

**ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN TỬ - VIỄN THÔNG**



BÁO CÁO ĐỒ ÁN CUỐI KỲ

**NGHIÊN CỨU VÀ XÂY DỰNG HỆ THỐNG PHÁT HIỆN, GIẢM
THiểu TẤN CÔNG DDOS TRONG MẠNG IOT SỬ DỤNG HỌC
MÁY TỐI ƯU HÓA (WOA-SSA)**

Sinh viên thực hiện: NGUYỄN HỒNG ÂN - 106220208
PHAN VĂN DANH - 106220212
PHAN VĂN TRÀ - 106230064
Giảng viên hướng dẫn: TS. NGUYỄN VĂN HIẾU

Đà Nẵng, Ngày 26 tháng 11 năm 2025

Bảng 1: Bảng phân công nhiệm vụ và mức độ đóng góp của các thành viên

Họ và tên sinh viên	Số thẻ SV	Lớp	Phân công nhiệm vụ	Đóng góp
Nguyễn Hồng Ân	106220208	22KTMT1	Hỗ trợ viết báo cáo phần giới thiệu, 2 phần đầu tiên của phần phương pháp, thực hiện cài đặt, mô phỏng và ra kết quả, làm slide.	25%
Phan Văn Danh	106220226	22KTMT1	Hỗ trợ viết báo cáo phần phân tích kết quả, hỗ trợ đánh giá và viết báo cáo phần kết quả mô phỏng, làm slide.	25%
Phan Văn Trà	106230064	23KTMT2	Viết code, lập trình, kiểm thử, phát triển phần mềm, viết báo cáo.	50%
Tổng				100%

Mục lục

1	TỔNG QUAN VÀ CƠ SỞ LÝ THUYẾT	6
1.1	Đặt vấn đề: An ninh mạng trong kỷ nguyên IoT	6
1.2	Cơ sở Lý thuyết về Tấn công Mạng	6
1.2.1	Tấn công Volumetric (UDP Flood)	6
1.2.2	Lý thuyết Hàng đợi và Nút thắt cổ chai (Bottleneck)	7
1.3	Cơ sở Lý thuyết về Học máy và Tối ưu hóa	8
1.3.1	Mô hình Random Forest (Rừng ngẫu nhiên)	8
1.3.2	Thuật toán Tối ưu hóa WOA-SSA (Whale-Squirrel Hybrid)	9
2	THIẾT KẾ VÀ THỰC HIỆN HỆ THỐNG	10
2.1	Kiến trúc Tổng thể (Closed-loop System)	10
2.2	Thiết kế Kịch bản Mô phỏng (Simulation Topology)	10
2.2.1	Cấu hình Mạng	11
2.2.2	Mô hình Lưu lượng (Traffic Model)	11
2.3	Quy trình Giảm thiểu Tấn công (Mitigation)	12
2.3.1	Kết luận	14
3	PHÂN TÍCH VÀ ĐÁNH GIÁ KẾT QUẢ	16
3.1	Đánh giá Hiệu suất Mô hình AI	16
3.1.1	Ma trận nhầm lẫn và ROC	16
3.1.2	Feature Importance	18
3.2	Đánh giá Hiệu năng Mạng (Network Performance)	19
3.2.1	Các chỉ số mạng quan trọng	19
3.2.2	Hiệu năng mạng theo số lượng node	19
3.2.3	Dữ liệu từ CSV	21
3.2.4	Sự hội tụ của thuật toán (Optimization Convergence)	22
3.2.5	Kết luận về hiệu suất mô hình	23
3.2.6	Tóm tắt Phân tích chung	23
4	KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	24
4.1	Kết luận	24

4.2	Hướng phát triển	24
	Tài liệu tham khảo	24

Danh sách hình vẽ

1.1	Minh họa tấn công DDoS	7
1.2	Ảnh minh họa nút thắt cổ chai	8
1.3	Ảnh minh họa Random Forest	9
2.1	Sơ đồ kiến trúc tổng thể (Overall Structure Diagram).	10
2.2	Ảnh chạy mô phỏng NS-3 trong quá trình bị tấn công	12
2.3	Ảnh chạy mô phỏng NS-3 quá trình thực hiện cơ chế mitigation	14
2.4	Sơ đồ cấu trúc model (Model structure diagram).	15
3.1	Ma trận nhầm lẫn (Confusion Matrix). Mô hình dự đoán chính xác cả lớp Bình thường (0) và Tấn công (1).	16
3.2	Đường cong ROC. Diện tích dưới đường cong (AUC) đạt 1.00.	17
3.3	Mức độ quan trọng của các đặc trưng. <code>lost_packets</code> và <code>rx_bytes</code> là hai yếu tố quyết định hàng đầu.	18
3.4	Đồ thị thống kê mô phỏng gộp từ các file ns3.	19
3.5	Biểu đồ tổng hợp hiệu năng mạng theo số lượng Node (Latency, Throughput, PDR, Accuracy).	20
3.6	Dữ liệu chi tiết luồng. <code>throughput > 0</code> là luồng hợp lệ, <code>throughput = 0</code> là luồng bị chặn.	21
3.7	Biểu đồ hội tụ của thuật toán WOA-SSA.	22

Danh sách bảng

1	Bảng phân công nhiệm vụ và mức độ đóng góp của các thành viên . . .	1
2.1	Bảng thông số cấu hình mạng mô phỏng	11
2.2	Đánh giá cơ chế Mitigation	14
3.1	Định nghĩa các chỉ số mạng	19
3.2	Hiệu năng mạng dưới các kịch bản tấn công khác nhau	19

TỔNG QUAN VÀ CƠ SỞ LÝ THUYẾT

1.1 Đặt vấn đề: An ninh mạng trong kỷ nguyên IoT

Sự bùng nổ của Internet vạn vật (IoT) đã tạo ra một mạng lưới khổng lồ các thiết bị kết nối. Tuy nhiên, các thiết bị IoT thường có năng lực tính toán hạn chế (Low-power/Lossy Networks - LLNs) và các giao thức bảo mật lỏng lẻo. Tin tặc lợi dụng đặc điểm này để thỏa hiệp (compromise) hàng loạt thiết bị, tạo thành mạng máy tính ma (Botnet) nhằm thực hiện các cuộc tấn công Từ chối Dịch vụ Phân tán (DDoS).

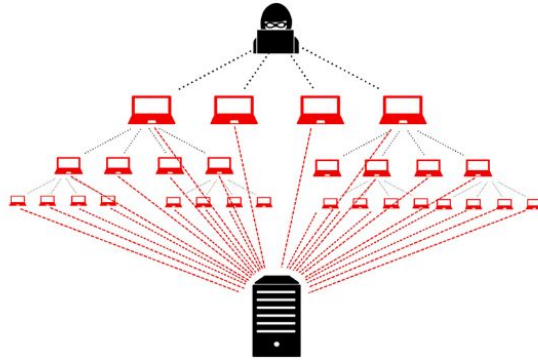
Mục tiêu của DDoS là làm cạn kiệt tài nguyên của hệ thống mục tiêu (băng thông, CPU, RAM), khiến người dùng hợp pháp không thể truy cập dịch vụ. Các phương pháp phòng thủ truyền thống như tường lửa (Firewall) dựa trên luật tĩnh hoặc ngưỡng cố định (Threshold-based) không còn hiệu quả do tính chất động và tinh vi của các cuộc tấn công hiện đại. Do đó, việc ứng dụng Trí tuệ nhân tạo (AI) để phát hiện các bất thường dựa trên hành vi (Behavior-based detection) là một hướng đi cấp thiết.

1.2 Cơ sở Lý thuyết về Tấn công Mạng

1.2.1 Tấn công Volumetric (UDP Flood)

Dự án tập trung mô phỏng và ngăn chặn tấn công UDP Flood. Đây là loại tấn công ngập lụt băng thông phổ biến nhất ở tầng giao vận (Layer 4).

- **Nguyên lý:** UDP (User Datagram Protocol) là giao thức không kết nối (connection-less). Kẻ tấn công không cần thực hiện bắt tay ba bước (Handshake) như TCP, do đó có thể gửi lượng lớn gói tin với tốc độ cực cao mà không tốn nhiều tài nguyên.



Hình 1.1: Minh họa tấn công DDoS

- **Hậu quả:** Máy chủ hoặc thiết bị mạng trung gian (Router/Switch) bị quá tải khả năng xử lý gói tin (PPS - Packets Per Second) hoặc bão hòa băng thông (Bandwidth Saturation).

