ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Võ Văn Hoàng

NGHIÊN CỨU NÂNG CAO HIỆU NĂNG PHÁT HIỆN XÂM NHẬP MẠNG THÔNG QUA TĂNG CƯỜNG CHẤT LƯỢNG TẬP DỮ LIỆU, SUY LUẬN KẾT HỢP SONG SONG VÀ CHIẾN LƯỢC LẤY MẪU

Chuyên ngành: Hệ thống thông tin
Mã: 9480104

TÓM TẮT LUẬN ÁN TIẾN SỸ
CHUYÊN NGÀNH HỆ THỐNG THÔNG TIN

Cán bộ hướng dẫn:
PGS.TS Nguyễn Ngọc Hóa
PGS.TS Nguyễn Ngọc Tú

Hà Nội - 2025

Công trình được hoàn thành tại: **Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội**.

Người hướng dẫn khoa học:

- Phó giáo sư, Tiến sĩ Nguyễn Ngọc Hóa

- Phó giáo sư, Tiến sĩ Nguyễn Ngọc Tú

Phản biện: **Phó giáo sư, Tiến sĩ Nguyễn Linh Giang**

Phản biện: **Phó giáo sư, Tiến sĩ Nguyễn Long Giang**

Phản biện: **Giáo sư, Tiến sĩ Nguyễn Hiếu Minh**

Luận án sẽ được bảo vệ trước Hội đồng cấp Đại học Quốc gia chấm luận án tiến sĩ họp tại Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội trước tháng 12 năm 2025.

NGHIÊN CỨU SINH **CÁN BỘ HƯỚNG DẪN**

Võ Văn Hoàng Nguyễn Ngọc Hoá Nguyễn Ngọc Tú

XÁC NHẬN CỦA ĐƠN VỊ ĐÀO TẠO

Có thể tìm hiểu luận án tại:

- Thư viện quốc gia Việt Nam.
- Trung tâm Thông tin - Thư viện, Đại học Quốc gia Hà Nội.

Giới thiệu

Động lực nghiên cứu

Các cuộc tấn công mạng ngày càng tinh vi, bộc lộ những hạn chế của các hệ thống dựa trên cơ sở dữ liệu có sẵn và tập luật truyền thống, vốn khó đối phó với tấn công khai thác zero-day, phần mềm độc hại đa hình và tỷ lệ cảnh báo giả cao. Trong khi đó, ML và DL mang lại những giải pháp thay thế mạnh mẽ, nhưng vẫn đối mặt với nhiều thách thức như dữ liệu nhiễu và mất cân bằng, dư thừa đặc trưng, cũng như khả năng diễn giải kém. Hơn nữa, nhiều mô hình chưa đáp ứng được yêu cầu suy luận thời gian thực và khả năng mở rộng trong thực tế. Để khắc phục những khoảng trống này, luận án đề xuất một lộ trình thống nhất gồm cân bằng dữ liệu, tinh chỉnh đặc trưng, tối ưu hóa mô hình và suy luận đa mô hình nhằm xây dựng hệ thống phát hiện xâm nhập và mã độc dựa trên AI chính xác, bền vững, dễ giải thích và có khả năng triển khai.

Thách thức nghiên cứu

Luận án này tập trung vào những thách thức chính sau:

1. Thách thức 1: Các bộ dữ liệu an ninh mạng thường mất cân bằng nghiêm trọng, khi phần lớn mẫu thuộc về lưu lượng hợp lệ hoặc một vài loại tấn công phổ biến, trong khi các mối đe dọa hiếm nhưng nguy hiểm lại bị thiếu mẫu đại diện. Điều này dẫn đến việc mô hình học bị thiên lệch và kém hiệu quả trong phát hiện các tấn công thiểu số.

2. Thách thức 2: Đạt được độ chính xác cao và tỷ lệ báo động giả thấp trong các hệ thống phát hiện xâm nhập dựa trên AI, đồng thời vẫn duy trì hiệu năng tổng thể và khả năng giải thích, là một thách thức nói chung đối với học máy.

3. Thách thức 3: Hệ thống phát hiện xâm nhập mạng phải xử lý khối lượng lớn lưu lượng mạng ở tốc độ đường truyền với độ trễ tối thiểu. Tuy nhiên, độ phức tạp tính toán của các mô hình học máy thường cản trở khả năng triển khai thời gian thực.

Mục tiêu nghiên cứu

- Mục tiêu 1: Tổng quan về các loại tấn công mạng. Nghiên cứu, đánh giá các phương pháp phát hiện xâm nhập và mã độc, phân tích ưu điểm và nhược điểm của từng phương pháp.

- Mục tiêu 2: Đề xuất phương pháp tăng cường chất lượng các mẫu tấn công thiểu số, lựa chọn các mẫu lớp đa số; xác định các đặc trưng quan trọng trong tập dữ liệu huấn luyện.

- Mục tiêu 3: Hướng tới việc tích hợp mạng nơ-ron với các mô hình boosting thông qua chiến lược soft voting và stacking nhằm tăng cường khả năng chống chịu tấn công mạng.

- Mục tiêu 4: Các hệ thống phát hiện dựa trên AI thường gặp vấn đề độ trễ suy luận và khả năng mở rộng hạn chế. Mục tiêu này nhằm thiết kế một kiến trúc phát hiện linh hoạt, thông lượng cao, hỗ trợ cơ chế cảm nhận luồng và suy luận song song.

Phạm vi nghiên cứu

Để đạt được các mục tiêu của luận án này, chúng tôi tập trung vào các lĩnh vực trọng tâm sau:

1. Nghiên cứu cấu trúc dữ liệu và tình trạng mất cân bằng lớp trong các bộ dữ liệu phát hiện xâm nhập, đồng thời khảo sát hiệu quả của các mô hình học máy và học sâu.
2. Tập trung nghiên cứu xây dựng các kiến trúc phát hiện nhẹ, thông lượng cao, phù hợp cho triển khai thời gian thực trong các mạng quy mô lớn.

Phương pháp nghiên cứu

Luận án này áp dụng một phương pháp nghiên cứu có hệ thống và phân tầng.

Đóng góp nghiên cứu

Các đóng góp chính của luận án bao gồm:

1. Chúng tôi đề xuất phương pháp tăng cường tập dữ liệu và tối ưu hóa tập đặc trưng. Cách tiếp cận này tích hợp việc sinh mẫu đối kháng để làm giàu cho các lớp thiểu số, đồng thời áp dụng kỹ thuật lọc để giữ lại những mẫu có ý nghĩa ngữ nghĩa từ lớp đa số.
2. Đề xuất phương pháp kết hợp mạng nơ-ron với các bộ phân loại boosting thông qua chiến lược soft voting và stacking. Để tận dụng ưu điểm bổ sung của học sâu và các mô hình boosting để nâng cao độ chính xác và tính kháng cự trong phát hiện xâm nhập mạng.
3. Chúng tôi thiết kế và triển khai NetIPS, một kiến trúc phát hiện và ngăn chặn xâm nhập nhẹ, hoạt động theo thời gian thực và được tối ưu cho các môi trường mạng quy mô lớn.

Cấu trúc luận án

Luận án này được cấu trúc thành bốn chương:

- **Chương 1:** Trình bày các kiến thức nền tảng về phát hiện xâm nhập và mã độc, tập trung vào các kỹ thuật học máy, học sâu và phương pháp ensemble.
- **Chương 2:** Đề xuất các phương pháp tăng cường dữ liệu cho học máy, với mục tiêu giải quyết sự mất cân bằng giữa các lớp thiểu số và đa số trong tập dữ liệu.
- **Chương 3:** Tập trung vào cải tiến mô hình học máy nhằm nâng cao hiệu năng. Chương này đề xuất kết hợp và tăng cường lẫn nhau giữa các loại mô hình khác nhau để nâng cao hiệu quả phát hiện xâm nhập và tăng cường khả năng kháng cự của hệ thống.
- **Chương 4:** Đề xuất cách tiếp cận triển khai thực tế hệ thống phát hiện xâm nhập trên các mạng quy mô lớn. Một quy trình toàn diện được giới thiệu, kết hợp giữa phân tích dựa trên chữ ký và hành vi, cùng với các chiến lược thực thi và lấy mẫu.

1 Cơ sở lý thuyết và tổng quan tài liệu

1.1 Khái niệm cơ bản

1.1.1 Hệ thống phát hiện xâm nhập

Hệ thống IDS đóng vai trò then chốt trong việc giám sát hoạt động mạng và máy chủ, trong đó NIDS phân tích luồng lưu lượng còn HIDS tập trung vào hành vi tại điểm cuối. Các phương pháp phát hiện bao gồm kỹ thuật dựa trên cơ sở dữ liệu mẫu, chỉ hiệu quả với các mối đe dọa đã biết, và phương pháp dựa trên bất thường, có khả năng nhận diện các xâm nhập mới nhưng thường tạo ra nhiều cảnh báo sai.

1.1.2 Các loại hình tấn công mạng phổ biến

Các loại tấn công mạng phổ biến được tóm tắt thông qua Table 1.1.

Bảng 1.1: Các loại tấn công mạng phổ biến

Attack Type	**Technique**	**Impact**	**Detection**
Denial-of-Service (DoS/DDoS)	Traffic floods, amplification	Service unavailability	Rate limiting, filtering
Scanning & Enumeration	Port/vulnerability scans	Reconnaissance	IDS, anomaly detection
Spoofing	IP/ARP/DNS falsification	Evasion, redirection	Authentication, ARP/DNS security
Man-in-the-Middle (MitM)	Interception, SSL stripping	Data theft, manipulation	Encryption, certificate pinning
Sniffing/ Eavesdropping	Passive/active traffic capture	Credential leakage	TLS, VPN
Replay/Session Hijacking	Packet replay, session ID theft	Unauthorized access	Token/session management, TLS
Malware Propagation	Worms, trojans, ransomware	Compromise, data loss	Antivirus, sandboxing
Phishing/Social Engineering	Deceptive messages, psychological tricks	Credential theft, initial access	User training, email filtering
SQLi/XSS/CSRF	Web input manipulation	Data theft, defacement	Input validation, WAF
APT	Multi-stage, stealthy infiltration	Espionage, long-term theft	Behavior analytics, EDR
Supply Chain	Third-party compromise	Widespread breach	Vendor management, code review
Insider Threat	Privileged misuse, data exfiltration	Confidentiality breach	Monitoring, least privilege, DLP

1.1.3 Học máy trong phát hiện tấn công mạng

ML đã trở thành một công cụ then chốt trong an ninh mạng hiện đại nhờ khả năng học các mẫu phức tạp và thích ứng với các mối đe dọa ngày càng tinh vi, vượt qua những hạn chế của phương pháp phát hiện dựa trên luật truyền thống. Các kỹ thuật ML được ứng dụng trong nhiều tác vụ như phát hiện xâm nhập, phân loại mã độc, phát hiện lừa đảo trực tuyến...

1.1.4 Mất cân bằng lớp trong dữ liệu an ninh mạng

Mất cân bằng lớp là một thách thức lớn trong các bộ dữ liệu an ninh mạng, khi số lượng mẫu bình thường vượt trội so với mẫu độc hại, và các loại tấn công hiếm nhưng quan trọng thường bị biểu diễn thiếu. Điều này làm hạn chế khả năng, hiệu năng của mô hình học máy.

1.1.5 Học kết hợp trong phát hiện xâm nhập

Học tập tổ hợp kết hợp nhiều mô hình cơ sở để đạt hiệu năng dự đoán tốt hơn, khiến nó hiệu quả trong an ninh mạng – nơi các mẫu tấn công rất đa dạng và không ngừng biến đổi. Bằng cách tích hợp các mô hình tổ hợp giúp cải thiện độ chính xác, khả năng khái quát hóa và khả năng chống chịu trước các kỹ thuật né tránh đối kháng.

1.2 Phương pháp tiếp cận trong phát hiện mối đe doạ

1.2.1 Phát hiện xâm nhập dựa trên AI

Phát hiện xâm nhập dựa trên AI tận dụng các mô hình học máy (ML) và học sâu (DL) để phân loại các luồng lưu lượng mạng thành lành tính hoặc độc hại. Các phương pháp tăng cường dần (gradient boosting) đã chứng minh hiệu quả trong lĩnh vực này nhờ khả năng giảm dần lỗi dự đoán và mô hình hóa các mẫu tấn công phức tạp. Mạng nơ-ron sâu (DNN) vượt trội trong việc nắm bắt các quan hệ phi tuyến tính trong dữ liệu lưu lượng thông qua các biểu diễn nhiều tầng. Mỗi mô hình ML/DL đều có những thế mạnh riêng; việc kết hợp chúng thông qua học tập tổ hợp giúp cải thiện độ chính xác phát hiện và khả năng chống chịu trước các cuộc tấn công đối kháng.

