**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN - TRUYỀN THÔNG**

A blue diamond shaped sign with a torch and a yellow circle

AI-generated content may be incorrect.

**NIÊN LUẬN NGÀNH**

**MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

**Đề tài:**

**MALWARE DETECTION USING**

**MACHINE LEARNING**

**Cán bộ hướng dẫn**

TS.GVC. Phan Thượng Cang

**Sinh viên thực hiện**

Quách Minh Kỳ B2110942

Dương Thị Kiều Trâm B2110953

Nguyễn Gia Bảo B2110960

Phạm Thế Vinh B2110982

**HỌC KỲ 2, NĂM HỌC 2024 – 2025**

**TỰ ĐÁNH GIÁ KẾT QUẢ THỰC HIỆN NIÊN LUẬN MMT&TT**

(Học kỳ 2, năm học 2024 - 2025)

**TÊN ĐỀ TÀI**: Malware Detection Using Machine Learning

**GIÁO VIÊN HƯỚNG DẪN:**

|  |  |  |
| --- | --- | --- |
| STT | HỌ VÀ TÊN | MSCB |
| 1 | PHAN THƯỢNG CANG | 001230 |

**SINH VIÊN THỰC HIỆN:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | HỌ VÀ TÊN | MSSV | THƯỞNG  *(Tối đa 1 điểm)* | ĐIỂM |
| 1 | QUÁCH MINH KỲ | B2110942 |  |  |
| 2 | DƯƠNG THỊ KIỀU TRÂM | B2110953 |  |  |
| 3 | NGUYỄN GIA BẢO | B2110960 |  |  |
| 4 | PHẠM THẾ VINH | B2110982 |  |  |

|  |  |
| --- | --- |
| 1. **HÌNH THỨC** *(Tối đa 0,5 điểm)* |  |
| **Bìa** *(Tối đa 0,25 điểm)* |  |
| * Các tiêu đề: Trường ĐHCT, Khoa CNTT * Loại niên luận: 3, Tên đề tài * Giáo viên hướng dẫn: chức danh, họ tên * Thông tin về các sinh viên thực hiện: họ tên, mã số, lớp * Năm thực hiện |  |
| **Bố cục** *(Tối đa 0,25 điểm)* |  |
| * Nhận xét của giáo viên hướng dẫn và giáo viên chấm * Mục lục: cấu trúc chương, mục và tiểu mục * Phụ lục (nếu có) * Tài liệu tham khảo |  |
| 1. **NỘI DUNG (Tối đa 5 điểm)** |  |
| **Giới thiệu** *(Tối đa 0,5 điểm)* |  |
| * Mô tả bài toán * Mục tiêu cần đạt, hướng giải quyết |  |
| **Lý thuyết** *(Tối đa 1,0 điểm)* |  |
| * Các khái niệm * Các phương pháp * Kết quả nghiên cứu lý thuyết vận dụng vào đề tài |  |
| **Ứng dụng** *(Tối đa 3,0 điểm)* |  |
| * Đặc tả hệ thống (0.5 điểm) * Các mô hình (MCD, MLD, PDM) và các ràng buộc toàn vẹn (1 điểm) * DFD, Sơ đồ chức năng, lưu đồ giải thuật giải quyết vấn đề (nếu có), diễn giải ý nghĩa các màn hình cập nhật (FORM) và các báo biểu (REPORT) (1 điểm) * Giới thiệu chương trình (0,5 điểm) |  |
| **Kết luận** *(Tối đa 0,5 điểm)* |  |
| * Nhận xét kết quả đạt được * Hạn chế * Hướng phát triển |  |
| 1. **CHƯƠNG TRÌNH DEMO** *(Tối đa 3,5 điểm)* |  |
| **Giao diện thân thiện với người dùng** *(0,5 điểm)* |  |
| **Hướng dẫn sử dụng** *(0,5 điểm)* |  |
| **Kết quả thực hiện đúng với kết quả của phần ứng dụng**   * Cài đặt CSDL cùng với các ràng buộc toàn vẹn  (0.5 điểm) * Forms cập nhật đúng (1 điểm) * Reports hợp lý (1 điểm) |  |

**Ghi chú:**

1. *Điểm trong khung “sinh viên thực hiện” là điểm kết quả cuối cùng của từng sinh viên trong quá trình thực hiện niên luận.*
2. *Nếu sinh viên demo chương trình và trả lời vấn đáp không đạt yêu cầu của giáo viên hướng dẫn thì sinh viên sẽ nhận điểm F cho học phần này.*

|  |  |
| --- | --- |
|  | Cần Thơ, ngày tháng năm 2025  GIÁO VIÊN CHẤM |
|  |  |
|  |  |

**MỤC LỤC**

[CHƯƠNG I. TỔNG QUAN 6](#_Toc196402399)

[1.1 BỐI CẢNH VÀ TẦM QUAN TRỌNG TRONG VIỆC PHÁT HIỆN MALWARE 6](#_Toc196402400)

[1.1.1. Bối cảnh 6](#_Toc196402401)

[1.1.2. Tầm quan trọng trong việc phát hiện Malware 6](#_Toc196402402)

[1.2. CÁC THÁCH THỨC HIỆN TẠI TRONG VIỆC PHÁT HIỆN MALWARE 6](#_Toc196402403)

[CHƯƠNG II. CƠ SỞ LÍ THUYẾT 8](#_Toc196402404)

[2.1. KHÁI NIỆM MALWARE 8](#_Toc196402405)

[2.2. PHÂN LOẠI MALWARE 8](#_Toc196402406)

