes\_forward': np.mean([pkt[TCP].window for pkt in flow\_packets if TCP in pkt and hasattr(pkt[TCP], 'window')]) if flow\_packets else 0,

'Flow Bytes/s': sum(packet\_lengths) / flow\_duration if flow\_duration > 0 else 0,

' PSH Flag Count': sum([pkt[TCP].psh for pkt in flow\_packets if TCP in pkt and hasattr(pkt[TCP], 'psh')]),

'Fwd IAT Total': sum(iats),

'Bwd IAT Total': sum(iats),

' min\_seg\_size\_forward': min(packet\_lengths) if packet\_lengths else 0,

'Total Length of Fwd Packets': sum(packet\_lengths),

' Flow Duration': flow\_duration,

' Packet Length Mean': np.mean(packet\_lengths) if packet\_lengths else 0,

' Avg Bwd Segment Size': np.mean(packet\_lengths) if packet\_lengths else 0,

' Bwd Packet Length Mean': np.mean(packet\_lengths) if packet\_lengths else 0,

' Subflow Fwd Bytes': sum(packet\_lengths),

' Fwd Packet Length Max': max(packet\_lengths) if packet\_lengths else 0,

' Fwd IAT Max': max(iats) if iats else 0,

' Flow IAT Max': max(iats) if iats else 0,

' ACK Flag Count': sum([pkt[TCP].ack for pkt in flow\_packets if TCP in pkt and hasattr(pkt[TCP], 'ack')]),

' Bwd Packet Length Min': min(packet\_lengths) if packet\_lengths else 0,

' Max Packet Length': max(packet\_lengths) if packet\_lengths else 0,

' Fwd Packet Length Mean': np.mean(packet\_lengths) if packet\_lengths else 0,

' Fwd IAT Std': np.std(iats) if iats else 0,

' Bwd IAT Min': min(iats) if iats else 0,

' Bwd Header Length': np.mean([len(pkt[TCP]) for pkt in flow\_packets if TCP in pkt]),

' Total Backward Packets': len(flow\_packets),

'Bwd Packet Length Max': max(packet\_lengths) if packet\_lengths else 0,

' Packet Length Std': np.std(packet\_lengths) if packet\_lengths else 0,

' Total Fwd Packets': len(flow\_packets),

' Subflow Bwd Packets': len(flow\_packets),

' Min Packet Length': min(packet\_lengths) if packet\_lengths else 0,

' Subflow Bwd Bytes': sum(packet\_lengths),

' Packet Length Variance': np.var(packet\_lengths) if packet\_lengths else 0,

' Fwd IAT Mean': np.mean(iats) if iats else 0,

' act\_data\_pkt\_fwd': 0,

' URG Flag Count': sum([pkt[TCP].urg for pkt in flow\_packets if TCP in pkt and hasattr(pkt[TCP], 'urg')]),

'Fwd PSH Flags': sum([pkt[TCP].psh for pkt in flow\_packets if TCP in pkt and hasattr(pkt[TCP], 'psh')]),

' Flow IAT Std': np.std(iats) if iats else 0,

' Fwd Header Length.1': np.mean([len(pkt[TCP]) for pkt in flow\_packets if TCP in pkt]),

' SYN Flag Count': sum([pkt[TCP].syn for pkt in flow\_packets if TCP in pkt and hasattr(pkt[TCP], 'syn')]),

' Bwd Packets/s': len(flow\_packets) / flow\_duration if flow\_duration > 0 else 0

}

# Thay thế giá trị NaN và vô cùng bằng 0

for key, value in packet\_data.items():

if isinstance(value, float) and (np.isnan(value) or np.isinf(value)):

packet\_data[key] = 0

data.append(packet\_data)

except Exception as e:

print(f"Error processing flow {flow\_key}: {e}")

return pd.DataFrame(data, columns=columns)

def get\_packet\_data():

packets = capture\_packets('Wi-Fi', 60)

return process\_packets(packets)

## 6. Mã nguồn Backend API

from flask import Flask, request, jsonify

import joblib

import pandas as pd

import traceback

app = Flask(\_\_name\_\_)

MODEL\_PATH = 'voting\_model.joblib'

SCALER\_PATH = 'scaler.joblib'