1.2.2 Phát hiện mã độc dựa trên AI

Trong bối cảnh phát hiện mã độc bằng AI, bài toán có thể được mô tả như sau: cho một tập dữ liệu D gồm các cặp (v, y), trong đó v là vector đặc trưng trích xuất từ tệp PE và $y \in 0, 1$ là nhãn tương ứng (0 đối với tệp lành tính, 1 đối với tệp mã độc). Nhiệm vụ đặt ra là xây dựng một mô hình học máy tổng quát $f : \mathbb{R}^n \to 0, 1$ có khả năng ánh xạ bất kỳ vector đặc trưng v của một tệp PE mới thành nhãn dự đoán $\hat{y} = f(v)$. Mục tiêu là tối đa hóa độ chính xác của mô hình đồng thời duy trì khả năng khái quát hóa vượt ngoài dữ liệu huấn luyện, đảm bảo hệ thống có thể phát hiện mã độc một cách tin cậy trên các tệp chưa từng thấy.

1.2.3 Xử lý mất cân bằng dữ liệu

Hầu hết các bộ dữ liệu đều gặp tình trạng mất cân bằng lớp nghiêm trọng, khi lưu lượng hợp lệ chiếm ưu thế trong khi các luồng tấn công lại bị biểu diễn thiếu. Sự mất cân bằng này làm suy giảm chất lượng huấn luyện mô hình cũng như độ chính xác dự đoán. Để khắc phục, các kỹ thuật cân bằng dữ liệu thường được áp dụng, chẳng hạn như undersampling đối với lớp chiếm đa số và oversampling cho lớp thiểu số.

1.3 Công trình liên quan

1.3.1 Học máy trong phát hiện xâm nhập

Bảng tóm tắt về các phương pháp này và hiệu năng của chúng được trình bày trong Table 1.2.

1.3.2 Học máy trong phát hiện mã độc

Bảng tóm tắt về các phương pháp này và hiệu năng của chúng được trình bày trong Table 1.3.

1.3.3 Tăng cường dữ liệu

Để giải quyết vấn đề mất cân bằng dữ liệu, một số kỹ thuật thường được sử dụng như giảm mẫu ở lớp chiếm đa số (undersampling), tăng mẫu ở lớp thiểu số (oversampling), v.v.

1.4 Thu thập dữ liệu

Luận án này sử dụng một số bộ dữ liệu công khai phổ biến, bao gồm: CSE-CIC-IDS2018, NSL-KDD, EMBER2017, EMBER2018 và BODMAS.

Bảng 1.2: Tóm tắt các công trình liên quan đến phát hiện xâm nhập mạng

Method	Venue	Approach	Dataset	Acc(%)
RF+ miniVGGNet [**00004**]	IEEE Access 2020	Kết hợp K-Means và ENN để cân bằng tập dữ liệu, sau đó sử dụng RF + miniVGGNet để phát hiện xâm nhập.	NSL-KDD, CIC-IDS2018	82.84, 96.99
WGAN+ LightGBM [**Jurnal_Zang_2021**]	Computer Science 2021	Áp dụng WGAN-GP để sinh dữ liệu cho các mẫu thuộc lớp thiểu số và sử dụng LightGBM cho tác vụ phân loại.	NSL-KDD, CIC-IDS2018	99.00, 96.00
MMM-RF [**jMMM-RF22**]	Computer & Security 2022	Sử dụng CFS để phân tích lưu lượng mạng, T-SNE để giảm chiều dữ liệu, và SMOTE để xử lý mất cân bằng trên tập dữ liệu CSE-CIC-IDS2018.	CIC-IDS2018	99.98
CNN, DBNs, LSTM [**J_Moha_2022**]	Computers and Electrical Engineering 2022	Chuyển đổi các đặc trưng luồng lưu lượng thành dạng sóng và sử dụng các kỹ thuật học sâu tiên tiến trong nhận dạng âm thanh/giọng nói để phát hiện xâm nhập.	CIC-IDS2017, NSL-KDD	99.21, 84.82
CNN+LSTM [**J_Farhan_2023**]	Digital Communications and Networks 2023	Sử dụng SMOTE để cân bằng lưu lượng bất thường, CNN để trích xuất đặc trưng sâu, sau đó áp dụng CNN-LSTM để phát hiện xâm nhập.	UNSW.NB15, CIC-IDS2017, NSL-KDD	99.21, 99.32, 98.45
FFO+PNN [**J_Alexandria_2023**]	Alexandria Engineering Journal 2023	Sử dụng kỹ thuật FFO để trích xuất đặc trưng và PNN để phân loại các lớp.	NSL-KDD	98.99
CNN+EQL [**J_Keyan_REN_2023**]	Computer Communications 2023	Sử dụng CNN kết hợp với cơ chế Attention để tạo thành khối CA (CA Block) nhằm tập trung vào việc trích xuất đặc trưng không gian–thời gian cục bộ, đồng thời áp dụng EQL v2 để tăng trọng số cho lớp thiểu số và cân bằng sự chú ý của mô hình học đối với các lớp này.	UNSW.NB15, NSL-KDD, CIC-IDS2017, CIC-DDoS2019	89.39, 99.77, 99.88, 99.58
PIGNUS [**jPIGMIS23**]	Computer & Security 2023	Sử dụng Auto-Encoders để chọn ra các đặc trưng tối ưu và Cascade Forward Back Propagation Neural Network (CFBPNN) cho tác vụ phân loại và phát hiện tấn công.	NSL-KDD	99.02

1.5 Thước đo đánh giá

Chúng tôi sử dụng các chỉ số chuẩn được tính toán từ ma trận nhầm lẫn, chẳng hạn như: *Acc* (Accuracy), *Prec* (Precision), *Rec* (Recall), v.v.

1.6 Khoảng trông nghiên cứu và hướng tiếp cận

- **Khoảng trống nghiên cứu 1:** Hầu hết các bộ dữ liệu phát hiện xâm nhập trong thực tế đều gặp vấn đề mất cân bằng nghiêm trọng, khi các lớp tấn công thiểu số bị thiếu hụt và khó học được hiệu quả.

  Hướng tiếp cận (chapter 2): Để khắc phục hạn chế này, chúng tôi đề xuất các phương pháp tăng cường dữ liệu nhằm cải thiện cả số lượng và chất lượng của tập huấn luyện, đồng thời tối ưu hóa không gian đặc trưng để tăng khả năng học.

- **Khoảng trống nghiên cứu 2:** Mặc dù đã có nhiều nghiên cứu tối ưu hóa mô hình máy học cho phát hiện xâm nhập, việc đạt được độ chính xác cao, tính ổn định và khả năng chống lại tấn công đối kháng vẫn là một thách thức dai dẳng.

  Hướng tiếp cận (chapter 3): Chúng tôi thiết kế một pipeline suy luận kết hợp mutual deep+boosting,

Bảng 1.3: Tóm tắt các công trình liên quan đến phát hiện mã độc

Method	Venue	Approach	Dataset	Acc(%)
CNN [**c_Marais_2021**]	Distributed Computing and Artificial Intelligence 2021	Phương pháp trong nghiên cứu này chuyển đổi các tệp nhị phân thành ảnh xám để phát hiện mã độc. Mô hình cũng tích hợp cơ chế Attention để xác định các phần đáng ngờ trong tệp.	EMBER 2018	94.00
DNN [**j_DIVAKARLA_2022**]	Procedia Computer Science 2022	Phương pháp này xây dựng một mô hình sinh tấn công cải tiến dựa trên GAN nhằm tăng cường hệ thống hiện tại vốn dựa trên DNN.	EMBER 2018	97.42
CNN [**j_lad_2022**]	International Journal of Computer Network and Information Security 2022	Phương pháp này sử dụng các kỹ thuật trích xuất đặc trưng, chuẩn hóa dữ liệu và làm sạch dữ liệu để xử lý sự mất cân bằng và tạp chất trong tập dữ liệu.	EMBER 2017 & 2018	97.53, 94.09
EII-MBS [**jEII-MBS2022**]	Computers & Security 2022	Kỹ thuật này tìm ra các mẫu trong cách các lệnh liên quan đến nhau và chuyển đổi thông tin đó thành các biểu diễn vector để phân loại các họ mã độc.	BODMAS	99.29
XGB-CATB-EXT [**j_Murat.2023**]	Computer, Material & Continua 2023	Kỹ thuật trong nghiên cứu này sử dụng mô hình kết hợp giữa học có giám sát và không giám sát để cải thiện khả năng phát hiện mã độc. Cụ thể, k-means được dùng để phân cụm dữ liệu trước khi một tập các thuật toán ML tiến hành phân loại.	EMBER 2018	96.77
MD-ADA [**j_MD-ADA_2024**]	Computers & Security 2024	Cách tiếp cận này kết hợp biểu diễn ảnh dựa trên CNN và thích ứng miền đối kháng (sử dụng GANs) để phân loại mã độc.	BODMAS	99.29
FCG-MFD [**jFCG-MFD2022**]	Journal of Network and Computer Applications 2025	Phương pháp này sử dụng đồ thị lời gọi hàm (function call graphs) và node2vec kết hợp với các ý tưởng từ xử lý ngôn ngữ tự nhiên (NLP) để hỗ trợ phân loại các họ mã độc.	BODMAS	99.28

khai thác sức mạnh bổ sung của các mô hình khác nhau nhằm nâng cao hiệu năng tổng thể và giảm thiểu các điểm yếu như poisoning vào mô hình.

- **Khoảng trống nghiên cứu 3:** Bất chấp những tiến bộ gần đây, phần lớn các khung IDS hiện tại vẫn chưa phù hợp với môi trường thông lượng cao do các nút thắt tính toán, logic phát hiện tĩnh, và thiếu cơ chế kiểm soát luồng thích ứng. Các hệ thống hiện nay thường không đáp ứng được ràng buộc về độ trễ thời gian thực hoặc khả năng mở rộng trong các mạng doanh nghiệp và ISP.

  Hướng tiếp cận (chapter 4): Chúng tôi đề xuất một hệ thống phòng chống xâm nhập có khả năng mở rộng và độ trễ thấp, gọi là **NetIPS**, được xây dựng dựa trên mô hình deep và boosting song song, tích hợp chiến lược cảm biến luồng và phân tích sandbox.

1.7 Tóm tắt chương

Tóm lại, chương này đã xác định các thách thức nghiên cứu và mục tiêu chính trong phát hiện xâm nhập và mã độc, đồng thời phác thảo những đóng góp khoa học chủ yếu cùng lộ trình nghiên cứu của luận án. Sự đối sánh giữa các đóng góp này với các chương kỹ thuật tương ứng cũng đã được trình bày, qua đó cung cấp một cấu trúc rõ ràng cho các phần tiếp theo của công trình.

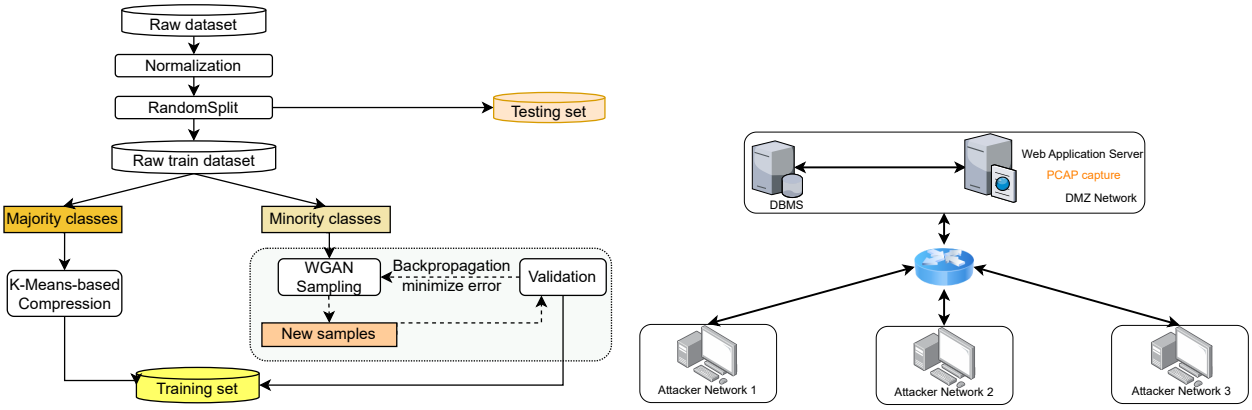
2 Tăng cường khả năng phát hiện xâm nhập dựa trên AI bằng cách bổ sung dữ liệu và tối ưu hóa đặc trưng

2.1 Phát biểu bài toán

Chúng tôi đề xuất hai giải pháp: (i) tăng cường dữ liệu huấn luyện, nén các lớp đa số và tạo ra các mẫu thiểu số thực tế nhằm cải thiện sự cân bằng và đa dạng của tập dữ liệu; và (ii) tối ưu hóa tập thuộc tính dựa trên SHAP (OFS) giúp loại bỏ các thuộc tính không liên quan, tăng cường khả năng diễn giải và giảm thời gian tính toán.