[2.2.1. C&C (Command and Control) 8](#_Toc196402407)

[2.2.2. DDoS (Distributed Denial of Service) 10](#_Toc196402408)

[2.2.3. Horizontal Port Scan 10](#_Toc196402409)

[2.2.4. Okiru 13](#_Toc196402410)

[2.3. Các phương pháp máy học 14](#_Toc196402411)

[2.3.1. Random forest (Rừng ngẫu nhiên) 14](#_Toc196402412)

[2.3.2. Naïve Bayes (Phân lớp) 14](#_Toc196402413)

[2.3.3. Decision Tree (Cây quyết định) 15](#_Toc196402414)

[2.3.4. XGBoost 16](#_Toc196402415)

[2.3.5. GRU (Gated Recurrent Unit) 16](#_Toc196402416)

[CHƯƠNG 3. PHƯƠNG PHÁP NGHIÊN CỨU 17](#_Toc196402417)

[3.1. Tổng quan về bộ dữ liệu 17](#_Toc196402418)

[3.2. Mô hình tổng quan 17](#_Toc196402419)

[3.3. Mô hình dự đoán 18](#_Toc196402420)

[3.4. Tiêu chí đánh giá mô hình 18](#_Toc196402421)

[3.4.1. Độ chính xác (Accuracy) 18](#_Toc196402422)

[3.4.2. Ma trận nhầm lẫn 18](#_Toc196402423)

[3.4.3. Biểu đồ AUC - ROC 18](#_Toc196402424)

[CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM VÀ ĐÁNH GIÁ 19](#_Toc196402425)

[4.1. Môi trường huấn luyện 19](#_Toc196402426)

[4.2. Mô hình huấn luyện 19](#_Toc196402427)

[4.3. Kết quả thực nghiệm 20](#_Toc196402428)

[4.3.1. Mô hình phân lớp 2 lớp (Binary Classifier) 20](#_Toc196402429)

[4.3.2. Mô hình phân lớp nhiều lớp (Multi-Class Classifier) 20](#_Toc196402430)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 22](#_Toc196402431)

[5.1. Kết quả đạt được 22](#_Toc196402432)

[5.2. Hướng phát triển 22](#_Toc196402433)

[TÀI LIỆU THAM KHẢO 23](#_Toc196402434)

# CHƯƠNG I. TỔNG QUAN

## 1.1 BỐI CẢNH VÀ TẦM QUAN TRỌNG TRONG VIỆC PHÁT HIỆN MALWARE

### 1.1.1. Bối cảnh

Trong bối cảnh công nghệ thông tin ngày càng phát triển và Internet trở thành một phần không thể hiếu trong đời sống, trong bối cảnh đó các doanh nghiệp tổ chức cũng dần chuyển sang áp dụng công nghệ số trong quản lí và vận hành. Điều này tạo ra một môi trường hấp dẫn cho các tác nhân xấu, bởi hệ thống mạng rộng lớn luôn chứa đựng nhiều điểm yếu có thể bị khai thác.

Malware không đơn giản là những virus truyền thống mà còn bao gồm các loại phần mềm độc hại khác như worm, trojan, ransomware, spyware, adware,… và các mã độc ẩn trong các ứng dụng di động hoặc trình duyệt web. Các tác nhân tấn công không ngừng nâng cao các kĩ thuật nhằm ẩn mình tinh vi hơn, dễ vượt qua những phát hiện của các phần mềm diệt virus truyền thống.

### 1.1.2. Tầm quan trọng trong việc phát hiện Malware

Việc phát hiện Malware có những lợi ích như sau:

- Bảo vệ dữ liệu và tài sản số: Malware có thể gây ra những thiệt hại nghiêm trọng như đánh cắp thông tin cá nhân, bí mật kinh doanh, hoặc thậm chí phong tỏa toàn hệ thống để đòi tiền chuộc (ransomware). Việc phát hiện sớm giúp ngăn chặn các cuộc tấn công này, bảo vệ dữ liệu và tài sản số của cá nhân và tổ chức.

- Duy trì hoạt động liên tục của hệ thống: Một cuộc tấn công Malware có thể làm gián đoạn hoạt động của hệ thống, gây thiệt hại về kinh tế cũng như ảnh hướng đến uy tín của doanh nghiệp. Việc phát hiện kịp thời giúp giảm thiểu thời gian gián đoạn, đảm bảo hệ thống hoạt động liên tục và ổn định.

- Nâng cao an ninh toàn diện: Phát hiện Malware không chỉ là việc bảo vệ hệ thống máy tính mà còn là một phần trong chiến lược an ninh mạng tổng thể. Nó giúp nhận diện những điểm yếu, từ đó cải thiện biện pháp phòng thủ và nâng cao khả năng ứng phó với các mối đe dọa khác.

- Hỗ trợ điều tra và truy vết: Khi có sự cố an ninh xảy ra, việc phát hiện và phân tích Malware đóng vai trò quan trọng trong việc điều tra, xác định nguyên nhân và truy tìm nguồn gốc của cuộc tấn công. Điều này giúp cải thiện khả năng phòng thủ và đưa ra các biện pháp pháp lý nếu cần thiết.

## 1.2. CÁC THÁCH THỨC HIỆN TẠI TRONG VIỆC PHÁT HIỆN MALWARE

Sự tiến hóa nhanh chóng của Malware: Các tác nhân tấn công liên tục thay đổi cấu trúc mã của Malware, tạo ra các phiên bản mới với những biến thể nhỏ, khiến cho các phương pháp phát hiện truyền thống dựa trên chữ ký trở nên không khả thi.