LABEL\_ENCODER\_PATH = 'label\_encoder.joblib'

# Tải mô hình, scaler và label encoder

try:

model = joblib.load(MODEL\_PATH)

scaler = joblib.load(SCALER\_PATH)

label\_encoder = joblib.load(LABEL\_ENCODER\_PATH)

except Exception as e:

print(f"Error loading model, scaler, or label encoder: {e}")

traceback.print\_exc()

EXPECTED\_COLUMNS = [

' Destination Port', ' Init\_Win\_bytes\_backward', ' Average Packet Size',

' Bwd Packet Length Std', 'Init\_Win\_bytes\_forward', 'Flow Bytes/s',

' PSH Flag Count', 'Fwd IAT Total', 'Bwd IAT Total', ' min\_seg\_size\_forward',

'Total Length of Fwd Packets', ' Flow Duration', ' Packet Length Mean',

' Avg Bwd Segment Size', ' Bwd Packet Length Mean', ' Subflow Fwd Bytes',

' Fwd Packet Length Max', ' Fwd IAT Max', ' Flow IAT Max', ' ACK Flag Count',

' Bwd Packet Length Min', ' Max Packet Length', ' Fwd Packet Length Mean',

' Fwd IAT Std', ' Bwd IAT Min', ' Bwd Header Length', ' Total Backward Packets',

'Bwd Packet Length Max', ' Packet Length Std', ' Total Fwd Packets',

' Subflow Bwd Packets', ' Min Packet Length', ' Subflow Bwd Bytes',

' Packet Length Variance', ' Fwd IAT Mean', ' act\_data\_pkt\_fwd',

' URG Flag Count', 'Fwd PSH Flags', ' Flow IAT Std', ' Fwd Header Length.1',

' SYN Flag Count', ' Bwd Packets/s'

]

@app.route('/api/evaluate', methods=['POST'])

def evaluate():

try:

data = request.json

if not data:

return jsonify({"error": "No data provided"}), 400

df = pd.DataFrame(data)

missing\_columns = [col for col in EXPECTED\_COLUMNS if col not in df.columns]

if missing\_columns:

return jsonify({"error": f"Missing columns: {', '.join(missing\_columns)}"}), 400

df = df[EXPECTED\_COLUMNS]

df\_scaled = scaler.transform(df)

predictions = model.predict(df\_scaled)

prediction\_labels = label\_encoder.inverse\_transform(predictions)

return jsonify(prediction\_labels.tolist()), 200

except Exception as e:

print(f"Error during evaluation: {e}")

traceback.print\_exc()

return jsonify({"error": str(e)}), 500

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

## 7. Mã nguồn Giao diện Quản lý

import packet\_capture

import requests

import json

import time

import pandas as pd

def send\_data\_to\_server(data):

url = 'http://localhost:5000/api/evaluate'

headers = {'Content-Type': 'application/json'}

try:

# Ghi dữ liệu ra file JSON với timestamp

time\_stamp = time.strftime("%Y-%m-%d\_%H-%M-%S", time.localtime())

with open(f'data\_{time\_stamp}.json', 'w') as f:

json.dump(data, f)

# Gửi request tới server

response = requests.post(url, headers=headers, json=data)

response.raise\_for\_status()

return response.json()

except requests.exceptions.RequestException as e:

print(f"An error occurred while sending data to the server: {e}")

return None

def display\_results\_as\_table(results):

df = pd.DataFrame(results)

print(df.to\_string(index=False))

def main():

print("Welcome to the Intrusion Detection System (IDS)")

while True:

print("\n1. Start IDS")

print("2. Exit")

choice = input("Enter your choice: ").strip()

if choice == '1':

print("Starting packet capture...")

try:

packet\_data = packet\_capture.get\_packet\_data()

if packet\_data.empty:

print("No packets captured.")

continue

print("Processing captured data...")

data = packet\_data.to\_dict(orient='records')

result = send\_data\_to\_server(data)

if result:

print("IDS Detection Results:")

display\_results\_as\_table(result)

else:

print("Failed to get results from server.")

except Exception as e:

print(f"An error occurred during packet capture or processing: {e}")

elif choice == '2':

print("Exiting...")

break

else:

print("Invalid choice, please try again.")

if \_\_name\_\_ == '\_\_main\_\_':

main()