2.2 Hướng tiếp cận

Phương pháp tiếp cận được đề xuất được thiết kế để đồng thời khắc phục vấn đề thiếu dữ liệu ở các lớp thiểu số, lựa chọn các mẫu chất lượng cao từ các lớp đa số, và xác định các đặc trưng có giá trị trong những tập dữ liệu có số lượng thuộc tính lớn - vấn đề thường gặp trong tập dữ liệu.



(a) Kiến trúc tăng cường dữ liệu dựa trên AWGAN (b) Kiến trúc Testbed sinh tấn công SQL-Injection

Hình 2.1: Kiến trúc AWGAN và Testbed

2.3 Tăng cường tập dữ liệu huấn luyện

2.3.1 Tăng cường tập dữ liệu huấn luyện thông qua nén và phóng đại

Chúng tôi đề xuất một phương pháp dựa trên khái niệm của thuật toán DSSTE được giới thiệu bởi [**00004**] nhằm tăng cường tập dữ liệu huấn luyện. Thuật toán của chúng tôi có tên là AugDS và được minh họa trong thuật toán 2.1.

2.3.2 Tăng cường tập dữ liệu huấn luyện thông qua AWGAN

Phương pháp AWGAN tạo ra các mẫu thực tế cho các lớp thiểu số bằng cách sử dụng WGAN. AWGAN được minh họa trong Figure 2.1a và được mô tả chi tiết trong thuật toán 2.2.

2.4 Tối ưu hoá đặc trưng

2.4.1 Trích xuất và làm sạch đặc trưng

Trích xuất và làm sạch đặc trưng là bước thiết yếu nhằm giảm chi phí tính toán và tránh dữ liệu nhiễu hoặc trùng lặp gây ra hiện tượng overfitting. Cách tiếp cận của chúng tôi loại bỏ các bản ghi null hoặc trùng lặp, đảm bảo chỉ giữ lại những mẫu duy nhất và có liên quan trong tập dữ liệu.

Algorithm 2.1 AugDS: Build the Augmented Dataset

Input: F - Raw Dataset, represented by a list of feature vectors; K - scaling factor
Output: T - Augmented Dataset;

$L \leftarrow ComputeLabels(F)$ ▷ Get all labels of dataset F
$F \leftarrow Normalize(F)$ ▷ normalizing all feature vectors
$ES = EditedNearestNeighbours(RT, |L|)$ ▷ determining the easy sets ES by finding L nearest neighbours samples
$DS = RT \setminus ES$ ▷ difficult set DS is the rest of RT
$Majors, Minors \leftarrow ComputeMajMin(DS)$
$S_{maj} \leftarrow \emptyset, S_{min} \leftarrow \emptyset$
for each $M \in Majors$ **do** ▷ Compression Step
  $C \leftarrow Clustering(M, |L|)$ ▷ computing the centroids C of $|L|$ clusters by using KMeans algorithm
  $M \leftarrow Compress(M, C, \tau)$ ▷ compressing majority samples using centroids C of L clusters
  $S_{maj} \leftarrow S_{maj} \cup M$
end for
for each $M \in Minors$ **do** ▷ Zooming Step
  for each $m \in range(K, K + \frac{number}{N_{S_{min}}})$ **do** ▷ Zooming Step, $N_{S_{min}}$ is number sample in S_{min}.
    $M \leftarrow Zoom(m)$ ▷ zoom range is $[1 - \frac{1}{K}, 1 + \frac{1}{K}]$ on both continuous and categorical features.
    $S_{min} \leftarrow S_{min} \cup M$
  end for
end for
$T = ES \cup S_{maj} \cup S_{min}$ ▷ synthese of new dataset T
return (T)

Algorithm 2.2 AWGAN: Create the Training & Testing Sets by Augmented WGAN

Input: F - Raw Dataset, represented by a list of feature vectors.
r - ratio between training and testing sets; default is 7:3.
τ - maximum samples in a label.
Output: T - Training Set; V - Testing Set.

$L \leftarrow GetLabels(F)$ ▷ Get all labels of dataset F.
$F \leftarrow Normalize(F)$ ▷ Normalize all feature vectors.
$(RT, V) \leftarrow SplitTrainTest(F, r)$ ▷ Split F randomly into the raw training set RT and testing set V with ratio of r.
$(S_{maj}, S_{min}) \leftarrow GetClasses(RT)$ ▷ Determine majority classes (S_{maj}) and minority classes (S_{min}) from RT
$T \leftarrow \emptyset$
for each $M \in S_{maj}$ **do** ▷ Compression each majority class
  $C \leftarrow Clustering(M, |L|)$ ▷ Compute the centroids C of $|L|$ clusters by using ENN
  $M \leftarrow Select(M, C, \tau)$ ▷ Compress majority samples using C of L clusters
  $T \leftarrow T \cup M$
end for
for each $M \in S_{min}$ **do** ▷ Generate samples for minority classes by WGAN
  while $|M| < \tau$ **do**
    $S \leftarrow WGAN\_Sampling(M)$ ▷ Generate new samples
    $M = Denoise(M, S)$ ▷ Eliminate noise samples
  end while ▷ Repeat until get enough samples τ.
  $T \leftarrow T \cup M$ ▷ Add realistic samples to T
end for
return (T, V)

2.4.2 Vector hoá đặc trưng

Dữ liệu thô, thường ở định dạng JSON, cần được chuyển đổi thành các vector số để phục vụ huấn luyện mô hình AI. Để thực hiện điều này, chúng tôi áp dụng feature hashing, ánh xạ các token vào các vector có độ dài cố định trong khi vẫn giữ được đặc trưng của dữ liệu [**cTPS-ISA22**]. Bằng cách sử dụng các hàm kernel và sign hash, các đặc trưng được vector hóa, chuẩn hóa và lưu trữ dưới dạng CSV. Cuối cùng, các thuộc tính dạng phân loại được mã hóa bằng label encoding và one-hot encoding, đảm bảo khả năng tương thích với các mô hình ML như GBM và mạng nơ-ron.

2.4.3 Chuẩn hoá đặc trưng

Chuẩn hóa các đặc trưng giúp chúng có giá trị trung bình bằng 0 và độ lệch chuẩn bằng 1, từ đó hỗ trợ quá trình học của thuật toán máy học. Kỹ thuật chuẩn hóa này giúp tăng tốc độ hội tụ và cải thiện hiệu năng tổng thể của mô hình.

2.4.4 Tối ưu hoá đặc trưng dựa trên SHAP

Phương pháp tối ưu hóa đặc trưng bằng SHAP (OFS), lựa chọn tập con các đặc trưng quan trọng nhất từ dữ liệu huấn luyện bằng cách kết hợp hiệu năng mô hình với khả năng giải thích. Giả mã tổng quát để tối ưu hóa tập đặc trưng bằng SHAP được trình bày trong thuật toán 2.3.

2.5 Thực nghiệm và đánh giá

2.5.1 Chuẩn bị dữ liệu

- DS1:CSE-CIC-IDS2018 and NSL-KDD datasets.

Algorithm 2.3 OFS: **O**ptimizing **F**eature **S**et Using **S**HAP

Input: DS - dataset with the feature set F; M - m AI models; τ - threshold to drop features.

$X, y \leftarrow DS$ ▷ Get dataframes for features and labels
$X \leftarrow \text{Normalize}(X)$ ▷ Normalize all features to [0,1]
$FS \leftarrow \emptyset$ ▷ Init the feature set list.
for each $m \in M$ **do** ▷ Determine the feature importance for each AI model m.
  $AI \leftarrow m.fit(X, y)$ ▷ Train m using the dataset.
  if m is a boosting model **then**
    $shap\_values_m \leftarrow SHAP.TreeExplainer(m)$ ▷ Compute the SHAP values of all features based on decision tree model.
  else
    $shap\_values_m \leftarrow SHAP.DeepExplainer(m)$ ▷ Compute the SHAP values of all features based on DL model.
  end if
  $FS.push(shap\_values_M)$ ▷ Push the Shapley values of the model M into the list FS.
end for
$OFS \leftarrow \emptyset$
for each $f \in F$ **do**
  $shap\_values \leftarrow FS[f]$ ▷ Get SHAP values of feature f on all models M.
  if $shap\_values \geq \tau$ **then**
    $OFS \leftarrow OFS \cup f$ ▷ Consider f being important and add to OFS in the case of all its SHAP values $\geq \tau$.
  end if
end for

Output: OFS - Optimized Feature Set.

Để tăng cường khả năng phát hiện SQL-injection, chúng tôi cũng xây dựng một hệ thống thử nghiệm như thể hiện trong Figure 2.1b. Cuối cùng, dựa trên quy trình chuẩn bị dữ liệu, chúng tôi thu được hai bộ dữ liệu tăng cường DS1, được minh họa trong Table 2.1a. Lưu ý rằng các bộ dữ liệu DS1 sẽ được đánh giá toàn diện trong chapter 3.

- DS2:Chúng tôi cũng lựa chọn CSE-CIC-IDS2018 và NSL-KDD để đánh giá thực nghiệm hiệu quả của 2.2. Cuối cùng, Bảng 2.1b tóm tắt số lượng mẫu cho từng lớp của cả hai bộ dữ liệu.

Bảng 2.1: Số lượng mẫu sau khi tăng cường dữ liệu

(a) Tăng cường dựa trên độ khó (Difficulty-Aware)

Label	Original	Train	Test
CSE-CIC-IDS2018			
Benign	4,360,029	20,000	6,000
Bot	282,310	20,000	6,000
DDoS-HOIC	668,461	20,000	6,000
DoS-GoldenEye	41,455	20,000	6,000
DoS-Hulk	434,873	20,000	6,000
Infiltration	160,604	20,000	6,000
SQL-Injection	26,797	20,000	6,000
DoS-SlowHTTPTest	19,462	13,623	4,491
DoS-Slowloris	10,285	14,826	2,373
DDoS-LOIC-UDP	1,211	1,588	279
BruteForce-Web	253	978	58
BruteForce-XSS	151	106	35
NSL-KDD			
Benign	61,343	20,000	6,000
DoS	39,927	20,000	6,000
Probe	8,153	20,000	1,881
R2L	697	4,467	161
U2R	36	36	8

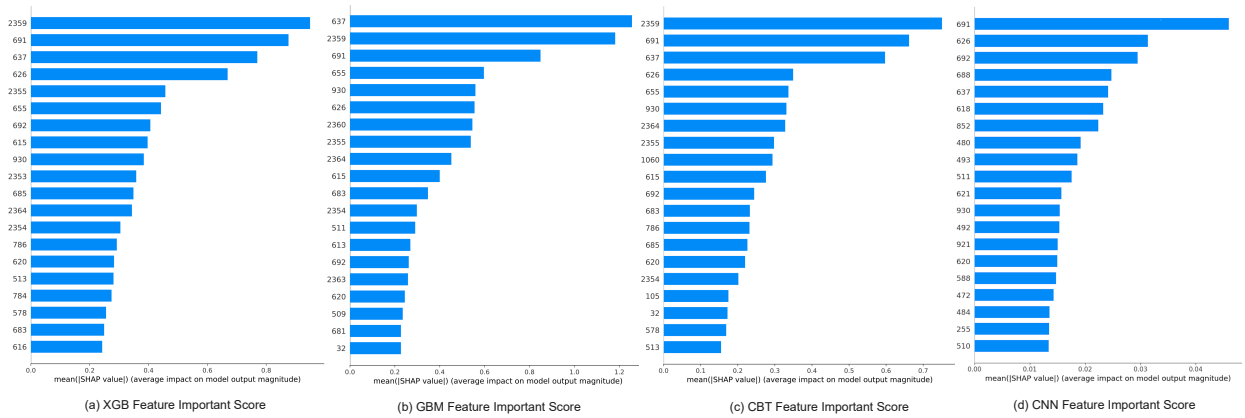
(b) Tăng cường dựa trên AWGAN

Label	Original	Train	Test
CSE-CIC-IDS2018			
Benign	4,360,029	14,000	6,000
Infiltration	160,604	14,000	6,000
Bot	282,310	14,000	6,000
DDoS-HOIC	668,461	14,000	6,000
DoS-GoldenEye	41,455	14,000	6,000
DoS-Hulk	434,873	14,000	6,000
DoS-SlowHTTPTest	13,067	14,000	4,082
DoS-Slowloris	6,977	14,000	2,093
DDoS-LOIC-UDP	1,120	14,000	336
BruteForce-Web	261	14,000	78
BruteForce-XSS	97	14,000	29
SQL-Injection	53	14,000	17
NSL-KDD			
Benign	61,343	14,000	6,000
DoS	39,927	14,000	6,000
Probe	8,333	14,000	2,500
R2L	637	14,000	191
U2R	40	14,000	12

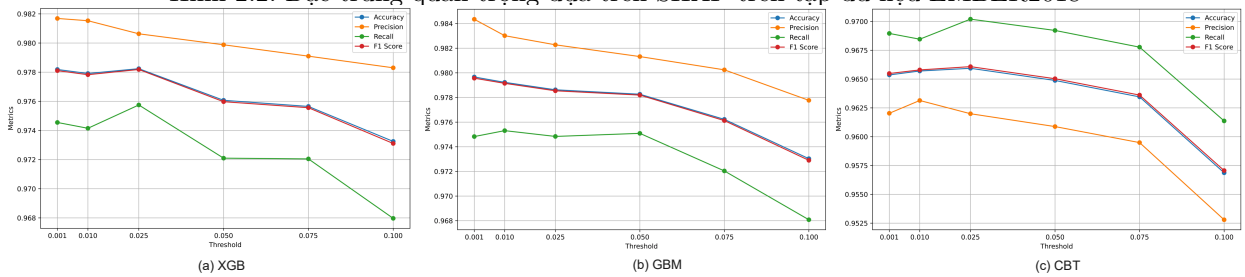
- DS3:EMBER2017, EMBER2018 và BODMAS được sử dụng để đánh giá thực nghiệm hiệu quả của 2.3, và kết quả đầu ra tạo thành DS3. Chúng tôi sử dụng sáu ngưỡng: 0.1, 0.075, 0.05, 0.025, 0.01 và 0.001. Với mỗi ngưỡng, các đặc trưng có giá trị SHAP $\geq$ ngưỡng được chọn, như thể hiện trong Figure 2.2 và Figure 2.4a. Chúng tôi nhận thấy rằng ngưỡng 0.025 cho kết quả tốt nhất, như minh họa trong Figure 2.3.