Đa dạng hóa môi trường và thiết bị: Việc ngày càng có nhiều thiết bị có thể kết nối Internet với tài nguyên hạn chế tạo ra một môi trường phức tạp, nơi các giải pháp bảo mật truyền thống không phải lúc nào cũng có thể áp dụng được. Từ đó đòi hỏi các giải pháp phát hiện Malware phải linh hoạt, có khả năng làm việc hiệu quả trên nhiều nền tảng và môi trường khác nhau.

Khối lượng dữ liệu lớn và phức tạp: Hệ thống mạng hiện đại sản sinh ra khối lượng dữ liệu khổng lồ, việc phân tích và xử lý để phát hiện bất thường đòi hỏi công nghệ tiên tiến và khả năng xử lý mạnh mẽ. Ngoài ra, việc kết hợp dữ liệu từ nhiều hệ thống, thiết bị và nguồn thông tin khác nhau để đưa ra cảnh báo chính xác là một thách thức không nhỏ.

# CHƯƠNG II. CƠ SỞ LÍ THUYẾT

## 2.1. KHÁI NIỆM MALWARE

Malware, viết tắt của “malicious software” (phần mềm độc hại), là một thuật ngữ chung dùng để chỉ bất kỳ chương trình máy tính nào được tạo ra với mục đích gây hại, xâm nhập trái phép, phá hoại hoặc lấy cắp thông tin từ hệ thống. Nói một cách khác, malware không phải là một loại phần mềm cụ thể mà là một tập hợp các chương trình độc hại có thể có những mục tiêu và cơ chế tấn công khác nhau.

Malware được thiết kế để xâm nhập vào máy tính, mạng và thiết bị điện tử nhằm:

- Phá hoại hoạt động của hệ thống (ví dụ: làm chậm, treo máy hoặc hỏng dữ liệu)

- Đánh cắp thông tin nhạy cảm như mật khẩu, thông tin tài chính và dữ liệu cá nhân

- Chiếm quyền kiểm soát từ xa của hệ thống, biến máy tính thành “bot” để thực hiện các cuộc tấn công mạng (như DDoS)

- Mã hóa dữ liệu và đòi tiền chuộc (ransomware) – Tạo điều kiện cho các hoạt động gián điệp hoặc theo dõi người dùng.

Malware thường lây lan qua nhiều kênh khác nhau như:

- Email chứa tệp đính kèm hoặc liên kết độc hại

- Trang web, quảng cáo, ứng dụng không an toàn

- Thiết bị lưu trữ di động như USB

- Lỗ hổng bảo mật trong hệ điều hành và các phần mềm ứng dụng.

## 2.2. PHÂN LOẠI MALWARE

### 2.2.1. C&C (Command and Control)

#### 2.2.1.1. Định nghĩa

C&C (Command and Control) là một cơ chế được sử dụng trong các mạng botnet hoặc phần mềm độc hại để duy trì liên lạc giữa kẻ tấn công và các thiết bị bị xâm nhập. Máy chủ C&C có thể gửi lệnh đến các máy bị nhiễm và nhận dữ liệu từ chúng, giúp kẻ tấn công kiểm soát từ xa các hệ thống bị xâm nhập.

#### 2.2.1.2. Mô tả kỹ thuật

Hệ thống C&C thường hoạt động theo các phương thức sau:

- Giao tiếp theo mô hình Client-Server: Máy bị nhiễm kết nối đến một máy chủ C&C do kẻ tấn công điều khiển.

- Giao tiếp theo mô hình P2P (Peer-to-Peer): Các máy bị nhiễm có thể liên lạc trực tiếp với nhau để nhận lệnh.

- Sử dụng giao thức phổ biến: HTTP, HTTPS, DNS tunneling hoặc các kênh mã hóa để che giấu dữ liệu giao tiếp.

- Cơ chế che giấu: Các phần mềm độc hại có thể thay đổi địa chỉ C&C thường xuyên hoặc sử dụng dịch vụ ẩn danh như Tor để tránh bị phát hiện.

#### 2.2.1.3. Ví dụ minh họa

*Giao tiếp theo mô hình Client-Server – Botnet TrickBot*

- Mô tả: TrickBot là một trojan ngân hàng, hoạt động dựa trên mô hình Client-Server. Khi máy tính bị nhiễm, nó sẽ kết nối đến một máy chủ C&C để nhận lệnh từ kẻ tấn công.

- Cách thức hoạt động:

+ Máy nạn nhân gửi dữ liệu đăng nhập ngân hàng về máy chủ C&C.

+ Máy chủ C&C có thể ra lệnh tải thêm module độc hại hoặc thực hiện giao dịch gian lận.Hacker có toàn quyền kiểm soát máy bị nhiễm từ xa.

*Giao tiếp theo mô hình P2P (Peer-to-Peer) – Botnet ZeroAccess*

- Mô tả: ZeroAccess là một botnet khai thác tiền điện tử, sử dụng mô hình P2P để trao đổi dữ liệu giữa các bot mà không cần một máy chủ C&C cố định.

- Cách thức hoạt động:

+ Khi máy tính bị nhiễm, nó sẽ liên lạc với một số máy ngang hàng khác thay vì kết nối trực tiếp đến một máy chủ trung tâm.

+ Các máy bị nhiễm chia sẻ danh sách địa chỉ IP của các bot khác, giúp botnet hoạt động ngay cả khi một số bot bị vô hiệu hóa.