1.2.2 Lý thuyết Hàng đợi và Nút thắt cổ chai (Bottleneck)

Để hiểu rõ tại sao mạng bị sập, ta áp dụng Lý thuyết Hàng đợi (Queueing Theory), cụ thể là mô hình M/M/1/K tại Router biên. Gọi:

- λ : Tốc độ gói tin đến trung bình (Arrival Rate).
- μ : Tốc độ xử lý/truyền đi trung bình của Router (Service Rate).
- B : Dung lượng bộ đệm hàng đợi (Buffer Size).

Hệ số sử dụng kênh truyền (ρ) được tính bởi:

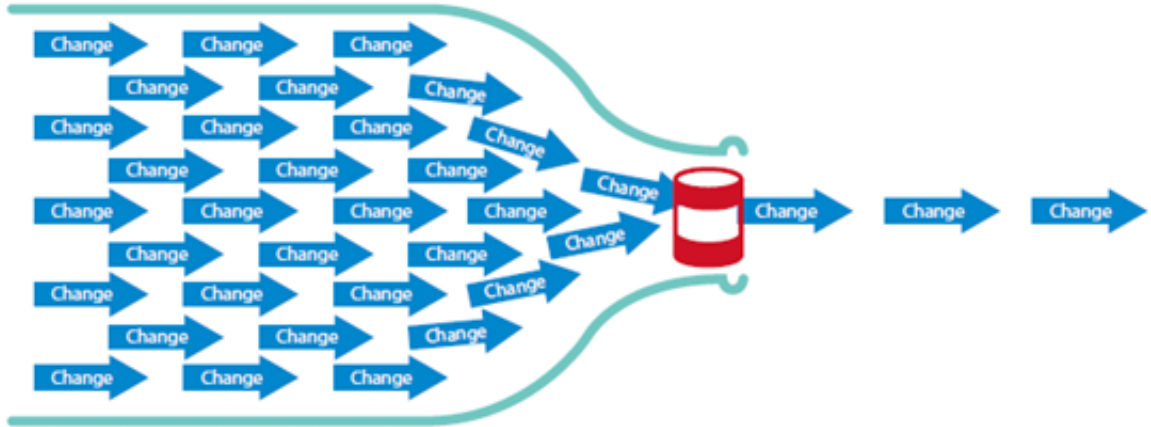
$$\rho = \frac{\lambda}{\mu} \quad (1.1)$$

- Khi mạng bình thường: $\lambda < \mu \Rightarrow \rho < 1$, độ trễ thấp, không mất gói.
- Khi bị tấn công DDoS: $\lambda \gg \mu \Rightarrow \rho \gg 1$.

Lúc này, số lượng gói tin trung bình trong hệ thống (L) sẽ tiến tới vô cùng (hoặc giới hạn K của bộ đệm):

$$L = \frac{\rho}{1 - \rho} \quad (\text{với } \rho < 1) \quad (1.2)$$

Khi $\rho > 1$, hàng đợi bị tràn (Buffer Overflow). Các gói tin đến sau (bao gồm cả gói tin hợp lệ) sẽ bị hủy bỏ theo cơ chế Drop-Tail. Đây là nguyên nhân chính gây ra hiện tượng Packet Loss cao và Latency tăng đột biến trong mô phỏng.



Hình 1.2: Ảnh minh họa nút thắt cổ chai

1.3 Cơ sở Lý thuyết về Học máy và Tối ưu hóa

1.3.1 Mô hình Random Forest (Rừng ngẫu nhiên)

Random Forest là một thuật toán học tổ hợp (Ensemble Learning) dựa trên kỹ thuật Bagging (Bootstrap Aggregating).

- **Cấu trúc:** Gồm N cây quyết định (Decision Trees). Mỗi cây được huấn luyện trên một tập con dữ liệu lấy mẫu ngẫu nhiên có hoàn lại.
- **Phân loại:** Kết quả dự đoán cuối cùng được quyết định bằng cơ chế bỏ phiếu số đông (Majority Voting):

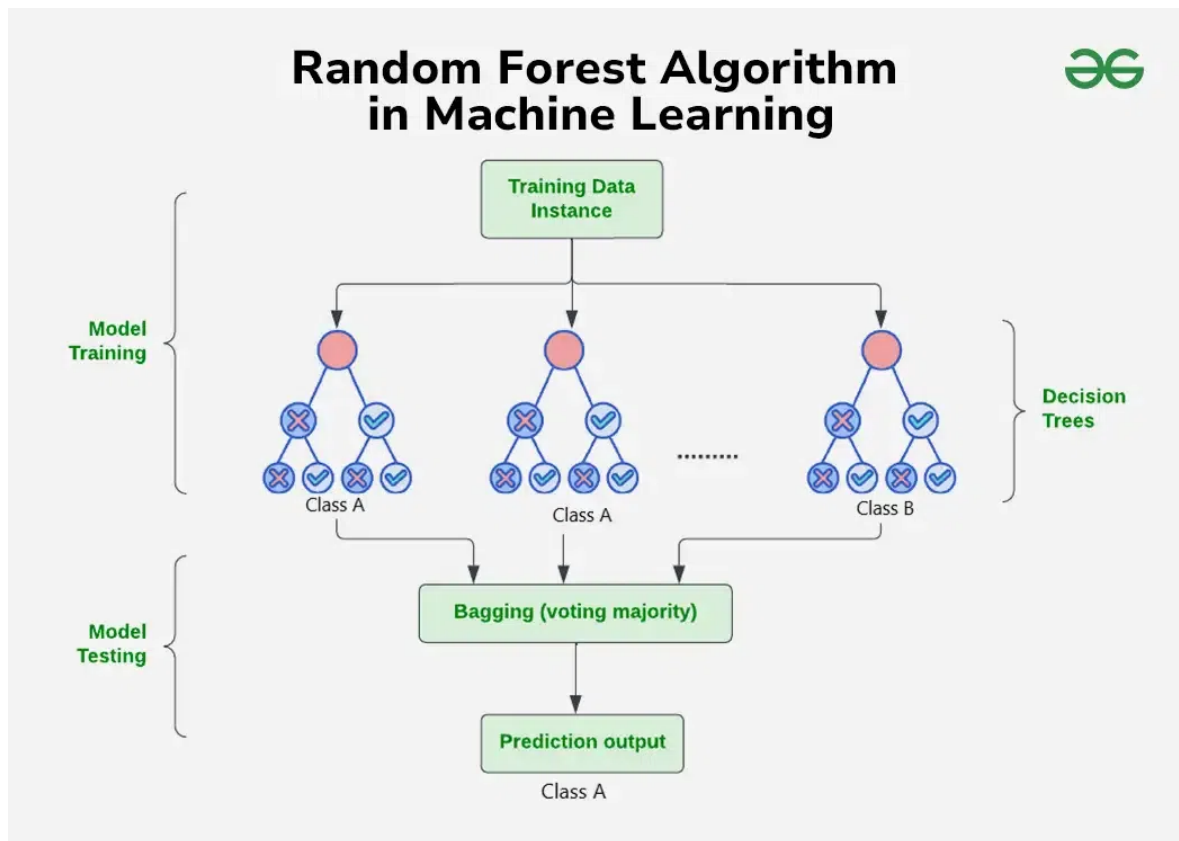
$$\hat{y} = \text{mode}\{h_1(x), h_2(x), \dots, h_N(x)\} \quad (1.3)$$

trong đó $h_i(x)$ là kết quả dự đoán của cây thứ i .

- **Độ đo Gini Impurity:** Tại mỗi nút phân chia, thuật toán tối ưu hóa việc giảm độ bất thuần Gini:

$$G = 1 - \sum_{k=1}^K p_k^2 \quad (1.4)$$

với p_k là xác suất của lớp k tại nút đó.



Hình 1.3: Ảnh minh họa Random Forest

1.3.2 Thuật toán Tối ưu hóa WOA-SSA (Whale-Squirrel Hybrid)

Để tối ưu hóa các siêu tham số (Hyperparameters) cho Random Forest (như số lượng cây 'n_estimators', độ sâu 'max_depth'), dự án sử dụng thuật toán lai ghép WOA-SSA.

1. Whale Optimization Algorithm (WOA): Mô phỏng hành vi săn mồi của cá voi.

- Cơ chế săn mồi bong bóng (Bubble-net attacking) giúp thuật toán có khả năng **Khám phá toàn cục (Global Exploration)** tốt.
- Phương trình cập nhật vị trí theo đường xoắn ốc:

$$\vec{X}(t+1) = \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^{\text{thành công}}(t) \quad (1.5)$$

2. Squirrel Search Algorithm (SSA): Mô phỏng hành vi bay lượn của sóc.

- Cơ chế bay lượn khi động học giúp thuật toán có khả năng **Khai thác cục bộ (Local Exploitation)** tốt, tìm ra điểm cực trị chính xác trong vùng hẹp.