2.5.2 Kết quả và đánh giá

- S1: Chúng tôi đánh giá kỹ lưỡng 2.1 trên bộ dữ liệu DS1 nhằm khảo sát hiệu quả của thuật toán trong việc xử lý vấn đề mất cân bằng lớp.



Hình 2.2: Đặc trưng quan trọng dựa trên SHAP trên tập dữ liệu EMBER2018



Hình 2.3: Hiệu năng dựa trên ngưỡng trên tập dữ liệu EMBER2018

- S2: AWGAN 2.2 được đánh giá trên DS2 nhằm kiểm chứng một cách nghiêm ngặt khả năng tạo ra các mẫu tổng hợp chân thực và đa dạng cho các lớp thiểu số.
- S3: OFS 2.3 được kiểm chứng trên DS3, tập trung vào các tác vụ phát hiện mã độc tĩnh.

Kết quả kịch bản S1

Kết quả được tóm tắt trong Table 2.1a. Hiệu ứng này cũng thể hiện rõ trong các trực quan hóa t-SNE: trước và sau khi cân bằng, như trong Figure 2.5a, Figure 2.5b và Figure 2.5a, Figure 2.5b.

Sự cải thiện trong phân bố và phân tách lớp này củng cố cho những mức tăng hiệu năng mà chúng tôi quan sát được, như thể hiện trong Figure 2.4b.

Kết quả kịch bản S2

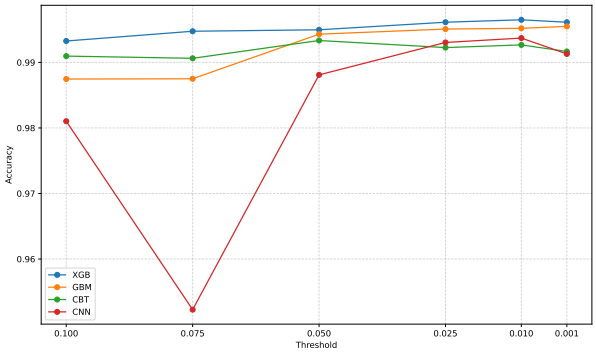
Table 2.1b tóm tắt số lượng mẫu cho mỗi lớp trong cả hai bộ dữ liệu. Các mô hình riêng lẻ được thể hiện trong Table 2.2. Figure 2.6a và Figure 2.6b minh họa dữ liệu gốc trước khi thực hiện tăng cường dựa trên AWGAN, trong khi Figure 2.6a và Figure 2.6b thể hiện các tập huấn luyện đã được tăng cường.

Bảng 2.2: Đánh giá các mô hình AI trên Tăng cường dữ liệu dựa trên WGAN (tính theo %)

Metric	CSE-CIC-IDS2018					NSL-KDD				
	XGB	CBT	GBM	BME	DNN	XGB	CBT	GBM	BME	DNN
F1	99.77	99.92	**99.95**	99.77	97.75	**99.48**	99.21	**99.48**	**99.48**	98.00
Acc	99.76	99.92	99.96	**99.98**	97.54	99.49	99.22	**99.56**	99.43	98.07
Prec	99.83	99.93	99.96	**99.98**	98.20	**99.49**	99.21	**99.49**	99.41	98.03
Rec	99.76	99.92	99.96	**99.98**	97.54	**99.49**	99.22	**99.49**	99.43	98.07
FPR	**0**	**0**	0.03	**0**	0.13	0.67	1.27	**0.63**	0.77	1.22
FNR	**0**	0.01	**0**	**0**	1.37	0.37	0.39	**0.30**	0.32	2.26
AUC	**100**	**100**	99.99	99.99	98.69	**99.99**	99.98	**99.99**	99.89	99.85

Kết quả kịch bản S3

Để đánh giá hiệu quả tối ưu hóa đặc trưng và cân bằng dữ liệu, chúng tôi so sánh hiệu năng mô hình trên các bộ dữ liệu gốc được trình bày trong Table 2.3 và trên các tập đặc trưng đã được tối ưu hóa được trình bày trong Table 2.4 đối với bộ dữ liệu EMBER2018 và BODMAS.

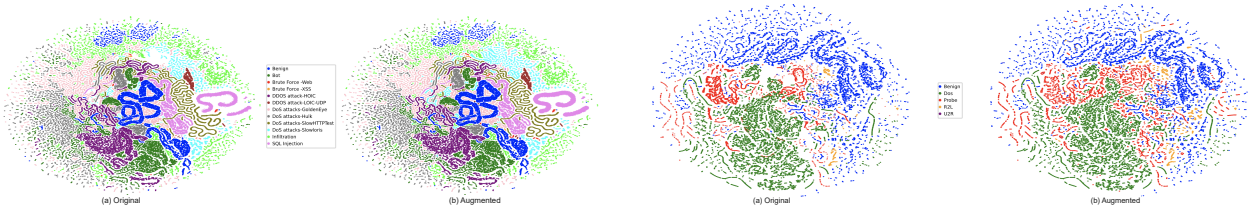


(a)

Metric	*CSE-CIC-IDS2018*			*NSL-KDD*		
	DNN	XGB	GBM	DNN	XGB	GBM
Acc	99.73	99.58	**99.74**	98.80	**99.66**	99.43
Prec	**99.80**	99.59	99.59	98.84	**99.66**	99.44
F1	**99.66**	99.58	99.58	98.80	**99.66**	99.43
Rec	**99.73**	99.58	99.58	**99.80**	99.66	99.43
AUC	99.96	**100**	**100**	99.84	**100**	99.92

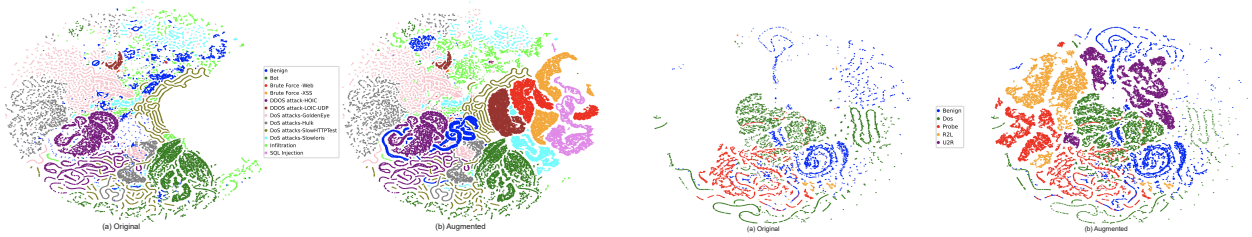
(b)

Hình 2.4: (a) Hiệu năng theo ngưỡng trên BODMAS; (b) Đánh giá mô hình trên dữ liệu tăng cường.



(a) CSE-CIC-IDS2018 (b) NSL-KDD

Hình 2.5: Trực quan hoá tập dữ liệu huấn luyện dựa trên nén và phóng đại



(a) CSE-CIC-IDS2018 (b) NSL-KDD

Hình 2.6: Trực quan hoá tập dữ liệu huấn luyện dựa trên AWGAN

Bảng 2.3: Đánh giá các mô hình AI trên các tập dữ liệu gốc (tính theo %)

Method	F1	Acc	Prec	Sens	FAR	FNR
EMBER2017 Evaluation						
XGB	99.16	99.16	99.16	99.16	0.84	0.84
CBT	**99.27**	**99.27**	**99.27**	**99.27**	**0.73**	**0.73**
GBM	98.67	98.67	98.67	98.67	1.33	1.33
CNN	95.95	96.04	93.72	95.95	3.35	4.05
EMBER2018 Evaluation						
XGB	97.63	97.63	97.63	97.63	2.37	2.37
CBT	97.19	97.19	97.19	97.19	2.81	2.81
GBM	**97.80**	**97.80**	**97.80**	**97.80**	**2.20**	**2.20**
CNN	94.03	94.02	94.16	94.02	5.97	5.98
BODMAS Evaluation						
XGB	98.71	98.69	99.68	97.75	0.32	2.25
CBT	**98.94**	**98.93**	99.88	**98.02**	0.12	**1.98**
GBM	98.90	98.89	**99.94**	97.88	**0.06**	2.12
CNN	98.90	98.89	99.87	97.96	0.13	2.04

Bảng 2.4: Đánh giá các mô hình AI dựa trên Tối ưu hóa tập đặc trưng (tính theo %)

Method	F1	Acc	Prec	Sens	FAR	FNR	F1	Acc	Prec	Sens	FAR	FNR
	BODMAS (4 features)						BODMAS (165 features)					
XGB	**89.76**	**90.78**	**85.19**	94.85	4.82	5.15	**99.28**	**99.39**	99.28	99.28	**0.61**	0.72
CBT	**89.76**	90.77	85.17	94.86	4.83	5.14	99.26	99.37	**99.29**	99.23	0.71	0.77
GBM	**89.76**	**90.78**	85.16	94.89	4.80	5.11	99.13	99.26	99.09	99.16	0.74	0.84
CNN	88.03	88.95	81.77	**95.34**	**4.66**	**4.66**	99.13	99.26	99.02	**99.24**	0.76	**0.74**
	EMBER2018 (170 features)						EMBER2018 (565 features)					
XGB	97.59	97.59	**97.84**	97.34	2.16	2.66	97.67	97.68	97.97	97.37	2.17	2.63
CBT	97.45	97.45	97.52	97.37	2.25	**2.53**	97.52	97.52	97.58	**97.46**	2.26	**2.54**
GBM	**97.85**	**97.86**	97.23	**97.47**	**2.13**	**2.53**	**97.88**	**97.89**	**98.34**	97.42	**2.16**	2.58
CNN	95.72	95.72	95.71	95.73	4.03	4.27	95.72	95.90	95.64	95.19	4.08	4.81

2.6 Tóm tắt chương

Các kết quả nghiên cứu này đã được trình bày một phần trong các công trình đã công bố, cụ thể: VVH-J2 giới thiệu một thuật toán giải quyết thách thức về mất cân bằng lớp thông qua kỹ thuật nén và phóng đại dữ liệu. VVH-J1 và VVH-C4 đề xuất các phương pháp dựa trên GAN có khả năng tạo ra các mẫu mới để tăng cường cho lớp thiểu số để giảm thiểu mất cân bằng dữ liệu. VVH-j3 đưa ra một phương pháp tối ưu hóa đặc trưng nhằm cải thiện chất lượng của bộ dữ liệu.

3 Tăng cường năng lực phát hiện xâm nhập dựa trên AI với suy luận kết hợp giữa Deep Learning và Boosting

3.1 Phát biểu bài toán

Các phương pháp dựa trên mô hình đơn lẻ thường dẫn đến hiệu năng không ổn định và khả năng chống chịu kém trước các mối đe dọa đối kháng. Để khắc phục, chúng tôi đề xuất một khung làm việc ensemble kết hợp giữa các mô hình học sâu và boosting thông qua cơ chế soft voting và stacking, nhằm cải thiện độ chính xác, tính bền vững và hiệu quả.

3.2 Phát hiện xâm nhập mạng thông qua phân tích chuyên sâu AI

3.2.1 Hướng tiếp cận

Chúng tôi đã phát triển giải pháp SDAID, một phương pháp phát hiện xâm nhập mạng toàn diện sử dụng phân tích dựa trên AI chuyên sâu để nhận diện hành vi bất thường, như minh họa trong Figure 3.1a và 3.1.