+ Hệ thống này làm cho việc gỡ bỏ hoặc triệt tiêu mạng botnet trở nên khó khăn hơn.

*Sử dụng giao thức phổ biến (HTTP, HTTPS, DNS tunneling) – Malware Cobalt Strike*

*-* Mô tả: Cobalt Strike là một công cụ tấn công mạng hợp pháp nhưng thường bị hacker lạm dụng để thực hiện tấn công APT. Nó sử dụng HTTP/HTTPS để giao tiếp với máy chủ C&C.

- Cách thức hoạt động:

+ Máy bị nhiễm gửi yêu cầu HTTP/HTTPS giả mạo như truy cập trang web hợp pháp để lấy lệnh từ máy chủ C&C.

+ Hacker có thể ra lệnh từ xa tải xuống mã độc bổ sung hoặc đánh cắp dữ liệu từ máy tính nạn nhân.

+ Việc sử dụng HTTPS giúp ẩn giấu nội dung trao đổi, tránh bị phát hiện bởi hệ thống giám sát mạng.

*Cơ chế che giấu (Tor, thay đổi địa chỉ C&C thường xuyên) – Malware DarkSide*

*-* Mô tả: DarkSide là một ransomware từng được sử dụng trong vụ tấn công Colonial Pipeline. Nó áp dụng cơ chế che giấu bằng cách sử dụng Tor để tránh bị phát hiện.

- Cách thức hoạt động:

+ Khi ransomware xâm nhập hệ thống, nó sẽ kết nối đến máy chủ C&C thông qua mạng Tor, giúp ẩn danh và khó truy vết.

+ Máy chủ C&C có thể thay đổi địa chỉ liên tục, khiến các tổ chức an ninh mạng khó khăn trong việc chặn đứng hoạt động của malware.

+ Sau khi mã hóa dữ liệu, DarkSide hiển thị thông báo yêu cầu tiền chuộc và cung cấp một địa chỉ Tor để nạn nhân liên hệ.

### 2.2.2. DDoS (Distributed Denial of Service)

#### 2.2.2.1. Định nghĩa

DDoS (Distributed Denial of Service – Tấn công từ chối dịch vụ phân tán) là một hình thức tấn công mạng nhằm làm gián đoạn hoạt động của một hệ thống, dịch vụ hoặc mạng bằng cách gửi một lượng lớn yêu cầu từ nhiều nguồn khác nhau, làm quá tải tài nguyên và khiến hệ thống không thể đáp ứng các yêu cầu hợp lệ.

#### 2.2.2.2. Mô tả kỹ thuật

Các cuộc tấn công DDoS thường khai thác số lượng lớn máy tính bị nhiễm mã độc (botnet) để gửi lượng lớn lưu lượng truy cập đến mục tiêu, có thể gây ra:

*Tấn công bão hòa băng thông (Volumetric Attack):*

+ UDP Flood: Gửi một lượng lớn gói tin UDP đến cổng ngẫu nhiên của máy mục tiêu, buộc hệ thống phải kiểm tra liên tục và dẫn đến cạn kiệt tài nguyên.

+ ICMP Flood (Ping Flood): Gửi hàng loạt gói tin ICMP (ping) nhằm tiêu tốn băng thông và tài nguyên xử lý của hệ thống.

Tấn công làm cạn kiệt tài nguyên hệ thống (Protocol Attack):

+ SYN Flood: Gửi hàng loạt yêu cầu kết nối TCP SYN nhưng không hoàn thành quá trình bắt tay (handshake), khiến hệ thống mục tiêu tiêu tốn tài nguyên chờ đợi phản hồi.

+ ACK Flood: Gửi hàng loạt gói tin TCP ACK, làm tốn tài nguyên của hệ thống mục tiêu khi cố gắng xử lý các gói tin này.

*Tấn công tầng ứng dụng (Application Layer Attack):*

HTTP Flood: Gửi một lượng lớn yêu cầu HTTP đến máy chủ web để làm nó quá tải.

Slowloris Attack: Gửi các yêu cầu HTTP một cách từ từ nhưng không hoàn thành, giữ kết nối mở lâu nhất có thể để chiếm hết tài nguyên.

#### 2.2.2.3. Ví dụ minh họa

*Cuộc tấn công DDoS vào GitHub (2018)*

*-* Mô tả: GitHub bị tấn công DDoS với lưu lượng lên tới 1.35 Tbps, được xem là một trong những cuộc tấn công lớn nhất lịch sử.

- Phương thức tấn công: Hacker lợi dụng Memcached amplification, một phương pháp khuếch đại lưu lượng tấn công bằng cách gửi yêu cầu nhỏ nhưng nhận phản hồi rất lớn từ các máy chủ Memcached không được bảo mật.

### 2.2.3. Horizontal Port Scan

#### 2.2.3.1. Định nghĩa

Horizontal Port Scan là một kỹ thuật dò quét cổng mạng trong đó kẻ tấn công quét cùng một số cổng trên nhiều địa chỉ IP khác nhau thay vì quét nhiều cổng trên một địa chỉ IP duy nhất. Mục tiêu chính của hình thức quét này là xác định các thiết bị hoặc dịch vụ cụ thể đang hoạt động trên một mạng rộng.