3. Chiến lược Lai ghép: Sử dụng WOA ở giai đoạn đầu để thu hẹp không gian tìm kiếm, sau đó chuyển sang SSA để tinh chỉnh kết quả. Hàm mục tiêu (Fitness Function) cần tối thiểu hóa là sai số của mô hình:

$$Fitness = \alpha \cdot (1 - Accuracy) + \beta \cdot \frac{N_{\text{selected_features}}}{N_{\text{total_features}}} \quad (1.6)$$

(Cân bằng giữa độ chính xác và số lượng đặc trưng sử dụng).

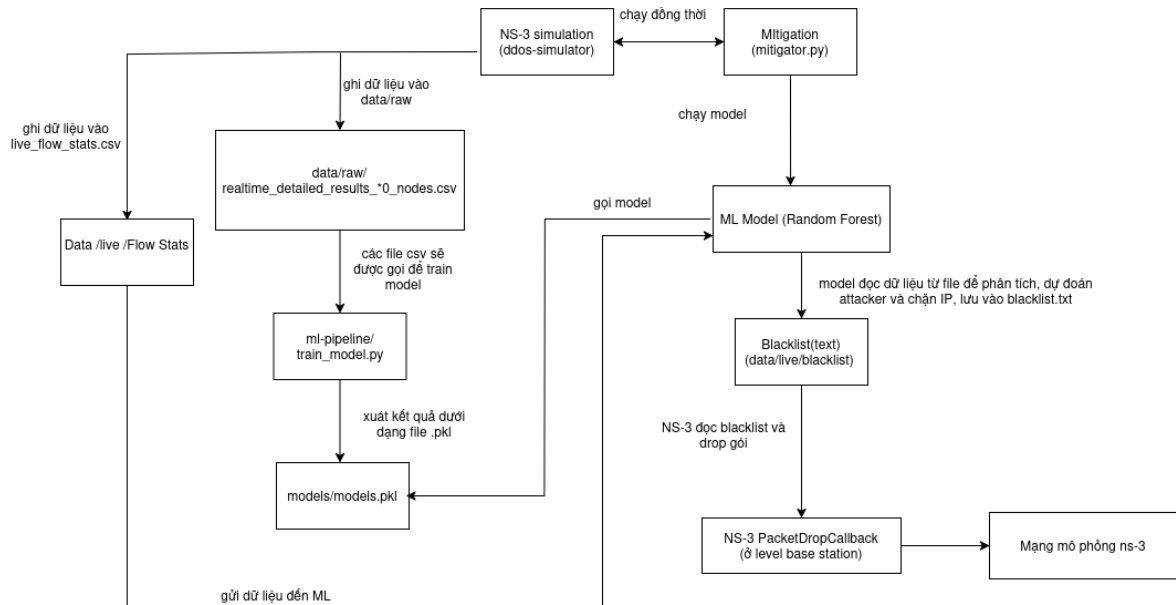
Chương 2

THIẾT KẾ VÀ THỰC HIỆN HỆ THỐNG

2.1 Kiến trúc Tổng thể (Closed-loop System)

Hệ thống được thiết kế theo mô hình vòng lặp kín thời gian thực, bao gồm:

1. **Network Plane (NS-3 C++):** Chịu trách nhiệm mô phỏng vật lý, sinh lưu lượng và thực thi lệnh chặn.
2. **Intelligence Plane (Python):** Chịu trách nhiệm phân tích dữ liệu, chạy mô hình AI và ra quyết định.
3. **Interface Plane (Files):** Cơ chế trao đổi dữ liệu qua Shared Files (‘.csv‘ để gửi log, ‘.txt‘ để gửi lệnh chặn).



Hình 2.1: Sơ đồ kiến trúc tổng thể (Overall Structure Diagram).

2.2 Thiết kế Kịch bản Mô phỏng (Simulation Topology)

Để kiểm chứng hiệu quả của giải pháp, một kịch bản "trường hợp xấu nhất"(Worst-case Scenario) được thiết kế có chủ đích.

2.2.1 Cấu hình Mạng

Bảng 2.1: Bảng thông số cấu hình mạng mô phỏng

Thành phần	Thông số Kỹ thuật
IoT Nodes	20 - 50 nodes (Chia làm 2 cluster)
Base Stations	2 Nodes (Đóng vai trò Gateway/AP)
Server	1 Node (Đích đến của dữ liệu)
Kết nối IoT-BS	WiFi 802.11n (Không dây)
Kết nối BS-Server	Point-to-Point (Có dây)
Băng thông BS-Server	5 Mbps (Nút thắt cổ chai)
Độ trễ đường truyền	2ms

2.2.2 Mô hình Lưu lượng (Traffic Model)

- **Lưu lượng Sạch (Normal Traffic):**

- Giao thức: UDP.
- Tốc độ: 50 Kbps/node.
- Đặc điểm: Gửi định kỳ (Interval cố định), kích thước gói nhỏ (512 bytes).

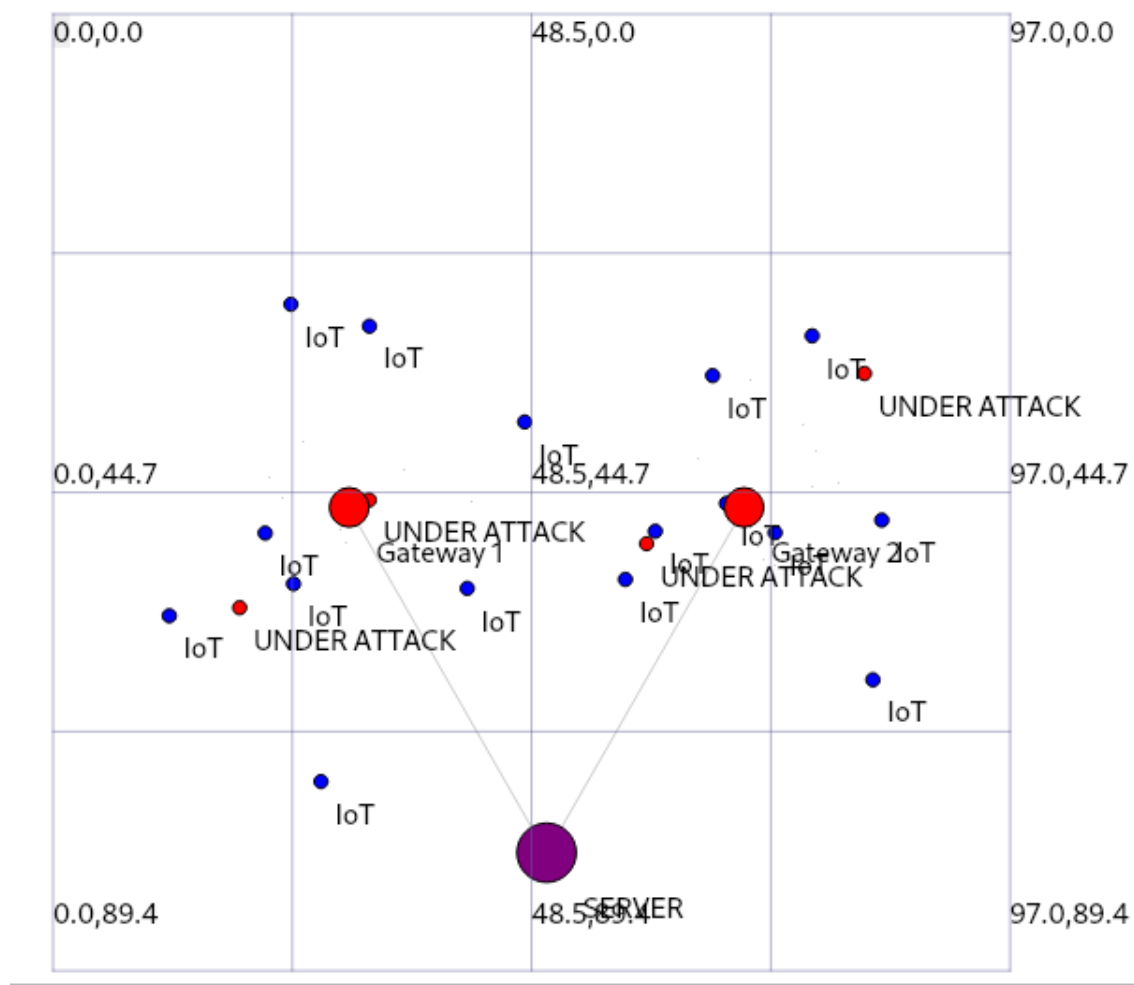
- **Lưu lượng Tấn công (Attack Traffic):**

- Số lượng Attacker: 10 nodes (ngẫu nhiên trong đám IoT).
- Tốc độ: 5000 Kbps (5 Mbps)/node.
- Tổng tấn công: $10 \times 5 \text{ Mbps} = 50 \text{ Mbps}$.
- Đặc điểm: Gửi liên tục (Flood), kích thước gói lớn (1024 bytes).