3.2.2 Mô hình hóa luồng lưu lượng mạng

Chúng tôi đề xuất sử dụng CICFlowMeter để thực hiện nhiệm vụ trích xuất đặc trưng.

3.2.3 Thuật toán phát hiện xâm nhập dựa trên DNN

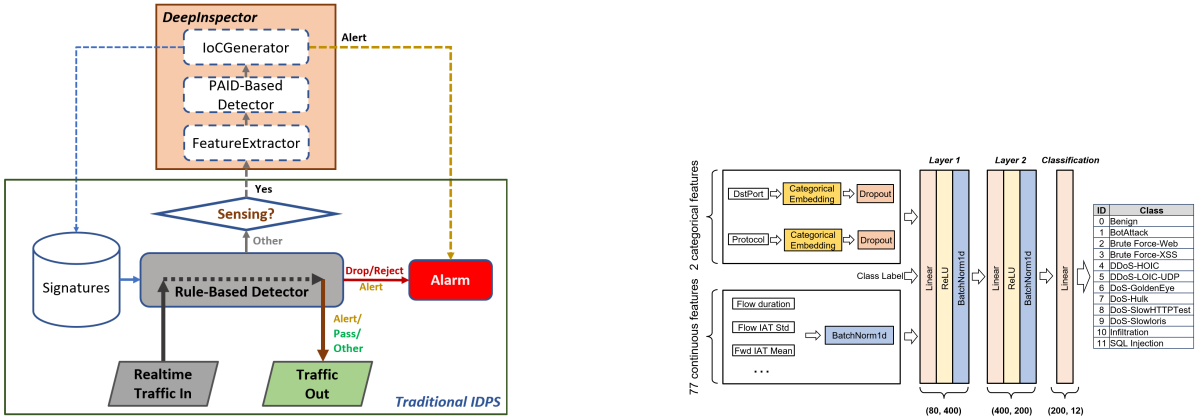
Mô hình DNN của chúng tôi được mô tả như trong Figure 3.1b.

3.2.4 Thuật toán phát hiện xâm nhập dựa trên Boosting

Các thuật toán boosting, chẳng hạn như XGBoost, xây dựng mô hình theo cách tuần tự, trong đó mỗi cây mới sẽ hiệu chỉnh các lỗi của cây trước đó, từ đó đạt được độ chính xác cao và khả năng mở rộng tốt. Bằng cách tinh chỉnh các siêu tham số, chúng tôi giảm hiện tượng overfitting và tăng cường khả năng khái quát hóa.

3.2.5 Tối ưu hoá tham số

Chúng tôi lựa chọn các tham số mô hình dựa trên một kỹ thuật gọi là Tối ưu hóa siêu tham số (Hyperparameter Optimization) [**jHyperparam21**]. Các giá trị tối ưu cũng được minh họa trong Table 3.1.



(a) (b)

Hình 3.1: (a) Phát hiện xâm nhập mạng bằng AI; (b) Phát hiện xâm nhập dựa trên DNN.

Algorithm 3.1 PAID: Perform an Ensemble Learning for AI-powered Intrusion Detection

Model: XGB - XGB trained model; DNN - DNN trained model; GBM - GBM trained model
Input: f - traffic flow.
Output: (msg, IoC) - (alert message; generated new IoC)

$R \leftarrow \emptyset$
$F \leftarrow CICFlowMeter(f)$ ▷ extract 83 features of traffic flow f
$Fin \leftarrow F \backslash [FlowID, SrcIP, SrcPort, Label]$ ▷ remove 4 unused features
$Cats \leftarrow [DstPort, Protocol]$ ▷ Categorical Variables
$Conts \leftarrow Fin \setminus Cats$ ▷ 77 Continuous Variables
Perform three processes P1,P2,P3:
P1: $dnn\_preds \leftarrow DNN.predict(Cats, Conts)$ ▷ perform the prediction using DNN model
P2: $xgb\_preds \leftarrow XGB.predict(Cats, Conts)$ ▷ perform the prediction using XGB model
P3: $gbm\_preds \leftarrow GBM.predict(Cats, Conts)$ ▷ perform the prediction using GBM model
Wait P1, P2, P3 finished.
$avgs \leftarrow (xgb\_preds + dnn\_preds + gbm\_preds)/3)$
$FC \leftarrow avgs.argmax(axis = 1)$ ▷ get the flow labels from 0 to 11
if $FC! = 0$ **then** ▷ classified as network attacks
  $msg \leftarrow Alert(FC)$ ▷ constitute an alert by using metadata from the flow f; set alert category being as label
  $R \leftarrow IoCGenerator(FC)$ ▷ generate a new IoC to handle the next similar flows
end if
return $msg; IoC$

3.2.6 Thực nghiệm và đánh giá

True Label \ Predicted Label	Benign	Bot	BruteForce-Web	BruteForce-XSS	DDOS-HOIC	DDOS-LOIC-UDP	DoS-GoldenEye	DoS-Hulk	DoS-SlowHTTPTest	DoS-Slowloris	Infiltration	SQL-Injection
Benign	5999 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0%	0 0%
Bot	0 0%	6000 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
BruteForce-Web	0 0%	0 0%	58 95%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
BruteForce-XSS	0 0%	0 0%	0 0%	34 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0%	0 0%
DDOS-HOIC	0 0%	0 0%	0 0%	0 0%	6000 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
DDOS-LOIC-UDP	0 0%	0 0%	0 0%	0 0%	0 0%	279 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
DoS-GoldenEye	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	6000 100%	0 0%	0 0%	0 0%	0 0%	0 0%
DoS-Hulk	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	6000 100%	0 0%	0 0%	0 0%	0 0%
DoS-SlowHTTPTest	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	4491 100%	0 0%	0 0%	0 0%
DoS-Slowloris	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	2373 100%	0 0%	0 0%
Infiltration	7 0%	0 0%	3 5%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0%	0 0%	5989 100%	0 0%
SQL-Injection	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	6000 100%

Bảng 3.2: Ma trận nhầm lẫn của S1

True Label \ Predicted Label	DoS	Probe	R2L	U2R	Benign
DoS	6000 100%	0 0%	0 0%	0 0%	0 0%
Probe	0 0%	1874 100%	0 0%	0 0%	7 0%
R2L	0 0%	0 0%	151 100%	1 14%	9 0%
U2R	0 0%	1 0%	0 0%	6 86%	1 0%
Benign	4 0%	6 0%	0 0%	0 0%	5990 100%

Bảng 3.3: Ma trận nhầm lẫn của S2

Để đánh giá 3.1, chúng tôi thực hiện 2 kịch bản S1 và S2 dựa trên CSE-CIC-IDS2018 và NSL-KDD.

Kết quả thực nghiệm kịch bản S1

Ma trận nhầm lẫn minh họa kết quả thí nghiệm của chúng tôi được thực hiện với phương pháp PAID, như thể hiện trong Table 3.2 và phần đầu của Table 3.4.

Bảng 3.1: Tối ưu hóa siêu tham số

Model	Hyperparameter	Value	Optimal
DNN	Learning rate	[0.001, 1.0]	0.003
	Batch size	[16, 32, 48, 64, 96, 128]	64
	Epochs	[1, 2, ..., 15, 16]	5
	Layers	[[200, 100], ..., [1000, 500]]	[400, 200]
XGB	Learning rate	[0,1]	0.01
	n_estimators	[1,∞]	30
	max_depth	[0,∞]	6
GBM	Learning rate	[0,1]	0.02
	min_samples_leaf	[1,∞]	30
	max_depth	[0,∞]	9

Bảng 3.4: Đánh giá hiệu năng dựa trên phát hiện xâm nhập mạng

Metric	*S1 (CSE-CIC-IDS2018)*				*S2 (NSL-KDD)*			
	DNN	XGB	GBM	**PAID**	DNN	XGB	GBM	**PAID**
Acc	99.73	99.58	99.74	**99.97**	98.80	99.66	99.43	**99.69**
Prec	99.80	99.59	99.59	**99.97**	98.84	99.66	99.44	**99.69**
F1	99.66	99.58	99.58	**99.97**	98.80	99.66	99.43	**99.69**
Rec	99.73	99.58	99.58	**99.97**	99.80	99.66	99.43	**99.69**
AUC	99.96	**100**	**100**	**100**	99.84	**100**	99.92	99.99

Kết quả thực nghiệm kịch bản S2

Do đó, chúng tôi trình bày kết quả thí nghiệm cho phương pháp PAID dưới dạng ma trận nhầm lẫn, như thể hiện trong Table 3.3 và phần thứ hai của Table 3.4.

3.2.7 So sánh với các phương pháp hiện nay

So sánh hiệu năng phát hiện xâm nhập giữa PAID và SOTA được tóm tắt trong Table 3.5.

3.3 Phát hiện mã độc thông qua học tổ hợp DL và Boosting

3.3.1 Hướng tiếp cận

Chúng tôi áp dụng học ensemble, bao gồm soft voting và stacking, để xây dựng các mô hình phân loại nhị phân cho bài toán phát hiện mã độc. Phương pháp được đề xuất trong nghiên cứu này có tên là MDOB, viết tắt của *"Enhancing Resilient and Explainable AI-Powered **M**alware **D**etection using Feature **O**ptimization and **M**utual **D**eep+**B**oosting Ensemble Learning."* Kiến trúc tổng thể của phương pháp MDOB được minh họa trong Figure 3.2a.

3.3.2 Học tăng cường tương hỗ giữa Deep Learning và Boosting

Chúng tôi đề xuất một phương pháp học tương hỗ tích hợp giữa học sâu (DL) và các mô hình gradient boosting (GBM) cho bài toán phát hiện mã độc, tận dụng AutoGluon để lựa chọn, tinh chỉnh và tối ưu hóa mô hình. Bằng cách kết hợp cả hai, hệ thống của chúng tôi nâng cao độ chính xác, khả năng chống chịu và tính thích ứng trước các mối đe dọa ngày càng phát triển.

3.3.3 Kết hợp học tập tổ hợp theo phương pháp Voting và Stacking

Our approach integrates voting and stacking learning to construct a more robust model using multiple AI-based classifiers. This process is illustrated in 3.2.

Bảng 3.5: So sánh PAID với các phương pháp SOTA khác

Method	Acc	Prec	F1	Rec
CSE-CIC-IDS2018-based Evaluation				
PAID (our)	**99.97**	**99.97**	**99.97**	**99.97**
WGAN+IDR [**jGANBalance22**]	–	99	98	97
RANet [**jRANet22**]	96.73	–	96.59	96.73
Adaboost [**00057**]	99.69	99.70	99.70	99.69
Autoencoder [**00056**]	99.20	95.00	-	98.90
AUE [**00052**]	97.90	98.00	98.00	98.00
DSSTE + miniVGGNet [**00004**]	96.99	97.46	97.04	96.97
LSTM + AM + SMOTE [**00054**]	96.20	96.00	93.00	96.00
NSL-KDD-based Evaluation				
PAID (our)	**99.69**	**99.69**	**99.69**	**99.69**
Autoencoder [**00044**]	99.20	-	-	99.27
Multiple LSTM [**00016**]	98.94	-	-	99.23
SMO [**00049**]	96.20	-	-	-
RANet [**jRANet22**]	83.23	–	82.57	83.23
DNN [**00027**]	78.50	81.00	76.50	78.50

Algorithm 3.2 VSEL: Combination of Voting and Stacking Ensemble Learning

Input: $TD = \{(X^i, y^i)\}_{i=1}^{N}$ - training dataset with optimized features; $MS = \{M_1, M_2, ..., M_m\}$ - set of m base models; $model\_params$ - optimized hyperparameters of m AI models; K - number of folds for building meta training dataset (MTD).

$MTD \leftarrow \emptyset$ ▷ Init MTD
$\{TD_1, TD_2, ..., TD_K\} \leftarrow Split(TD, K)$ ▷ Split the training dataset into K folds
for each fold $k \in 1..K$ **do**
  $TD_{\text{train}} \leftarrow TD \setminus TD_k$; $TD_{\text{val}} \leftarrow TD_k$ ▷ Use $K-1$ folds for training and 1 fold for validation
  for each $M_i \in MS$ **do**
    $M_i \leftarrow \text{Train}(M_i, TD_{\text{train}}, model\_params[M_i])$
  end for
  for each $(X, y) \in TD_{\text{val}}$ **do**
    $meta \leftarrow \emptyset$; $vote\_sum \leftarrow 0$ ▷ Create the meta-feature vector
    for each $M_i \in MS$ **do**
      $p_i \leftarrow M_i(X)$ ▷ Predict the probability for X using the trained base model M_i
      meta.push(p_i)
      $vote\_sum \leftarrow vote\_sum + p_i$
    end for
    $p_{\text{vote}} \leftarrow vote\_sum/m$ ▷ Calculate soft voting prediction from all base models
    meta.push(p_{vote}) ▷ Add soft voting result as an additional feature m+1 in the meta-layer
    $MTD.push(meta, y)$ ▷ Add the meta-feature vector and corresponding label to MTD
  end for
end for
$MM \leftarrow \text{AutoML.SelectBestModel}(MTD)$ ▷ Perform AutoML on MTD to select the best as the meta model
for each $M_i \in MS$ **do**
  $M_i \leftarrow \text{Train}(M_i, TD, model\_params[M_i])$ ▷ Retrain all base models on the whole training dataset to be used in final prediction
end for

Output: MS - n trained AI models; MM - trained meta model.