#### 2.2.3.2. Mô tả kỹ thuật

Quy trình thực hiện:

- Chọn cổng cần quét: Kẻ tấn công xác định một hoặc vài cổng dịch vụ phổ biến như:

+ Cổng 22 (SSH): tìm máy chủ SSH

+ Cổng 80, 443 (HTTP/HTTPS): tìm máy chủ web

+ Cổng 3389 (RDP): tìm máy chủ điều khiển từ xa Windows

+ Cổng 445 (SMB): tìm máy chủ chia sẻ tệp Windows

- Thu thập danh sách địa chỉ IP: kẻ tấn công sử dụng các công cụ hoặc các dịch vụ quét mạng để lấy danh sách địa chỉ IP của mục tiêu.

- Thực hiện quét ngang: Gửi các gói tin SYN (TCP SYN Scan) đến các địa chỉ IP khác nhau nhưng cùng một cổng. Sau đó, ghi nhận các phản hồi từ các IP nào có dịch vụ mở trên cổng đó. Nếu có phản hồi từ nhiều máy chủ trên cùng một cổng, kẻ tấn công có thể thực hiện các bước khai thác sâu hơn.

- Phân tích kết quả và khai thác lỗ hổng: Dựa vào phản hồi, kẻ tấn công sẽ kiểm tra dịch vụ nào đang chạy trên cổng đó. Nếu phát hiện phiên bản lỗi thời hoặc cấu hình yếu, họ có thể thực hiện tấn công xâm nhập hoặc dò mật khẩu.

Công cụ thường sử dụng:

- Nmap: công cụ quét mạng mạnh mẽ

- Masscan: công cụ quét nhanh hơn Nmap, có thể quét hàng triệu IP trong vài phút

- Zmap: quét toàn bộ Internet trong thời gian ngắn

- Shodan: thu thập thông tin thiết bị kết nối Internet

- Angry IP Scanner: công cụ quét mạng dễ sử dụng với giao diện đồ họa

#### 2.2.3.3. Ví dụ minh họa

Ví dụ về Tấn công Horizontal Port Scan trên cổng 80 (HTTP/Web Server)

- Bước 1: Xác định dải IP mục tiêu

+ Hacker thu thập thông tin về dải IP nội bộ mà mục tiêu đang sử dụng, giả sử dải IP đang dùng là 10.0.5.1 đến 10.0.5.100.

+ Mục đích: quét tất cả IP trong dải này trên cổng 80 để tìm máy chủ web đang hoạt động

- Bước 2: Sử dụng một số công cụ như Nmap để quét cổng

nmap -p 80 10.0.5.1-100 –open

Trong đó:

+ -p 80: chỉ quét cổng 80

+ 10.0.5.1-100: quét từ IP 10.0.5.1 đến 10.0.5.100

+ --open: chỉ hiển thị các IP có cổng 80 đang mở

- Bước 3: Phân tích kết quả quét: Giả sử Nmap trả về kết quả danh sách sau:

Nmap scan report for 10.0.5.15

80/tcp open http

Nmap scan report for 10.0.5.42

80/tcp open http

Nmap scan report for 10.0.5.78

80/tcp open http

=> 3 máy chủ đang chạy dịch vụ web (cổng 80 mở)

- Bước 4: Khai thác lỗ hổng:

+ Hacker truy cập vào trang web của các IP này:

* 10.0.5.15: trang quản trị nội bộ của mục tiêu (ví dụ http://10.0.5.15/admin)
* 10.0.5.42: Hệ thống lưu trữ tài liệu (ví dụ http://10.0.5.42/files)
* 10.0.5.78: Máy chủ cũ chưa được cập nhật, chạy phiên bản WordPress lỗi thời

+ Hacker tập trung vào 10.0.5.78 (WordPress cũ) và sử dụng công cụ WPScan để quét lỗ hổng: wpscan --url http://10.0.5.78 --enumerate p

=> Kết quả: phát hiện plugin “Social Media Widget” có lỗ hổng SQL Injection

- Bước 5: Thực hiện tấn công

+ Hacker khai thác lỗ hổng SQL Injection để truy cập cơ sở dữ liệu và đánh cắp thông tin người dùng.

+ Sau đó, hacker leo thang đặc quyền và chiếm quyền điều khiển máy chủ.

#### 2.2.3.4. Cách phòng chống

- Phát hiện và ngăn chặn quét cổng:

+ Sử dụng Firewall: cấu hình firewall để giới hạn IP có thể truy cập vào các cổng quan trọng, sử dụng các công cụ để chặn các yêu cầu quét cổng.

+ Cấu hình Intrusion Detection System (IDS) / Intrusion Prevention System (IPS): cài đặt các công cụ như Snort, Suricata hay Zeek để phát hiện quét cổng bất thường.

+ Sử dụng honeypot để bẫy hacker: dùng công cụ Cowrie hoặc Dionaea để giả mạo dịch vụ mở nhằm theo dõi hành vi của hacker.

- Giảm thiểu bề mặt tấn công:

+ Đóng các cổng không cần thiết

+ Cấu hình Access Control List (ACL) trên Router: chỉ cho phép các địa chỉ IP nội bộ hoặc IP đáng tin cậy truy cập vào các dịch vụ quan trọng.

+ Sử dụng VPN hoặc Port Knocking

- Giám sát hệ thống và phân tích log:

+ Kiểm tra log tường lửa để phát hiện quét cổng

+ Sử dụng SIEM để phân tích dữ liệu mạng: công cụ như Splunk, ELK Stack, hoặc Graylog giúp tổng hợp dữ liệu mạng để phát hiện hoạt động đáng ngờ.