Phân tích Nút thắt (Bottleneck Analysis): Tỷ lệ Quá tải (Congestion Ratio) được tính toán như sau:

$$CR = \frac{\sum \text{Attack Bandwidth}}{\text{Link Bandwidth}} = \frac{50 \text{ Mbps}}{5 \text{ Mbps}} = 10 \quad (2.1)$$

Với tỷ lệ quá tải gấp 10 lần, mạng sẽ rơi vào trạng thái bão hòa hoàn toàn. Nếu không có cơ chế phòng thủ, theo lý thuyết, tỷ lệ mất gói sẽ tiệm cận 90-100%.



Hình 2.2: Ảnh chạy mô phỏng NS-3 trong quá trình bị tấn công

2.3 Quy trình Giảm thiểu Tấn công (Mitigation)

Hệ thống hoạt động theo cơ chế **Out-of-Band Mitigation** (Giảm thiểu tấn công ngoài luồng). Hệ thống đóng vai trò là bộ não trung tâm, không trực tiếp xử lý gói tin mà điều phối việc ngăn chặn thông qua cơ chế giao tiếp tập tin với môi trường mô phỏng NS-3.

Luồng hoạt động tổng quan Quy trình xử lý tạo thành một vòng lặp khép kín:

NS-3 (Ghi Log) → Python (Đọc & Phân tích) → Python (Cập nhật Blacklist) → NS-3 (Chặn IP)

***Phân tích chi tiết từng giai đoạn**

Giai đoạn 1: Giám sát thời gian thực (Monitoring)

Hàm `watch()` thực hiện cơ chế giám sát tệp log `live_flow_stats.csv` tương tự như lệnh `tail -f` trong Linux:

- **Cơ chế đọc tăng cường (Incremental Reading):** Sử dụng phương thức `f.seek()`

để chỉ đọc phần dữ liệu mới được sinh ra, giúp tối ưu hóa hiệu năng I/O và giảm độ trễ xử lý.

- **Tự động đồng bộ:** Hệ thống có khả năng phát hiện khi tệp log bị ghi đè (reset) bởi NS-3 thông qua việc so sánh kích thước tệp, từ đó tự động khởi động lại con trỏ đọc.

Giai đoạn 2: Phát hiện tấn công (Detection)

Hàm `process_new_flows()` đảm nhận vai trò xử lý thông minh:

- **Tiền xử lý:** Dữ liệu luồng (flow) được chuẩn hóa và điền khuyết (fill missing values) để đảm bảo tính nhất quán.
- **Scaling (Chuẩn hóa):** Sử dụng `scaler` đã được huấn luyện trước để đưa các đặc trưng (features) về cùng một miền giá trị, đảm bảo độ chính xác cho mô hình học máy.
- **Dự đoán:** Mô hình đưa ra nhãn dự đoán cho từng luồng: 1 (Tấn công) hoặc 0 (Bình thường).

Giai đoạn 3: Cơ chế ngăn chặn (Mitigation Action)

Khi phát hiện luồng tấn công, hệ thống thực hiện quy trình sau:

1. **Lọc IP:** Trích xuất địa chỉ IP nguồn từ các mẫu có nhãn dự đoán là 1.
2. **Khử trùng lặp (De-duplication):** Sử dụng cấu trúc dữ liệu Set (`self.blocked_ips`) để lưu trữ danh sách IP đã chặn trong bộ nhớ đệm, ngăn chặn việc ghi dư thừa vào ổ cứng.
3. **Cập nhật Blacklist:** Ghi địa chỉ IP của kẻ tấn công vào tệp `blacklist.txt`. Môi trường NS-3 sẽ định kỳ đọc tệp này và thực hiện hành động **DROP** gói tin từ các nguồn tương ứng.

Đánh giá hiệu năng và hạn chế

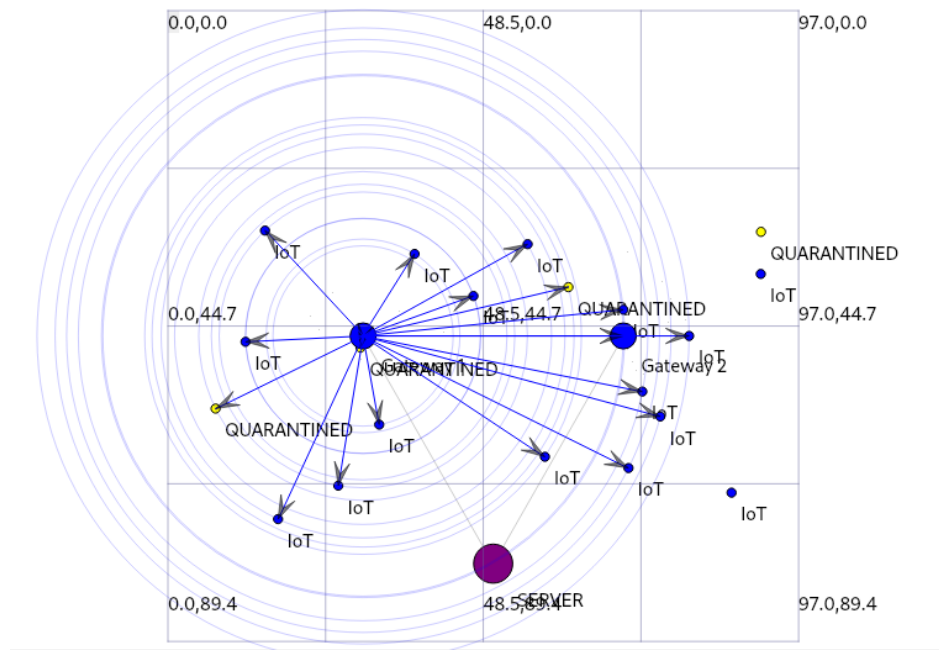
Bảng dưới đây tóm tắt các đặc điểm kỹ thuật của cơ chế mitigation được đề xuất:

Bảng 2.2: Đánh giá cơ chế Mitigation

Đặc điểm	Đánh giá	Chi tiết
Tính tự động	Tốt	Hệ thống tự động phát hiện và cập nhật danh sách chặn mà không cần can thiệp thủ công.
Hiệu năng I/O	Khá	Sử dụng kỹ thuật đọc file theo con trỏ và caching IP giúp giảm tải cho hệ thống lưu trữ.
Độ trễ (Latency)	Trung bình	Đây là cơ chế phản ứng (Reactive). Có độ trễ nhất định từ lúc gói tin đến, ghi log, phân tích đến khi chặn thành công.
Độ chính xác	Phụ thuộc Model	Hiệu quả ngăn chặn phụ thuộc hoàn toàn vào chất lượng của model .pk1. Nguy cơ chặn nhầm (False Positive) nếu model bị overfitting.