3.3.4 Tối ưu hoá tham số

Để tối ưu hóa các mô hình ML trong phương pháp của chúng tôi, chẳng hạn như huấn luyện các mô hình riêng lẻ, chúng tôi sử dụng Optuna [**Conf_Akiba_2019_Optuna**]. Công việc này được thực hiện thông qua 3.3.

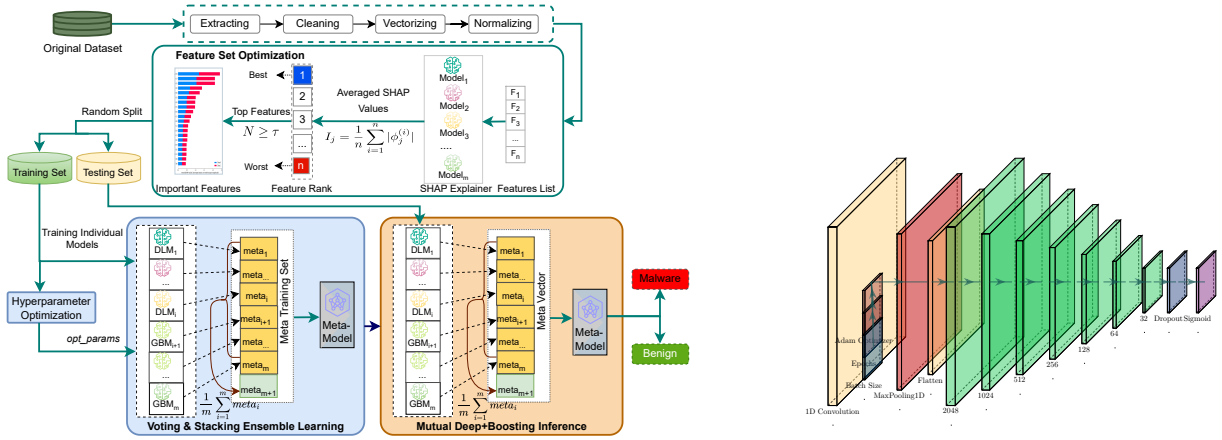
3.3.5 Thực nghiệm và đánh giá

Chúng tôi đã tiến hành hai kịch bản để đánh giá MDOB, chi tiết như sau:

- Kịch bản $S1$: Tập trung sử dụng bộ dữ liệu EMBER2018 để đánh giá phương pháp MDOB được đề xuất.
- Kịch bản $S2$: Đánh giá phương pháp MDOB được đề xuất bằng cách sử dụng bộ dữ liệu BODMAS.

Kết quả thực nghiệm kịch bản S1

Figure 3.3a minh họa quá trình tinh chỉnh mô hình CNN. Figure 3.3b so sánh điểm F1 của các mô hình khác nhau trên bộ dữ liệu EMBER2018 với 565 đặc trưng.



(a) (b)

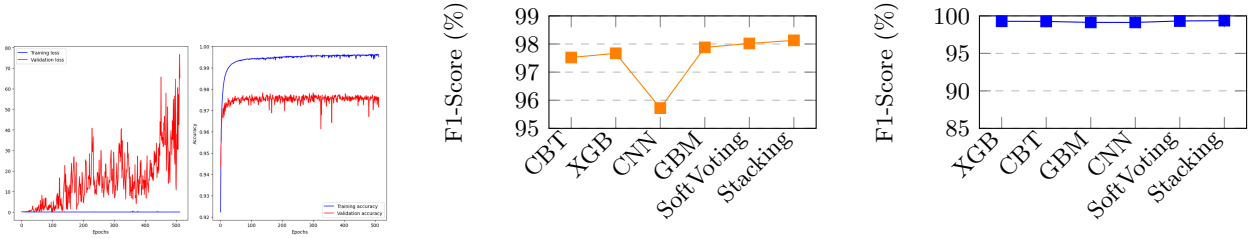
Hình 3.2: (a) Kiến trúc phát hiện mã độc dựa trên MDOB; (b) Kiến trúc mô hình CNN.

Algorithm 3.3 Hyperparameter Optimization using Optuna

Input: $model$ - AI model; $D_{train} = (X_{train}, y_{train})$ - training set; $D_{test} = (X_{test}, y_{test})$ - testing set; N_{trials} - number of trials; $T_{timeout}$ - optimization timeout; $params$ - list of hyperparameters.

function OBJECTIVE($trial$)
  $model\_params \leftarrow \{p_1, p_2, p_3, \ldots, p_n\}$ ▷ Initialize dictionary of hyperparameters for the model
  for $p \in params$ **do** ▷ Use Optuna to suggest hyperparameter values for each parameter p
    $model\_params[p] \leftarrow$ trial.suggest_⟨parameter_type⟩("p", ⟨min_value⟩, ⟨max_value⟩)
  end for
  $clf \leftarrow$ model(**model_params) ▷ Instantiate model with current parameters
  $clf.fit(X_{\text{train}}, y_{\text{train}})$ ▷ Train model on training data
  $preds \leftarrow clf.predict(X_{\text{test}})$ ▷ Make predictions on testing data
  $metric \leftarrow$ performance_metric($y_{\text{test}}, preds$) ▷ Compute evaluation metric
  return metric
end function
Initialize an empty dictionary $opt\_params = \emptyset$
Optimize the objective function using Optuna:
$study \leftarrow optuna.create\_study(direction = \text{“}maximize\text{”})$
study.optimize($objective$, n_trials=N_{trials}, timeout=T_{timeout})
$trial \leftarrow$ study.best_trial
$opt\_params \leftarrow trial.params$ ▷ Get optimized model parameters from the best trial
return opt_params

Output: opt_params - optimized hyperparameters.



(a) (b) (c)

Hình 3.3: (a) Hiệu năng huấn luyện CNN; (b) Kết quả trên EMBER2018; (c) Kết quả trên BODMAS.

Bảng 3.6: Đánh giá các mô hình AI dựa trên phát hiện mã độc (tính theo %)

Learning	Method	F1	Acc	Prec	Sens	FAR	FNR	F1	Acc	Prec	Sens	FAR	FNR
		BODMAS (165 features)						EMBER 2018 (565 features)					
Baseline	XGB	99.28	99.39	99.28	99.28	0.61	0.72	97.67	97.68	97.97	97.37	2.17	2.63
	CBT	99.26	99.37	99.29	99.23	0.71	0.77	97.52	97.52	97.58	97.46	2.26	2.54
	GBM	99.13	99.26	99.09	99.16	0.74	0.84	97.88	97.89	98.34	97.42	2.16	2.58
	CNN	99.13	99.26	99.02	99.24	0.76	0.74	95.72	95.90	95.64	95.19	4.08	4.81
Mutual DLM+GBM	Voting	99.32	99.42	99.34	99.30	0.66	**0.70**	98.02	97.89	98.38	97.65	2.03	2.35
Mutual Voting+Stacking	**MDOB**	**99.37**	**99.46**	**99.48**	**99.26**	**0.54**	0.74	**98.13**	**98.14**	**98.58**	**97.68**	**1.93**	**2.32**

Kết qủa thực nghiệm kịch bản S2

Kết quả được tóm tắt trong Table 3.6. Figure 3.3c trình bày hiệu năng F1-score trên bộ dữ liệu BODMAS với 165 đặc trưng.

Bảng 3.7: So sánh MDOB với các phương pháp SOTA (tính theo %)

Method	Venue	Acc	Prec	F1	Sens
	EMBER2018				
MDOB (our)	-	**98.14**	**98.58**	**98.13**	**97.68**
AutoML [**jAutoML2024**]	Computers & Security 2024	95.80	–	95.80	–
dualFFNN k-medoids [**jCOSE_FFNN_2023**]	Computers & Security 2023	98.02	–	–	–
Consensus [**j_Murat.2023**]	CMC 2023	96.77	–	96.77	–
DL [**jtelecom2023**]	Telecom 2023	95.57	–	–	–
MLMD [**jPCS2022**]	CAI 2023	97.42	–	–	–
DNN [**j_lad_2022**]	IJNIS 2022	94.09	90.14	88.66	88.85
	BODMAS				
MDOB (our)	-	**99.46**	**99.48**	**99.37**	**99.26**
EII-MBS [**jEII-MBS2022**]	Computers & Security 2022	99.29	98.26	94.23	98.07
MD-ADA [**j_MD-ADA_2024**]	Computers & Security 2024	99.29	–	99.13	–
FCG-MFD [**jFCG-MFD_2025**]	JNCA 2025	99.28	–	99.14	–

3.3.6 So sánh với các phương pháp hiện nay

So sánh các phương pháp triển khai phát hiện mã độc giữa MDOB và SOTA được tóm tắt trong Table 3.7.

3.4 Tóm tắt chương

Trong chương này, chúng tôi tập trung vào việc cải thiện hiệu năng và khả năng chống chịu của các hệ thống phát hiện xâm nhập và mã độc thông qua học ensemble và sự tương tác tương hỗ giữa các mô hình máy học. Dựa trên các bộ dữ liệu đã được tăng cường trong chapter 2, chương này giải quyết những hạn chế của các mô hình đơn lẻ và đề xuất một khung hợp nhất tận dụng sức mạnh bổ trợ của cả học sâu và các thuật toán boosting hiện đại.

4 Giải pháp phòng thủ xâm nhập quy mô lớn dựa trên AI với chiến lược cảm nhận luồng và suy luận tổ hợp song song

4.1 Phát biểu bài toán

Các mô hình truyền thống dựa trên chữ ký hoặc học sâu đơn lẻ bị hạn chế bởi độ trễ và khả năng thích ứng, thường cho hiệu năng kém trước các cuộc tấn công đang phát triển trong các mạng quy mô lớn. Để khắc phục những vấn đề này, chúng tôi đề xuất NetIPS, một hệ thống phòng chống xâm nhập chủ động tích hợp cảm biến luồng, suy luận song song và kiến trúc nhẹ trong không gian người dùng.

4.2 Đề xuất khung phát hiện xâm nhập toàn diện

4.2.1 Hướng tiếp cận

Phương pháp phát hiện xâm nhập toàn diện của chúng tôi sử dụng phân tích dựa trên AI chuyên sâu để nhận diện hành vi bất thường và các chữ ký của những cuộc xâm nhập trước đó, gọi là APELID, như minh họa trong Figure 4.1 và 4.1.

4.2.2 Phát hiện xâm nhập dựa trên học tổ hợp song song

Hai ý tưởng đã thúc đẩy phương pháp phát hiện xâm nhập của chúng tôi: cách tiếp cận học ensemble và tính toán song song. 4.2 trình bày thuật toán PELID của chúng tôi.

4.2.3 Chiến lược phát hiện xâm nhập thời gian thực dựa trên AI

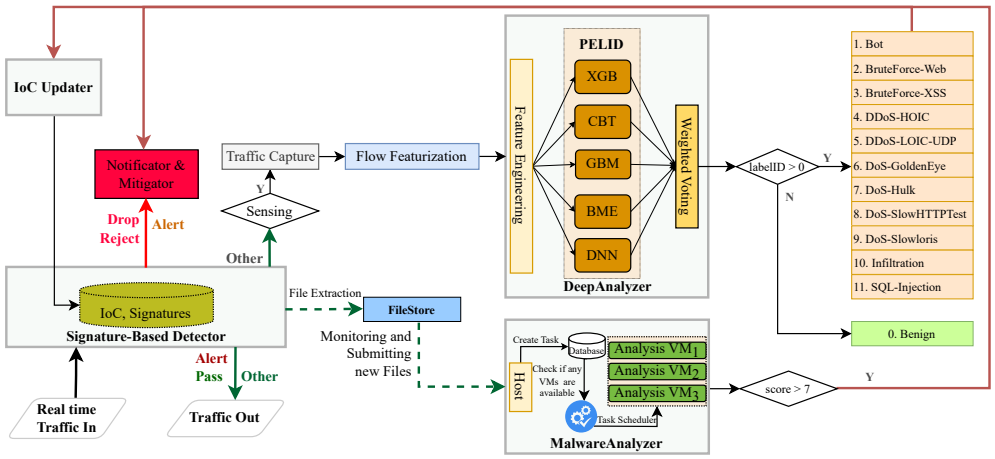
Đối với lưu lượng mạng quy mô lớn, việc phân tích chuyên sâu chắc chắn sẽ gây ra tình trạng nghẽn trong IDPS. Do đó, chúng tôi đề xuất một chiến lược hiệu quả để cảm nhận các luồng lưu lượng. Cụ thể, chúng tôi kiểm soát chiến lược lấy mẫu phân tích chuyên sâu theo chu kỳ bằng cách sử dụng 6 biến: DI_Cycle, DIC_Min, DIC_Max, và DI_Window, DIW_Min, DIW_Max.