### 2.2.4. Okiru

#### 2.2.4.1. Định nghĩa

     Okiru là một biến thể của mã độc Mirai, được phát hiện lần đầu vào năm 2018 bởi nhóm bảo mật MalwareMustDie. Khác với các biến thể Mirai, Okiru nhắm mục tiêu vào các thiết bị IoT sử dụng bộ xử lý Argonaut RISC Core (ARC), một loại bộ xử lý phổ biến trong các thiết bị như thiết bị đeo thông minh, thiết bị y tế, và thiết bị công nghiệp.

#### 2.2.4.2. Mô tả kỹ thuật

*Phương pháp lây nhiễm*

- Tấn công Telnet Brute-force: Okiru tìm kiếm các thiết bị IoT có cổng Telnet mở và thực hiện brute-force bằng danh sách mật khẩu mặc định hoặc yếu.

- Tận dụng lỗ hổng chưa vá: Nó có thể khai thác các lỗ hổng trong firmware của router, camera giám sát, hoặc thiết bị IoT để chiếm quyền điều khiển.

- Lây lan qua mạng nội bộ: Nếu một thiết bị trong mạng bị nhiễm, Okiru có thể quét các thiết bị khác để tìm điểm yếu và nhân rộng.

*Kiểm soát từ xa thông qua C&C (Command and Control)*

- Liên lạc với máy chủ C&C: Sau khi lây nhiễm, Okiru sẽ kết nối đến máy chủ điều khiển để nhận lệnh.

- Giao tiếp mã hóa: Okiru có thể sử dụng giao thức mã hóa để tránh bị phát hiện.  
 - Cập nhật mã độc tự động: Okiru có thể nhận cập nhật từ C&C để thay đổi kỹ thuật.

     Chiếm quyền điều khiển thiết bị

- Đưa vào chế độ “bot”: Sau khi xâm nhập,biến thiết bị thành bot và chờ lệnh từ C&C.

- Tự động thực thi mã độc: Nó có thể ghi đè lên hệ thống tệp, thay đổi firmware, hoặc sử dụng rootkit để ẩn mình.

- Ngăn chặn cập nhật và khởi động lại thiết bị: Một số biến thể của Okiru có thể vô hiệu hóa tính năng cập nhật hoặc làm cho thiết bị không thể khởi động lại.

*Triển khai các cuộc tấn công DDoS*

- UDP Flooding: Gửi hàng triệu gói tin UDP đến máy chủ mục tiêu để làm tắc nghẽn băng thông.

- TCP SYN Flooding: Gửi hàng loạt yêu cầu TCP SYN để làm quá tải tài nguyên máy chủ. - HTTP Flooding: Tấn công vào các website bằng cách gửi hàng triệu request HTTP để làm sập dịch vụ.

#### 2.2.4.3. Ví dụ minh họa

Kẻ tấn công quét mạng tìm thiết bị yếu:

- Kẻ tấn công sử dụng công cụ quét mạng (ví dụ: Shodan, Masscan) để tìm các thiết bị IoT có cổng Telnet mở (port 23).

- Họ phát hiện hệ thống camera mục tiêu đang sử dụng mật khẩu mặc định và chip NEC V850.

Khai thác lỗ hổng và xâm nhập:

- Kẻ tấn công dùng danh sách mật khẩu phổ biến (default credentials) để đăng nhập vào camera qua Telnet.

- Thành công: Họ chiếm quyền điều khiển phần lớn camera trong thời gian ngắn.

Triển khai malware Okiru:

- Sau khi xâm nhập, kẻ tấn công tải malware Okiru từ máy chủ của chúng lên các camera.

- Malware này thay thế file thực thi mặc định của camera (ví dụ: `/bin/busybox`) bằng một phiên bản đã bị mã độc hóa.

- Okiru được cấu hình để kết nối ngược về máy chủ C&C (Command and Control) có địa chỉ IP `185.130.5.xxx`.

Hình thành mạng botnet và tấn công DDoS:

- Các camera bị nhiễm trở thành "bot" trong mạng lưới của kẻ tấn công.

- Một thời gian sau, kẻ tấn công ra lệnh cho botnet Okiru thực hiện cuộc tấn công DDoS vào trang web cần tấn công

- Cơ chế tấn công: Các camera đồng loạt gửi hàng triệu request HTTP/S đến máy chủ của mục tiêu tấn công, làm nghẽn băng thông và khiến trang web sập.

## 2.3. Các phương pháp máy học

### 2.3.1. Random forest (Rừng ngẫu nhiên)

Random Forest là một thuật toán học máy thuộc nhóm ensemble learning (học máy tập hợp), được sử dụng phổ biến trong các bài toán phân loại (classification) và hồi quy (regression).

Random Forest hoạt động bằng cách tạo ra một tập hợp (rừng) các cây quyết định (Decision Trees) và kết hợp kết quả của chúng để đưa ra dự đoán chính xác hơn.

Mô hình Random Forest Mô hình Random Forest là một kỹ thuật học máy tập hợp (ensemble learning), được xây dựng từ nhiều cây quyết định (Decision Trees) và kết hợp kết quả của chúng để tạo ra một mô hình mạnh mẽ hơn.

Mô hình bao gồm nhiều cây quyết định, trong đó: -Mỗi cây được huấn luyện trên một tập con dữ liệu được lấy mẫu ngẫu nhiên (bootstrap sampling).

Mỗi lần phân tách trong cây, một tập con ngẫu nhiên của các đặc trưng (features) được chọn để tìm ra điểm phân tách tốt nhất.

Khi dự đoán, mô hình tổng hợp kết quả của các cây theo:

- Bỏ phiếu đa số (Majority Voting) cho bài toán phân loại.