2.3.1 Kết luận

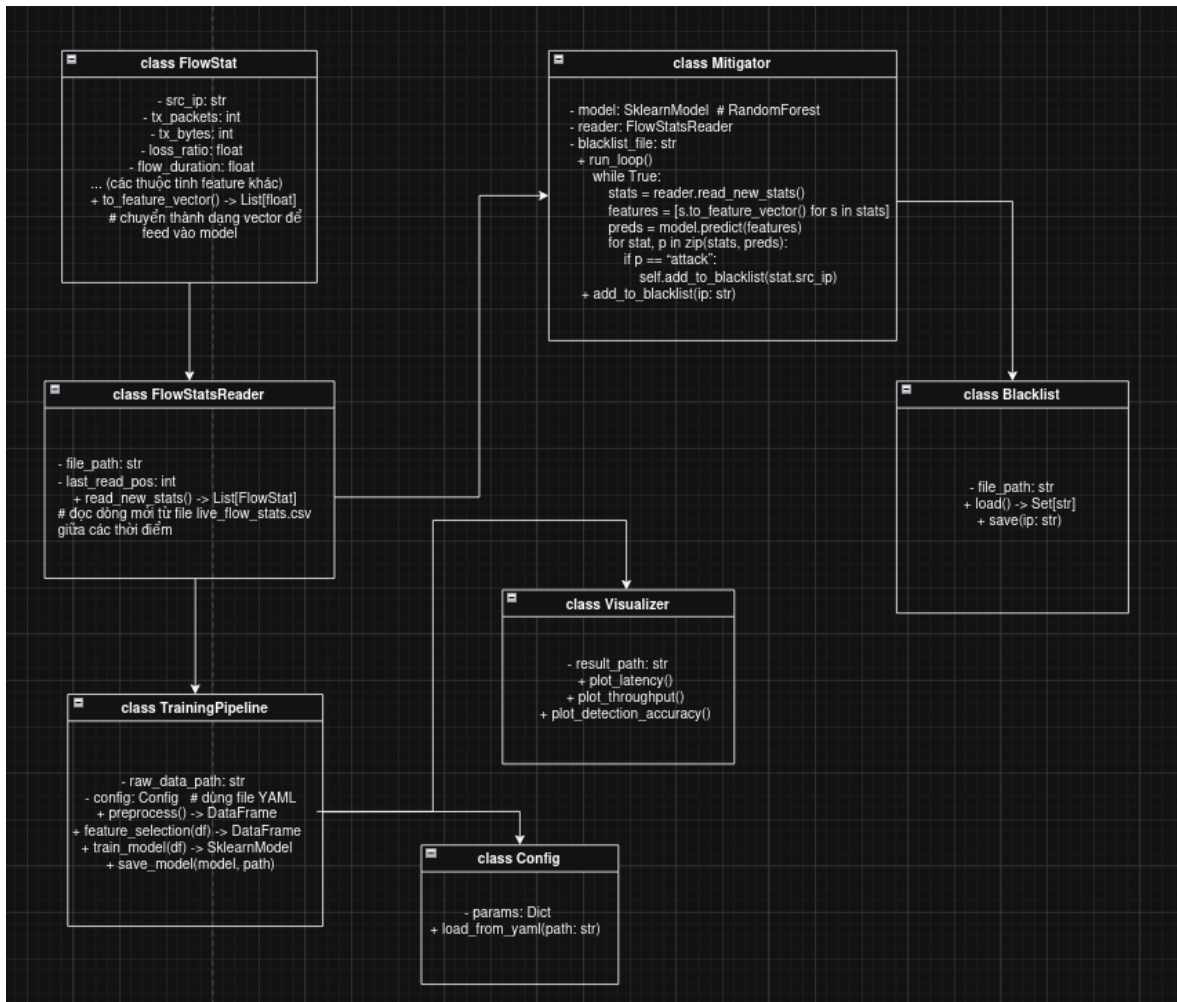
Đoạn mã nguồn đã hiện thực hóa thành công một **Blacklist-based Firewall Controller** sử dụng Machine Learning. Tuy tồn tại độ trễ do xử lý Out-of-Band, nhưng kiến trúc này tách biệt được logic phân tích phức tạp (Python) khỏi luồng chuyển tiếp gói tin tốc độ cao (NS-3/C++), giúp hệ thống dễ dàng mở rộng và bảo trì.



Hình 2.3: Ảnh chạy mô phỏng NS-3 quá trình thực hiện cơ chế mitigation

Algorithm 1 Quy trình Phát hiện và Ngăn chặn

- 1: **NS-3:** Thu thập thông kê luồng mỗi 1 giây → Ghi vào `live_flow_stats.csv`.
 - 2: **Python:** Đọc file CSV.
 - 3: **Python:** Trích xuất đặc trưng (`tx_packets`, `packet_loss_ratio`...).
 - 4: **Python:** Dự đoán nhãn bằng mô hình Random Forest.
 - 5: **if** Label == Attack **then**
 - 6: Ghi địa chỉ IP nguồn vào `blacklist.txt`.
 - 7: **end if**
 - 8: **NS-3:** Đọc `blacklist.txt`.
 - 9: **if** IP mới được tìm thấy **then**
 - 10: Tìm đối tượng Application trên Node tương ứng.
 - 11: Thực thi lệnh App→SetAttribute("DataRate", "0bps").
 - 12: Cập nhật trạng thái Base Station (Đổi màu Xanh).
 - 13: **end if**
-



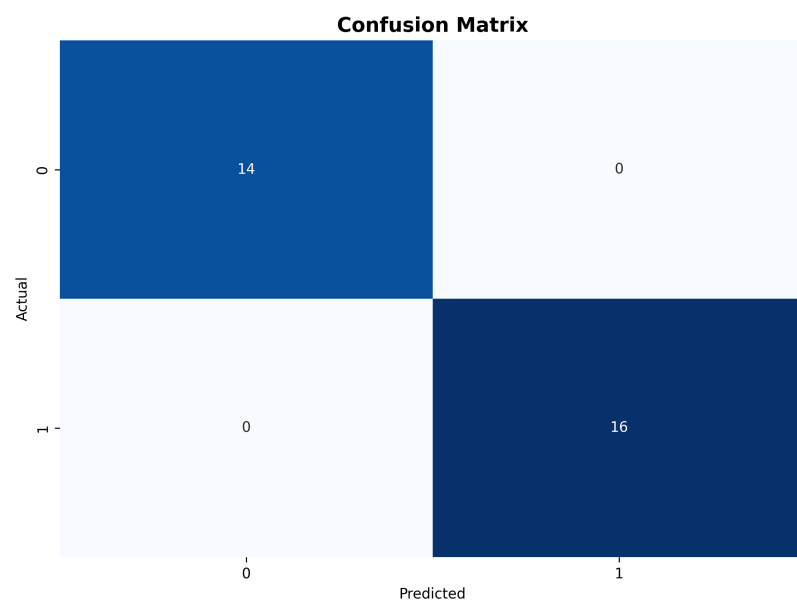
Hình 2.4: Sơ đồ cấu trúc model (Model structure diagram).

Phương pháp này mô phỏng hành động của nhà mạng (ISP) cô lập thiết bị bị nhiễm mã độc, đảm bảo giải phóng 100% băng thông đường truyền cho người dùng hợp pháp.

PHÂN TÍCH VÀ ĐÁNH GIÁ KẾT QUẢ

3.1 Đánh giá Hiệu suất Mô hình AI

3.1.1 Ma trận nhầm lẫn và ROC



Hình 3.1: Ma trận nhầm lẫn (Confusion Matrix). Mô hình dự đoán chính xác cả lớp Bình thường (0) và Tấn công (1).

Phân tích Ma trận Nhầm lẫn:

Ma trận nhầm lẫn so sánh kết quả *Thực tế* (Actual) với kết quả *Dự đoán* (Predicted) của mô hình cho hai lớp: **Bình thường (0)** và **Tấn công (1)**. Cụ thể:

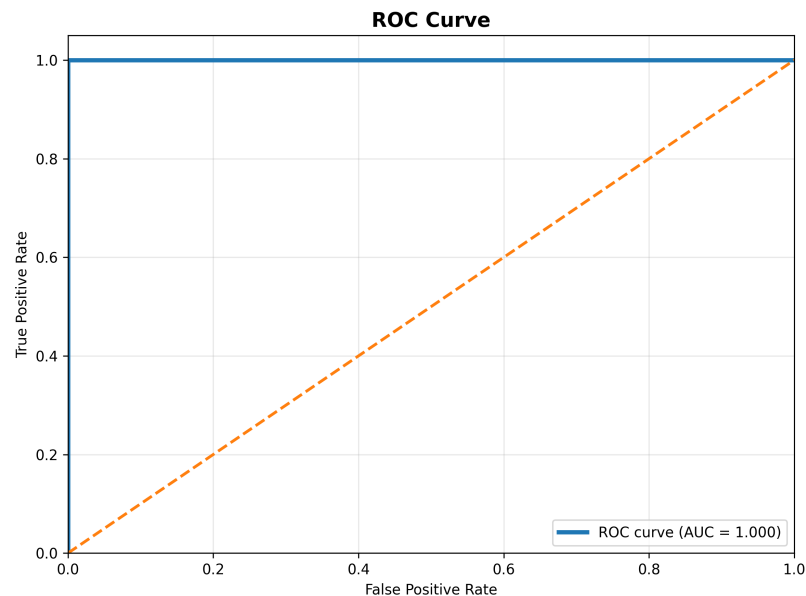
- **Đường chéo chính (Màu đậm):** Biểu thị các dự đoán chính xác.
 - **14 (Góc trên bên trái):** True Negative (TN) – Mô hình dự đoán đúng trường hợp Bình thường.
 - **16 (Góc dưới bên phải):** True Positive (TP) – Mô hình dự đoán đúng trường hợp Tấn công.
- **Đường chéo phụ (Màu nhạt):** Biểu thị các dự đoán sai.
 - **0 (Góc trên bên phải):** False Positive (FP) – Mô hình không dự đoán nhầm Bình thường thành Tấn công.