4.2.4 Phát hiện mã độc bằng phương pháp Sandbox

Để nâng cao khả năng phát hiện các tệp mã độc được truyền qua mạng, giải pháp APELID mà chúng tôi đề xuất được tích hợp với một *MalwareAnalyzer* dựa trên phương pháp sandbox, như minh họa trong Figure 4.1. 4.3 minh họa chiến lược của chúng tôi để phân tích và nhận diện tệp mã độc này.

4.3 Thực nghiệm và đánh giá

1. RQ1: Việc kết hợp nhiều mô hình AI trong PELID, bao gồm cả ML truyền thống và DL, có giúp nâng cao hiệu năng phát hiện xâm nhập mạng và giảm thời gian phân tích hay không?
2. RQ2: Khi triển khai hệ thống IDPS inline trong mạng nội bộ với lưu lượng lớn, liệu nó có đủ nhanh để thực hiện phân tích chuyên sâu các luồng mạng phục vụ phát hiện xâm nhập bằng mô hình AI được tạo ra từ phương pháp APELID, nhằm đảm bảo xử lý luồng mạng theo thời gian thực hay không?
3. RQ3: Có thể triển khai phát hiện tệp mã độc trong hệ thống IDPS inline kết hợp với phân tích chuyên sâu dựa trên mô hình AI hay không?



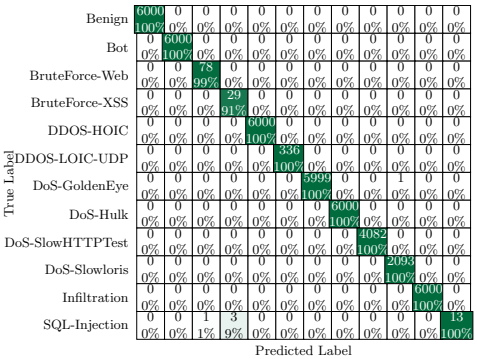
Hình 4.1: Kiến trúc của hệ thống phát hiện xâm nhập tổng thể

Algorithm 4.1 Holistic intrusion Detection by flow sensing strategy and deep analysis

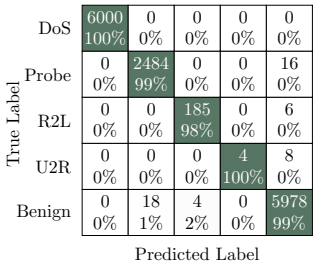
Input: f - Traffic In Flow, S - Signature Set, $Sensing$ - perform AI-powered deep analysis or not, F - Files that transfer between network.
Output: f, msg, S - (Traffic Out Flow; Alert Message; Updated Signature Set)

$action \leftarrow RuleBasedDetector(f, S)$
$IoC_{\text{set}} \leftarrow \emptyset$
if $action = Drop/Reject$ **then** ▷ Drop/Reject flow due of a detected critical attack
    Drop/Reject(f)
    $msg \leftarrow' CriticalAttack'$
    return $(none, msg, S)$
else if $action = Alert$ **then** ▷ Generate an alert
    $msg \leftarrow' Alert\_based\_on\_Signature'$
else if $action = Pass$ **then** ▷ Stop further inspection of the flow
    $msg \leftarrow None$
else if $Sensing = True$ **then** ▷ f does not match any rules, then AI-powered deep analysis is triggered by the sensing mechanism
    $(msg_{deep}, IoC_{deep}) \leftarrow DeepAnalyzer(f)$ ▷ Inspect F deeply by PELID and return a message and new IoC if an intrusion attack is detected.
    $IoC_{set} \leftarrow IoC_{set} \cup IoC_{deep}$ ▷ Update IoC_{set} with new indication of compromise IoC_{deep}
end if
for each $t \in F$ **do**
    $(msg_t, IoC_t) \leftarrow MalwareAnalyzer(t)$ ▷ Analysis t deeply by Sandbox return a message and new IoC_t if an malware file is detected.
    $IoC_{set} \leftarrow IoC_{set} \cup IoC_t$ ▷ Update IoC_{set} with new indication of compromise IoC_t
end for
$S \leftarrow S \cup IoC_{set}$ ▷ Update S with new indication of compromise IoC_{set}
return (f, msg, S)

4.3.1 Kết quả thực nghiệm



(a)



(b)

Bảng 4.1: (a) Ma trận nhầm lẫn của PELID trên CSE-CIC-IDS2018; (b) Ma trận nhầm lẫn của PELID trên NSL-KDD.

Kết quả thực nghiệm của tập dữ liệu CSE-CIC-IDS2018

Kết quả chi tiết của thí nghiệm trên bộ dữ liệu CSE-CIC-IDS2018 được minh họa trong phần đầu của Table 4.2 và ma trận nhầm lẫn thể hiện trong Table 4.1a.

Algorithm 4.2 PELID: Parallel Ensemble Learning-based Intrusion Detection

Model: XGB, GBM, CBT, BME, DNN - XGB, GBM, CBT, BME and DNN trained model, and their ensemble weight ω_i where $\sum_{i=1}^{5} \omega_i = 1$.
Input: f - traffic flow.
Output: (msg, R) - (alert messages; new generated rules)

$R \leftarrow \emptyset$
$F \leftarrow Featurize(f)$ ▷ Extract features of traffic flow f.
$Fin \leftarrow Normalize(F)$ ▷ Perform the feature engineering: remove unused features and normalize the rest.
$Cats \leftarrow [DstPort, Protocol]$ ▷ Categorical variables
$Conts \leftarrow Fin \setminus Cats$ ▷ Continuous variables
Perform in parallel five processes P1, P2, P3, P4, P5:
P1: $pXGB \leftarrow XGB.predict(Cats, Conts)$ ▷ Perform the prediction using XGB.
P2: $pGBM \leftarrow GBM.predict(Cats, Conts)$ ▷ Perform the prediction using GBM.
P3: $pCBT \leftarrow CBT.predict(Cats, Conts)$ ▷ Perform the prediction using CBT.
P4: $pBME \leftarrow BME.predict(Cats, Conts)$ ▷ Perform the prediction using BME.
P5: $pDNN \leftarrow DNN.predict(Cats, Conts)$ ▷ Perform the prediction using DNN.
Wait P1, P2, P3, P4, P5 finished.
$scores \leftarrow (pXGB * \omega_1 + pGBM * \omega_2 + pCBT * \omega_3 + pBME * \omega_4 + pDNN * \omega_5)$
$FC \leftarrow scores.argmax(axis = 1)$ ▷ Get the flow predicted label.
if $FC! = 0$ **then** ▷ Classified as network attacks
  $msg \leftarrow Alert(FC, f)$ ▷ Generate an alert by using metadata from the flow f; set alert category being as predicted label.
  $R \leftarrow RuleGenerator(FC, f)$ ▷ Generate a new signature based on its indicator of compromise.
end if
return $msg; R$

Algorithm 4.3 Malware Detection

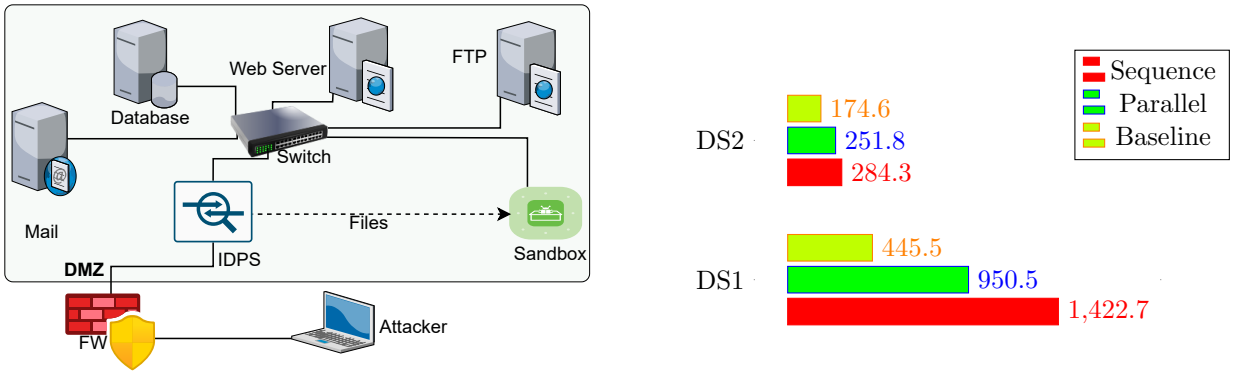
Input: F - New files transferred in network and accumulated in $FileStore$ folder.
Output: (msg, R) - (Alert Message, New Rules generated based malware detected files).

$Ready \leftarrow Wait\_Sandbox\_Ready$ ▷ Blocking-function until Sandbox is ready.
$IngestFiles(F)$ ▷ Send F in the $FileStore$ folder to Sandbox
$score = HybridAnalyzer(F)$ ▷ Determine the overall score of both static and dynamic analysis.
if $score > 7$ **then** ▷ Critical suspicious file
  $R \leftarrow RuleGenerator(F)$ ▷ Update the rule to block connection.
  $msg \leftarrow$ '$Detected\_Malware\_Files$'
  return msg, R
end if

Kết quả thực nghiệm của tập dữ liệu NSL-KDD

Phần thứ hai của Table 4.2 trình bày kết quả thực nghiệm trên bộ dữ liệu NSL-KDD, và Table 4.1b minh họa ma trận nhầm lẫn của mô hình PELID.

Kết quả thực nghiệm săn tìm mã độc



(a) (b)

Hình 4.2: (a) Kịch bản săn tìm mã độc; (b) So sánh thời gian xử lý của PELID (ms).

Kịch bản này bao gồm hai mạng hoàn toàn tách biệt: Mạng DMZ (bao gồm Web server (HTTP và FTP), Mail Server (SNMP), và Attacks-Network), như minh họa trong Figure 4.2a. Chúng tôi đã so sánh kết quả thực nghiệm với Virus Total (VT), được trình bày trong Table 4.3.

4.3.2 Đánh giá tác động

Hiệu quả của PELID trong phát hiện xâm nhập

So sánh với các mô hình AI đơn lẻ, như minh họa trong Table 4.2. Những kết quả này cho phép chúng tôi trả lời **RQ1**: việc kết hợp nhiều mô hình AI trong PELID giúp cải thiện khả năng phát hiện xâm nhập mạng.

Bảng 4.2: Đánh giá các mô hình AI dựa trên PELID (tính theo %)

Metric	CSE-CIC-IDS2018						NSL-KDD					
	XGB	CBT	GBM	BME	DNN	**PELID**	XGB	CBT	GBM	BME	DNN	**PELID**
F1	99.77	99.92	99.95	99.77	97.75	**99.99**	99.48	99.21	99.48	99.48	98.00	**99.63**
Acc	99.76	99.92	99.96	99.98	97.54	**99.99**	99.49	99.22	99.56	99.43	98.07	**99.65**
Prec	99.83	99.93	99.96	99.98	98.20	**99.99**	99.49	99.21	99.49	99.41	98.03	**99.65**
Rec	99.76	99.92	99.96	99.98	97.54	**99.99**	99.49	99.22	99.49	99.43	98.07	**99.65**
FPR	**0**	**0**	0.03	**0**	0.13	**0**	0.67	1.27	0.63	0.77	1.22	**0.37**
FNR	**0**	0.01	**0**	**0**	1.37	**0**	0.37	0.39	0.30	0.32	2.26	**0.34**
AUC	**100**	**100**	99.99	99.99	98.69	**100**	**99.99**	99.98	**99.99**	99.89	99.85	**99.99**