- Trung bình kết quả (Averaging Predictions) cho bài toán hồi quy.

### 2.3.2. Naïve Bayes (Phân lớp)

Naïve Bayes là một tập hợp các thuật toán phân loại dựa trên định lý Bayes với giả định mạnh mẽ rằng các đặc trưng đầu vào là độc lập có điều kiện với nhau. Đây là một trong những phương pháp học máy đơn giản nhưng hiệu quả, đặc biệt trong các bài toán phân loại văn bản, phát hiện spam, phân loại cảm xúc và nhiều ứng dụng khác.

#### Định lý Bayes

Định lý Bayes phát biểu rằng, với hai sự kiện A và B, xác suất xảy ra A khi biết B được tính như sau:

Trong đó:

* là xác suất hậu nghiệm, tức là xác suất của A khi đã biết B
* là xác suất có điều kiện, tức là xác suất của B khi biết rằng A xảy ra
* là xác suất tiên nghiệm, hay còn gọi là xác suất ban đầu của A
* là xác suất toàn phần của B

#### Giả định Naïve Bayes

     Naïve Bayes mở rộng định lý Bayes bằng giả định rằng các đặc trưng đầu vào là độc lập có điều kiện với nhau, tức là:

     Điều này giúp đơn giản hóa tính toán đáng kể, làm cho Naïve Bayes trở thành một thuật toán cực kỳ nhanh.

### 2.3.3. Decision Tree (Cây quyết định)

**Decision Tree** là một trong những thuật toán nền tảng và được ưa chuộng trong lĩnh vực máy học (machine learning). Thuộc nhóm phương pháp học có giám sát, Decision Tree tỏa sáng nhờ khả năng xử lý linh hoạt cả bài toán phân loại lẫn hồi quy. Điểm nổi bật của thuật toán này nằm ở sự đơn giản, trực quan và khả năng diễn giải dễ dàng, giúp người dùng từ chuyên gia đến người mới bắt đầu có thể hiểu rõ quy trình ra quyết định của mô hình. Với cấu trúc dạng cây rõ ràng, Decision Tree không chỉ mạnh mẽ về hiệu suất mà còn mang tính thẩm mỹ trong cách trình bày logic, khiến nó trở thành lựa chọn hàng đầu trong nhiều ứng dụng thực tiễn.

#### Mô Hình Cây Quyết Định

Decision Tree hoạt động bằng cách chia nhỏ tập dữ liệu đầu vào thành các tập con dựa trên các thuộc tính (features) được chọn. Quá trình phân chia này diễn ra liên tục, tạo nên một cấu trúc phân nhánh, cho đến khi thỏa mãn một điều kiện dừng, chẳng hạn như đạt độ sâu tối đa của cây, số lượng mẫu tại một nút quá nhỏ, hoặc không còn thuộc tính nào để phân chia thêm. Cấu trúc của cây quyết định bao gồm các thành phần chính sau:

**- Nút gốc (Root Node):** Đại diện cho toàn bộ tập dữ liệu ban đầu, là điểm xuất phát của quá trình phân chia.

**- Nút trong (Internal Node):** Biểu thị một thuộc tính cụ thể và một quy tắc phân. Mỗi nút trong đóng vai trò như một câu hỏi logic, định hướng dữ liệu đi theo các nhánh khác nhau.

**- Nút lá (Leaf Node):** Là điểm kết thúc của mỗi nhánh, đại diện cho kết quả dự đoán cuối cùng, chẳng hạn như một lớp trong bài toán phân loại hoặc một giá trị số trong bài toán hồi quy.

Với cấu trúc trực quan, Decision Tree không chỉ dễ dàng được hình dung mà còn mang lại khả năng giải thích vượt trội. Mỗi đường đi từ nút gốc đến nút lá giống như một chuỗi các quyết định logic, giúp người dùng dễ nắm bắt và truyền đạt cách mô hình đưa ra kết luận.Vì vậy, nó đặc biệt phù hợp với các bài toán đòi hỏi tính minh bạch, chẳng hạn như trong y học (dự đoán bệnh), tài chính (đánh giá rủi ro tín dụng), hoặc các lĩnh vực cần giải thích rõ ràng cho các bên liên quan.Decision Tree còn là nền tảng cho các thuật toán mạnh mẽ hơn như Random Forest hay Gradient Boosting, chứng tỏ tầm quan trọng và tính linh hoạt của nó trong thế giới máy học. Với sự kết hợp giữa hiệu quả tính toán, tính trực quan và khả năng áp dụng rộng rãi, Decision Tree không chỉ là một công cụ, mà còn là một nghệ thuật trong việc mô hình hóa và giải quyết các bài toán phức tạp.

### 2.3.4. XGBoost

XGBoost là một giải thuật học máy mạnh mẽ và tối ưu, được xây dựng dựa trên kỹ thuật Gradient Boosting. Nó nổi bật nhờ khả năng xử lý dữ liệu lớn, tốc độ huấn luyện nhanh và khả năng tránh quá khớp (overfitting) hiệu quả. XGBoost đã giành được nhiều chiến thắng trong các cuộc thi phân tích dữ liệu như Kaggle nhờ hiệu suất và độ chính xác cao.

#### Đặc điểm chính của XGBoost:

- Tốc độ nhanh và hiệu suất cao.

- Giảm hiện tượng quá khớp thông qua điều chuẩn hóa.

- Tích hợp kiểm định chéo và dừng sớm.

- Hỗ trợ xử lý dữ liệu bị thiếu.