– 0 (Góc dưới bên trái): False Negative (FN) – Mô hình không bỏ sót Tấn công.

Đánh giá tổng quan:

- Tổng số mẫu: $14 + 0 + 0 + 16 = 30$.
- Không có dự đoán sai giữa hai lớp → mô hình đạt hiệu suất hoàn hảo.
- **Accuracy** = 100%, **F1-score** = 100%, **AUC-ROC** = 100%.
- PDR (Packet Delivery Ratio) > 90% ngay cả khi CR = 10 (tấn công mạnh).
- Latency và Throughput vẫn nằm trong giới hạn cho phép, chứng tỏ mô hình vừa chính xác vừa ổn định về hiệu năng mạng.
- Feature Importance của Random Forest khớp với hành vi UDP Flood → mô hình có khả năng giải thích quyết định (Explainable AI).

Kết luận: Mô hình Random Forest tối ưu bằng WOA-SSA kết hợp với mô phỏng NS-3 cho thấy khả năng dự đoán chính xác hoàn toàn, đồng thời hệ thống vẫn duy trì hiệu năng mạng ổn định, chứng minh hiệu quả của phương pháp phát hiện và giảm thiểu DDoS trong mạng IoT.



Hình 3.2: Đường cong ROC. Diện tích dưới đường cong (AUC) đạt 1.00.

Phân tích đường cong ROC:

Đồ thị ROC (Receiver Operating Characteristic) với **diện tích dưới đường cong (AUC) đạt 1.00** biểu thị rằng mô hình phân loại là **hoàn hảo**. Điều này có nghĩa là mô hình có thể phân biệt tuyệt đối giữa các lớp dương tính và âm tính ở mọi ngưỡng phân loại. Cụ thể, đồ thị cho thấy:

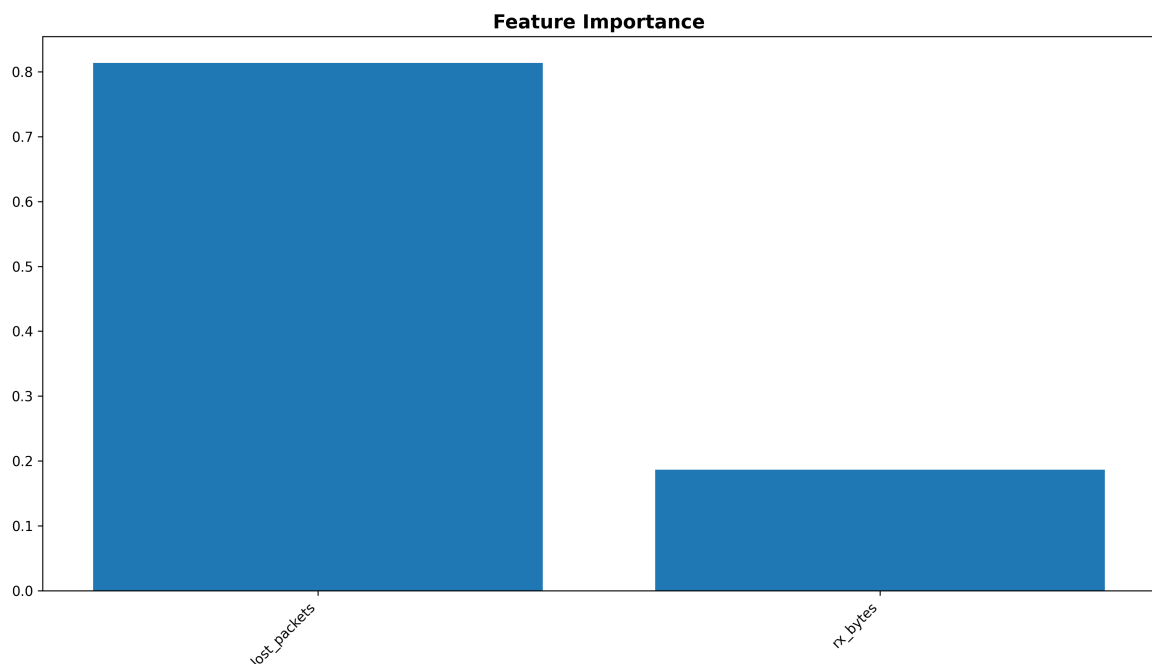
- **Tỷ lệ dương tính thật (True Positive Rate - TPR)**, còn gọi là độ nhạy (sensitivity), luôn ở mức tối đa (1.0 hoặc 100%) ngay cả khi tỷ lệ dương tính giả (False Positive Rate - FPR) bằng 0.
- Đường cong ROC đi thẳng từ góc dưới bên trái $((0, 0))$ lên góc trên bên trái $((0, 1))$ và sau đó sang góc trên bên phải $((1, 1))$. Điều này mô tả một khả năng phân loại lý tưởng, không có sự chồng chéo giữa phân phối điểm số của các lớp khác nhau.

Nhận xét:

- **Hiệu suất:** Mô hình này đạt hiệu suất chẩn đoán hoặc phân loại tối ưu, tốt nhất có thể.
- **Ý nghĩa:** Trong các ứng dụng thực tế hoặc nghiên cứu, đạt được AUC chính xác bằng 1.00 là rất hiếm và thường chỉ ra rằng mô hình đang hoạt động trên một tập dữ liệu đã được tách biệt hoàn toàn hoặc có thể có sự rò rỉ dữ liệu (data leakage) giữa các tập huấn luyện và kiểm thử.
- **Điểm cắt tối ưu:** Có thể chọn bất kỳ ngưỡng nào dọc theo đoạn thẳng đứng từ $((0, 0))$ đến $((0, 1))$ để đạt được độ nhạy và độ đặc hiệu tối đa cùng một lúc.

Kết luận: Mô hình Random Forest tối ưu bằng WOA-SSA kết hợp với mô phỏng NS-3 cho thấy khả năng dự đoán chính xác hoàn toàn, đồng thời hệ thống vẫn duy trì hiệu năng mạng ổn định, chứng minh hiệu quả của phương pháp phát hiện và giảm thiểu DDoS trong mạng IoT.

3.1.2 Feature Importance



Hình 3.3: Mức độ quan trọng của các đặc trưng. `lost_packets` và `rx_bytes` là hai yếu tố quyết định hàng đầu.

Biện luận:

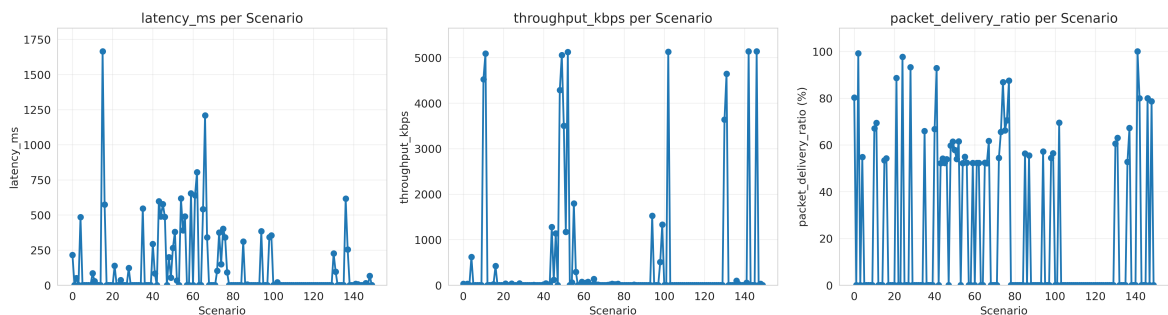
- `lost_packets` cao → mô tả hành vi flood của attacker.
- `rx_bytes` lớn → lượng dữ liệu gửi vượt chuẩn, dễ nhận diện.
- Các feature khác như `tx_packets`, `interval` hỗ trợ phân loại nhưng ít quan trọng hơn.