Bảng 4.3: Kết quả săn tìm mã độc

N	Malware	Type	Hash	VT	APELID
1	QuasarRAT	.exe	832ab3a898d188426d3541e1533b55f9	56/68	Yes
2	Loki	.xlsx	5b6aec60c3be4724f7980a659206531a	29/58	Yes
3	STRRAT	.jar	2199150e7d79d0e831cda314c7ce6f56	28/62	Yes
4	AsynRAT	.doc	da6419e4d4e4528990898bcfdaa85e01	32/60	Yes
5	SnakeKeylogger	.exe	715b0f6390ba4387a4155c1d59a3669c	49/69	Yes
6	AgentTesla	.exe	5c590fcb32aedec16532aa857eec28b5	40/66	Yes
7	OskiStealer	.xlsx	6a9203346218dded19d0a8a1dee24023	20/59	Yes
8	NanoCore	.exe	4bae18ac4a73ff38f7ed718365e6c2b2	41/67	Yes
9	DanaBot	.exe	5f4731a4ef7d1484893213caaf6a6685	42/69	Yes
10	DCRAT	.exe	ea800644b9dfd027807447fdd98241aa	50/68	Yes
11	YellowCockatoo	.dll	df7b2ece343c52df774d72e12ea09009	51/69	Yes
12	RemoteManipulator	.exe	4c5649e9b9a2d9997ac2600a804e0aeb	41/68	Yes
13	Pony	.exe	ab468a5b5cd9470c0895097efa2a687f	63/71	Yes
14	Stealc	.exe	cea30f806e644cebe48399eefa345e51	47/71	Yes
15	njRat	.exe	b17414d6949c2e013de14fdc268cfc89	65/71	Yes
16	RedLineStealer	.exe	8a61e10948c23a9a5c353d28b8738490	35/71	Yes
17	Guildma	.zip	8a61e10948c23a9a5c353d28b8738490	35/71	Yes
18	Gozi	.js	1df2e7a13459223b2cc55b93744add77	24/71	Yes
19	DarkTortilla	.exe	1c354a83f81063dc75612a9a7bd51225	54/71	Yes
20	VectorStealer	.xlsx	5b47098a17ecd534de15df03b12beacb	40/71	Yes

Bảng 4.4: So sánh APELID với các phương pháp SOTA (tính theo %)

CSE-CIC-IDS2018		
APELID (our)	**99.99**	**99.99**
MMM-RF [**jMMM-RF22**]	99.98	–
GAN+RF [**art_Lee_2021**]	99.83	98.68
KNN-MQBHOA [**article_Ghanbar_2023**]	99.78	99.56
HDLNIDS [**article_Qazi_2023**]	98.90	98.63
CNN [**IJNC293**]	98.17	95.00
AUE [**00052**]	97.90	98.00
miniVGGNet [**00004**]	96.99	97.46
NSL-KDD		
APELID (our)	**99.65**	**99.65**
KNN-MQBHOA [**article_Ghanbar_2023**]	99.00	99.00
FFO-PNN [**J_Alexandria_2023**]	98.99	96.97
DLNID [**art_Fu_2022**]	90.73	86.38
GMM-WGAN-IDS [**art_Cui_2022**]	86.59	88.55
Adaptive-Ensemble [**art_Gao_2019**]	85.20	86.50
CAFE-CNN [**art_Shams_2021**]	83.34	85.35

Hiệu quả của PELID về mặt thời gian xử lý

Figure 4.2b cho thấy thời gian dự đoán trung bình của PELID, và các kết quả thực nghiệm trong Bảng 4.2 đã giải quyết được **RQ2** và **RQ3**.

4.3.3 So sánh với các phương pháp hiện nay

Table 4.4 chứng minh rằng APELID vượt trội hơn so với SOTA và đạt điểm số cao nhất trên tất cả các chỉ số đánh giá, từ đó trả lời cho **RQ1**.

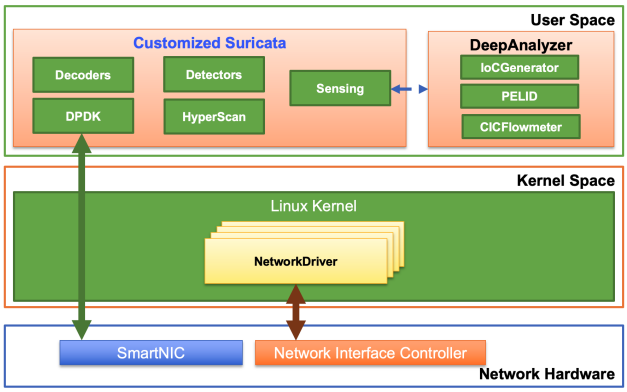
4.4 NetIPS: Triển khai Hệ thống phát hiện và phòng ngừa xâm nhập mạng

4.4.1 Mô hình triển khai

Kiến trúc được minh họa trong Figure 4.3 và được chia thành ba lớp. Lớp dưới cùng là phần cứng mạng, bao gồm SmartNIC (bộ tăng tốc mạng) và các giao diện mạng truyền thống, được sử dụng để phân tích lưu lượng và quản lý NetIPS.

4.4.2 Sử dụng Hyperscan trong phát hiện dựa trên tập luật

Trong Bộ phát hiện dựa trên luật (Rule-based Detector), kỹ thuật Hyperscan được sử dụng để nâng cao hiệu quả của quá trình so khớp tập luật. Phương pháp này cho khả năng so khớp hiệu quả hơn so với các phương pháp khác (chẳng hạn như Aho-Corasick, Boyer-Moore).



Hình 4.3: Kiến trúc NetIPS dựa trên APELID

4.4.3 Tăng tốc Phát hiện Xâm nhập dựa trên AI trong không gian người dùng

Trong NetIPS, việc xử lý gói tin được tối ưu hóa bằng cách thay thế NIC truyền thống bằng Napatech SmartNIC và tận dụng thư viện DPDK để bỏ qua chi phí xử lý của kernel, từ đó giảm chuyển ngữ cảnh và độ trễ.

4.5 Tóm tắt chương

Chương 4 giải quyết thách thức then chốt trong việc triển khai các hệ thống phát hiện và phòng chống xâm nhập dựa trên AI trong các môi trường thực tế quy mô lớn, nơi các yêu cầu về hiệu năng thời gian thực, khả năng mở rộng và độ tin cậy trong vận hành được đặt lên hàng đầu. Dựa trên các cải tiến về dữ liệu và mô hình ensemble được phát triển trong các chương trước, chương này giới thiệu và đánh giá một kiến trúc toàn diện cho giải pháp phòng thủ mạng thực tiễn với thông lượng cao.

Kết luận và Hướng nghiên cứu tương lai

Đóng góp chính

- Đề xuất một pipeline học máy với tăng cường dữ liệu và tối ưu hóa đặc trưng (tăng cường bằng WGAN + tối ưu hóa đặc trưng dựa trên SHAP) nhằm cân bằng và nâng cao chất lượng bộ dữ liệu huấn luyện, từ đó cải thiện khả năng phát hiện các tấn công thuộc lớp thiểu số.
- Giới thiệu khung suy luận tương hỗ giữa deep learning và boosting, giúp tăng cường độ chính xác và khả năng chống chịu của các hệ thống phát hiện xâm nhập và mã độc.
- Đề xuất giải pháp giải quyết điểm nghẽn dữ liệu trong phòng chống xâm nhập mạng quy mô lớn thông qua chiến lược cảm nhận luồng dữ liệu dựa trên khoảng thời gian và tần suất, kết hợp với quá trình suy luận song song của các mô hình deep và boosting tương hỗ.
- Tích hợp các phương pháp được đề xuất vào hệ thống NetIPS phát hiện và phòng chống xâm nhập theo thời gian thực, tận dụng các mô hình dựa trên AI ở tầng người dùng để xử lý lưu lượng lớn (quy mô doanh nghiệp và ISP), giúp hệ thống phù hợp với triển khai thực tiễn.

Hạn chế của Luận án

Mặc dù nghiên cứu đã đạt được những kết quả đầy hứa hẹn, nhưng cũng cần thừa nhận một số hạn chế sau:

- Tất cả các thử nghiệm được tiến hành trên các bộ dữ liệu cố định đã được chuẩn bị sẵn, điều này có nghĩa là chúng tôi chưa đánh giá được mức độ thích ứng của mô hình trong các tình huống thực tế hoặc khi dữ liệu thay đổi theo thời gian.
- Thành phần NetIPS chưa được kiểm chứng rộng rãi trong nhiều kịch bản thực tế khác nhau. Đặc biệt, chưa có các đánh giá toàn diện về hiệu năng phần cứng và tính khả thi khi triển khai trong các mạng sản xuất quy mô lớn.
- Thiết kế thí nghiệm hiện tại chưa bao gồm các nghiên cứu cắt bỏ (ablation studies) để định lượng mức đóng góp của từng thành phần hoặc kỹ thuật đối với hiệu năng tổng thể. Các đánh giá như vậy có thể cung cấp thêm chi tiết về hiệu quả của hệ thống và định hướng cho việc tối ưu hóa trong tương lai.
- Các mô hình chủ yếu được huấn luyện trên dữ liệu mạng có cấu trúc hoặc dữ liệu PE. Những vectơ tấn công phức tạp hơn như lưu lượng được mã hóa, mã độc nhiều giai đoạn hoặc tấn công chuỗi cung ứng chưa nằm trong phạm vi của nghiên cứu này.

Định hướng nghiên cứu trong tương lai

Dựa trên những nền tảng được xây dựng trong luận án này, một số hướng nghiên cứu có thể được tiếp tục khám phá như sau:

- Học trực tuyến và học liên tục: Tích hợp các phương pháp học trực tuyến và huấn luyện gia tăng (incremental retraining) vào pipeline phát hiện có thể giúp mô hình thích ứng với các mối đe dọa đang phát triển và xử lý hiệu quả hơn trong môi trường động.
- Đa dạng hóa nguồn dữ liệu: Các hệ thống trong tương lai có thể khai thác nhiều loại dữ liệu khác nhau như hành vi của máy chủ, cây tiến trình, hoạt động của người dùng, cũng như các mẫu trong lưu lượng được mã hóa, tất cả trong một khung phát hiện thống nhất.
- Tích hợp phản ứng và phòng thủ tự động: Nâng cao hệ thống phát hiện bằng các hành động tức thời như tự động chặn mối đe dọa, cập nhật luật, hoặc ưu tiên cảnh báo có thể kết nối việc phát hiện đơn thuần với phòng thủ chủ động.
- Nâng cao khả năng giải thích: Xây dựng các công cụ đơn giản, dễ sử dụng để giải thích cách thức hoạt động của hệ thống AI, đặc biệt là cho các hệ thống đầu cuối, sẽ giúp tăng cường niềm tin và hỗ trợ các nhà phân tích an ninh làm việc hiệu quả hơn với công cụ AI.

Personal Publications

Journals

VVH-J1 **Hoang V. Vo** and Hanh P. Du and Hoa N. Nguyen, AI-powered intrusion detection in large-scale traffic networks based on flow sensing strategy and parallel deep analysis, *Journal of Network and Computer Applications* 220 (2023) 103735. DOI: 10.1016/j.jnca.2023.103735; (IF 8.0, SCI-E, top 2% Q1-Scopus)

VVH-J2 **Hoang V. Vo** and Hanh P. Du and Hoa N. Nguyen, APEPID: Enhancing real-time intrusion detection with augmented WGAN and parallel ensemble learning, *Computers and Security* 136 (2024) 103567. DOI: 10.1016/j.cose.2023.103567; (IF 5.4, SCI-E, top 7% Q1-Scopus)

VVH-J3 **Hoang V. Vo** and Hanh P. Du and Hoa N. Nguyen, MDOB: Enhancing Resilient and Explainable AI-Powered Malware Detection Using Feature Set Optimization and Mutual Deep+Boosting Ensemble Inference. *Journal of Information Security and Applications* 2025 93 (2025) 104175. DOI: 10.1016/j.jisa.2025.104175; (IF 3.7, SCI-E, top 8% Q1-Scopus)

Conferences

VVH-C1 **Hoang V. Vo**, Hoa N. Nguyen, Tu N. Nguyen, Hanh P. Du, *SDAID: Towards a Hybrid Signature and Deep Analysis-based Intrusion Detection Method*, in: GLOBECOM 2022 - 2022 IEEE Global Communications Conference, 2022, pp. 2615–2620. DOI: 10.1109/GLOBECOM48 099.2022.10001582. (WoS, Scopus)

VVH-C2 **Hoang V. Vo**, Duong H. Nguyen, Tuyen T. Nguyen, Hoa N. Nguyen, Duan V. Nguyen, *Leveraging AI-Driven Realtime Intrusion Detection by Using WGAN and XGBoost*, in: Proceedings of the 11th International Symposium on Information and Communication Technology, Association for Computing Machinery, New York, NY, USA, 2022, p. 208–215. DOI: 10.1145/3568562.3568660. (WoS, Scopus)

VVH-C3 **Hoang V. Vo**, Phong H. Nguyen, Hau T. Nguyen, Duy B. Vu, Hoa N. Nguyen, *Enhancing AI-Powered Malware Detection by Parallel Ensemble Learning*, in: 2023 RIVF International Conference on Computing and Communication Technologies (RIVF), 2023, pp. 503–508. DOI: 10.1109/RIVF60135.2023.10471855. (WoS)

VVH-C4 **Hoang V. Vo**, Hanh P. Du and Hoa N. Nguyen, *AWDLID: Augmented WGAN and Deep Learning for Improved Intrusion Detection*, 2024 1st International Conference On Cryptography And Information Security (VCRIS), Hanoi, Vietnam, 2024, pp. 1-6, DOI: 10.1109/VCRIS63677.2024.10813392. (WoS)

references