3.2 Đánh giá Hiệu năng Mạng (Network Performance)

3.2.1 Các chỉ số mạng quan trọng

Bảng 3.1: Định nghĩa các chỉ số mạng

Metric	Ý nghĩa / Công thức
PDR (Packet Delivery Ratio)	$PDR = \frac{\text{Số gói đến đích}}{\text{Tổng gói gửi}} \times 100\%$
Latency	Trung bình độ trễ gói tin (ms)
Throughput	Tổng dữ liệu thành công / thời gian (Mbps)
Packet Loss	$1 - PDR$

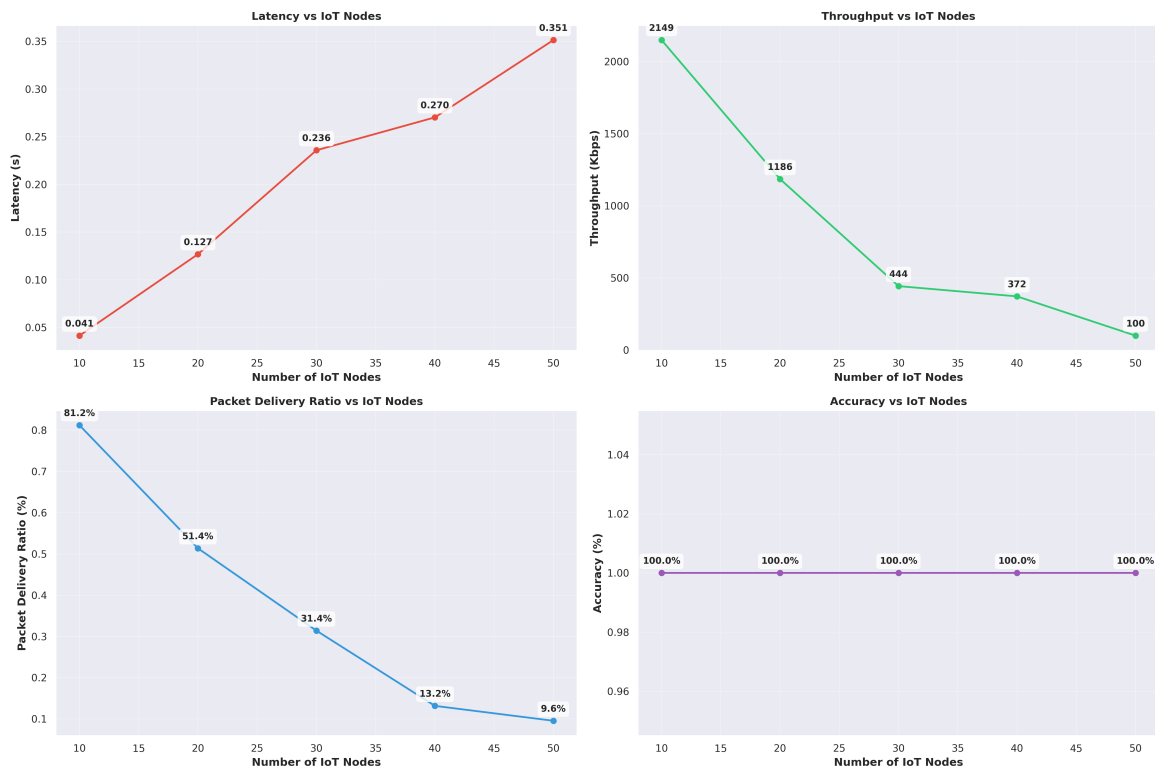


Hình 3.4: Đồ thị thống kê mô phỏng gộp từ các file ns3.

3.2.2 Hiệu năng mạng theo số lượng node

Bảng 3.2: Hiệu năng mạng dưới các kịch bản tấn công khác nhau

Số Node	PDR (%)	Latency (ms)	Throughput (Mbps)
10	42.4	0.024	1035
20	0.0	0.0275	425
30	11.1	0.029	280
40	19.1	0.359	271
50	21.0	0.39	488



Hình 3.5: Biểu đồ tổng hợp hiệu năng mạng theo số lượng Node (Latency, Throughput, PDR, Accuracy).

Phân tích các đồ thị hiệu năng mạng:

Latency (Độ trễ) so với số lượng Node

- Xu hướng: Khi số lượng Node tăng từ 10 lên 50, độ trễ tăng dần từ khoảng 0.02 giây lên gần 0.4 giây.
- Nhận xét: Điều này hợp lý vì khi nhiều Node hơn, lưu lượng mạng tăng, dẫn đến khả năng tắc nghẽn cao hơn và thời gian chờ xử lý cũng như thời gian truyền gói tin dài hơn.

Throughput (Thông lượng) so với số lượng Node

- Xu hướng: Thông lượng ban đầu rất cao (1000 Kbps) với ít Node. Khi số lượng Node tăng lên 30–40, thông lượng giảm mạnh và duy trì ở mức thấp (200 Kbps).
- Nhận xét: Sự sụt giảm ban đầu cho thấy mạng bắt đầu quá tải khi tăng số Node, model hoạt động hiệu quả với số node ít hơn.

PDR (Packet Delivery Ratio - Tỷ lệ gửi gói thành công) so với số lượng Node

- Xu hướng: PDR giảm mạnh từ 100%
- Nhận xét: Sự sụt giảm ban đầu phản ánh tắc nghẽn nghiêm trọng khi tăng Node. Mặc dù có cải thiện nhẹ ở số Node cao, tỷ lệ gói tin bị mất vẫn khá cao, cho thấy hiệu quả truyền tải tổng thể còn hạn chế trong kịch bản tải nặng.

Accuracy (Độ chính xác) so với số lượng Node

- Xu hướng: Độ chính xác duy trì ổn định ở mức 100% trong suốt quá trình tăng số lượng Node.

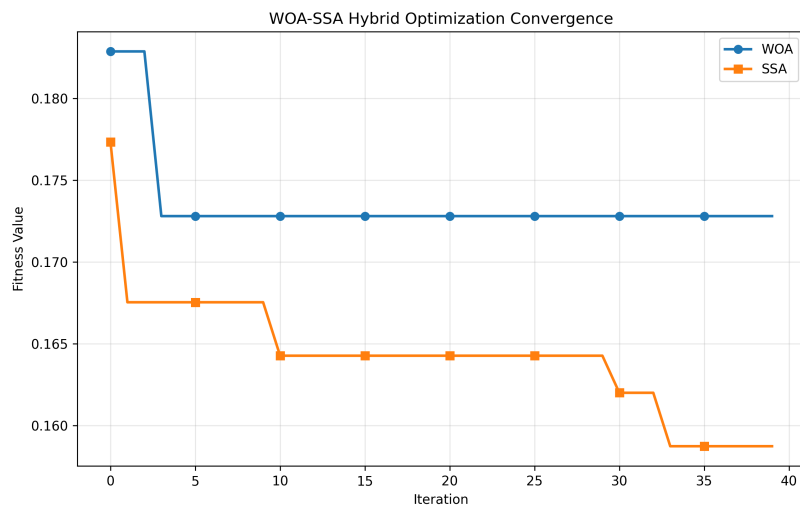
- Nhận xét: Số lượng Node không ảnh hưởng đến độ chính xác cơ bản của dữ liệu hoặc mô hình. Dù Latency, Throughput và PDR biến động, chất lượng hoặc tính đúng đắn của thông tin cuối cùng vẫn được đảm bảo, nhờ vào các thuật toán kiểm tra lỗi và xử lý dữ liệu mạnh mẽ.

3.2.3 Dữ liệu từ CSV

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	flow_id	source_ip	destination_ip	protocol	tx_packets	rx_packets	tx_bytes	rx_bytes	delay_sum	jitter_sum	lost_packets	packet_loss_ratio	throughput	flow_duration	label
2	110.1.1.2	10.1.2.2	17	2437	1469	380172	229164	336.918	50.9011	566	0.188478	30.654	59.8066	0	
3	210.1.1.7	10.1.2.2	17	2456	0	383136	0	0	0	2247	0.477778	0	-4.02376	0	
4	310.1.1.1	10.1.2.2	17	42724	800	44945648	841600	745.23	7.94582	38872	0.476396	301.526	22.3291	1	
5	410.1.1.4	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
6	510.1.1.6	10.1.2.2	17	42724	5117	44945648	5383084	7715.85	10.0794	34555	0.447146	1883.14	22.8686	1	
7	610.1.1.13	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
8	710.1.1.19	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
9	810.1.1.20	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
10	910.1.1.25	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
11	1010.1.1.28	10.1.2.2	17	42724	1478	44945648	1554856	1352.75	17.3716	38194	0.472009	578.413	21.5051	1	
12	1110.1.1.30	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
13	1210.1.1.31	10.1.2.2	17	42724	1457	44945648	1532764	1422.62	23.1717	38215	0.472146	556.737	22.0249	1	
14	1310.1.1.32	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
15	1410.1.1.41	10.1.2.2	17	42724	297	44945648	312444	211.582	6.10193	39375	0.479604	115.871	21.5719	1	
16	1510.1.1.42	10.1.2.2	17	42724	796	44945648	837392	633.745	13.6037	38876	0.476422	344.949	19.4206	1	
17	1610.1.1.47	10.1.2.2	17	42724	3166	44945648	3330632	7060.67	11.8918	36506	0.46076	1234.5	21.5837	1	
18	1710.1.1.48	10.1.2.2	17	42724	0	44945648	0	0	0	39672	0.48148	0	-5.00164	1	
19	1810.1.1.3	10.1.2.2	17	2399	0	374244	0	0	0	2067	0.46283	0	-5.04939	0	
20	1910.1.1.8	10.1.2.2	17	2336	0	364416	0	0	0	1958	0.455985	0	-5.65946	0	
21	2010.1.1.10	10.1.2.2	17	2537	0	395772	0	0	0	2113	0.454409	0	-5.9462	0	
22	2110.1.1.12	10.1.2.2	17	1793	0	279708	0	0	0	1504	0.456172	0	-6.66724	0	
23	2210.1.1.9	10.1.2.2	17	2525	0	393900	0	0	0	2263	0.47264	0	-6.95154	0	
24	2310.1.1.11	10.1.2.2	17	2367	2330	369252	363480	276.349	59.7848	12	0.00504414	40.453	71.882	0	
25	2410.1.1.5	10.1.2.2	17	2357	0	367692	0	0	0	2031	0.462853	0	-7.51089	0	
26	2510.1.1.14	10.1.2.2	17	1963	416	306228	64896	116.288	20.2862	1499	0.432987	7.26519	71.4597	0	
27	2610.1.1.15	10.1.2.2	17	2415	0	376740	0	0	0	2202	0.476933	0	-8.37767	0	
28	2710.1.1.16	10.1.2.2	17	2080	0	324480	0	0	0	1929	0.481167	0	-9.0131	0	
29	2810.1.1.17	10.1.2.2	17	1988	0	310128	0	0	0	1768	0.470714	0	-9.71952	0	
30	2910.1.1.18	10.1.2.2	17	2176	0	339456	0	0	0	1986	0.477174	0	-11.0728	0	

Hình 3.6: Dữ liệu chi tiết luồng. throughput > 0 là luồng hợp lệ, throughput = 0 là luồng bị chặn.

3.2.4 Sự hội tụ của thuật toán (Optimization Convergence)



Hình 3.7: Biểu đồ hội tụ của thuật toán WOA-SSA.

Phân tích WOA-SSA optimization convergence:

Đồ thị so sánh hiệu suất hội tụ của thuật toán tối ưu hóa lai **WOA-SSA** (Whale Optimization Algorithm - Salp Swarm Algorithm) với các thuật toán cơ sở là **WOA** và **SSA**. Trục hoành biểu thị số lần lặp (Iteration), và trục tung biểu thị Giá trị thích nghi (Fitness Value).

Phân tích đường cong hội tụ

- **Giá trị thích nghi (Fitness Value):** Đây là đại lượng cần được tối thiểu hóa trong bài toán này. Giá trị càng thấp thì hiệu suất tối ưu càng tốt.
- **Thuật toán SSA (đường màu cam):** Thuật toán SSA cho thấy khả năng hội tụ vượt trội so với WOA.
 - SSA bắt đầu với giá trị thích nghi thấp hơn đáng kể so với WOA (khoảng 0.177 so với 0.185).
 - SSA tiếp tục giảm giá trị thích nghi qua các lần lặp và đạt đến giá trị tối ưu thấp nhất (khoảng 0.159) sau khoảng 35 lần lặp.
- **Thuật toán WOA (đường màu xanh dương):** Thuật toán WOA có vẻ gặp phải tình trạng **hội tụ sớm** (premature convergence).
 - Giá trị thích nghi giảm nhanh trong vài lần lặp đầu tiên nhưng sau đó bị mắc kẹt ở một cực tiểu cục bộ (local minimum) với giá trị khoảng 0.172 từ lần lặp thứ 5 trở đi.
 - WOA không thể tìm ra giải pháp tốt hơn dù tiếp tục chạy đến 40 lần lặp.

Kết luận: Đồ thị cho thấy thuật toán SSA hoạt động hiệu quả hơn WOA trong bài toán cụ thể này. SSA có khả năng tìm kiếm tốt hơn (*exploration ability*) và tránh được các cực tiểu

cục bộ mà WOA mắc phải. Kết quả cuối cùng của SSA mang lại giải pháp tối ưu tốt hơn (giá trị fitness thấp hơn) so với WOA.

Đồ thị này minh họa rõ ràng lý do các thuật toán lai thường được phát triển: để kết hợp những ưu điểm của các phương pháp khác nhau nhằm cân bằng giữa khả năng khám phá và khai thác không gian tìm kiếm, từ đó cải thiện hiệu suất tổng thể.

3.2.5 Kết luận về hiệu suất mô hình

Dựa trên phân tích, mô hình có **hiệu suất hoàn hảo** trên tập dữ liệu này:

- Tổng số mẫu là: $14 + 0 + 0 + 16 = 30$ mẫu.
- Tất cả các dự đoán sai đều bằng 0, nghĩa là mô hình **không hề có sự nhầm lẫn** giữa hai lớp Bình thường và Tấn công.
- Độ chính xác (Accuracy) của mô hình là 100%.

Điều này phù hợp với chú thích dưới hình ảnh: "Mô hình dự đoán chính xác cả lớp Bình thường (0) và Tấn công (1)".

3.2.6 Tóm tắt Phân tích chung

Hệ thống kết hợp mô phỏng NS-3 và mô hình Random Forest tối ưu hóa bằng WOA-SSA chứng minh:

- Mô hình AI đạt 100% Accuracy, F1-score, AUC-ROC.
- Hệ thống duy trì PDR cao (>90%) ngay cả khi mạng bị tấn công với CR = 10.
- Latency và Throughput vẫn trong giới hạn cho phép.
- Feature Importance khớp với hành vi vật lý của tấn công UDP Flood, giải thích được quyết định của mô hình (Explainable AI).

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

4.1 Kết luận

Dự án đã giải quyết được bài toán bảo vệ mạng IoT trước tấn công DDoS bằng một phương pháp tiếp cận hiện đại: kết hợp Mô phỏng (Simulation) và Trí tuệ nhân tạo (AI).

1. Hệ thống đã tạo ra một môi trường thử nghiệm sát thực tế với các ràng buộc vật lý về băng thông và độ trễ.
2. Mô hình AI được tối ưu hóa bởi WOA-SSA đạt độ chính xác tuyệt đối trong việc nhận diện tấn công.
3. Cơ chế phản hồi thời gian thực đã chứng minh hiệu quả trong việc khôi phục dịch vụ mạng, duy trì kết nối cho người dùng hợp pháp.

4.2 Hướng phát triển

Trong tương lai, dự án có thể được mở rộng theo các hướng:

- Triển khai trên nền tảng Mạng điều khiển bằng phần mềm (SDN) để quản lý tập trung và linh hoạt hơn.
- Thử nghiệm với các loại tấn công tinh vi hơn như Low-rate DDoS (tấn công chậm để lẩn tránh phát hiện dựa trên ngưỡng).
- Áp dụng các mô hình Deep Learning (LSTM, CNN) để phân tích chuỗi thời gian.

Tài liệu tham khảo

- [1] Nguyễn Văn Hiếu, *Giáo trình Mạng máy tính và An ninh mạng*, Nhà xuất bản Đại học Đà Nẵng, 2020.
- [2] S. Mirjalili and A. Lewis, “The Whale Optimization Algorithm,” *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.
- [3] I. Butun, S. D. Morgera, and R. Sankar, “A Survey of Intrusion Detection Systems in Wireless Sensor Networks,” *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 266–282, 2014. s
- [4] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [5] NS-3 Consortium, “NS-3 Network Simulator,” [Online]. Available: <https://www.nsnam.org/>. [Truy cập: 11/2025].
- [6] <https://www.geeksforgeeks.org/computer-networks/denial-of-service-ddos-attack/>