- Tối ưu hóa bộ nhớ và tính toán song song.

#### Cách thức hoạt động

XGBoost là một thuật toán dựa trên phương pháp Gradient Boosting, trong đó mô hình được xây dựng tuần tự theo từng bước nhỏ, mỗi bước cố gắng giảm thiểu hàm mất mát bằng cách thêm các mô hình con (cây quyết định) để cải thiện dần độ chính xác.Thay vì xây dựng một mô hình phức tạp từ đầu, XGBoost kết hợp nhiều mô hình đơn giản (weak learners, thường là cây quyết định nông) theo cách tuần tự, mỗi mô hình mới được huấn luyện để sửa lỗi (dự đoán sai) của mô hình trước đó.

### 2.3.5. GRU (Gated Recurrent Unit)

GRU (Gated Recurrent Unit) là một loại mạng nơ-ron hồi tiếp (RNN) được thiết kế để xử lý dữ liệu tuần tự như chuỗi thời gian, văn bản, hoặc lưu lượng mạng. GRU là một giải pháp hiệu quả để khắc phục vấn đề biến mất gradient (vanishing gradient) trong RNN truyền thống, giúp mô hình học được các phụ thuộc dài hạn trong dữ liệu. So với các mô hình khác trong Deep Learning, GRU nổi bật nhờ sự đơn giản, hiệu quả tính toán, và khả năng cân bằng giữa hiệu suất và tài nguyên.

Mục tiêu của GRU được thiết kế để giải quyết vấn đề mất thông tin trong quá trình truyền ngược lâu dài của RNN truyền thống. Nó giúp mạng nhớ những thông tin quan trọng trong chuỗi dài, đồng thời quên đi những thông tin không cần thiết.

GRU sử dụng hai cổng chính để kiểm soát luồng thông tin:

**-** Update Gate (Cổng cập nhật): Quyết định mức độ thông tin từ trạng thái ẩn trước đó được giữ lại và kết hợp với thông tin mới.

**-** Reset Gate (Cổng đặt lại): Quyết định mức độ quên thông tin từ trạng thái ẩn trước khi tính toán trạng thái ẩn mới.

Trong phạm vi Deep Learning, GRU là một công cụ mạnh mẽ, cân bằng giữa hiệu suất và tính đơn giản, đơn giản hóa kiến trúc so với các mô hình khác, GRU dễ triển khai và huấn luyện, GRU có thể kết hợp với các mô hình khác để nâng cao hiệu suất. Mặc dù không phải là mô hình tiên tiến nhất (so với Transformer hay các mô hình lớn như GPT), GRU vẫn là lựa chọn phổ biến cho các ứng dụng cần xử lý chuỗi với tài nguyên hạn chế hoặc dữ liệu vừa và nhỏ.

# CHƯƠNG 3. PHƯƠNG PHÁP NGHIÊN CỨU

## 3.1. Tổng quan về bộ dữ liệu

Đây là một phần của bộ dữ liệu lớn hơn có tên là **IoT-23**, được thu thập bởi Stratosphere Laboratory thuộc Đại học Kỹ thuật Séc (CTU) ở Prague.

Mục tiêu của bộ dữ liệu IoT-23 là cung cấp một tập dữ liệu lớn và thực tế về lưu lượng mạng của các thiết bị Internet of Things (IoT) bị nhiễm mã độc và lưu lượng mạng bình thường của các thiết bị IoT. Điều này giúp các nhà nghiên cứu phát triển các thuật toán học máy để phát hiện phần mềm độc hại.

Các tệp chứa thông tin về các kết nối mạng được ghi lại trong kịch bản này, sử dụng định dạng log của công cụ Zeek (trước đây là Bro). Điểm đặc biệt là tệp này đã được gắn nhãn, cho biết liệu một kết nối cụ thể có liên quan đến hoạt động độc hại hay không.

Thông tin về bộ dữ liệu:

|  |  |  |
| --- | --- | --- |
| **STT** | **Loại Malware** | **Số lượng mẫu tin** |
| 1 | Benign | 8.780.158 |
| 2 | Malicious C&C | 22.701 |
| 3 | Malicious DDoS | 5.778.154 |
| 4 | Malicious Oriku | 4.501.025 |
| 5 | Malicious PartOfAHorizontalPortScan | 5.929.101 |
| **Tổng cộng** | | **25.011.139** |

Bộ dữ liệu bao gồm 21 đặc trưng (features) và 1 nhãn (label):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên feature** | **Sự miêu tả** | **STT** | **Tên feature** | **Sự miêu tả** |
| 1 | ts | Dấu thời gian của sự kiện kết nối. | 12 | conn\_state | Trạng thái kết nối. |
| 2 | uid | Mã định danh duy nhất cho kết nối. | 13 | local\_orig | Chỉ ra kết nối có được coi là cục bộ hay không. |
| 3 | id\_orig\_h | Địa chỉ IP nguồn. | 14 | local\_resp | Chỉ ra kết nối có được coi là cục bộ hay không. |
| 4 | id\_orig\_p | Cổng nguồn. | 15 | missed\_bytes | Số lượng byte bị thiếu trong kết nối. |
| 5 | id\_resp\_h | Địa chỉ IP đích. | 16 | history | Lịch sử các trạng thái kết nối. |
| 6 | id\_resp\_p | Cảng đích. | 17 | orig\_pkts | Số lượng gói tin được gửi từ nguồn đến đích. |
| 7 | proto | Giao thức mạng được sử dụng (ví dụ: 'tcp'). | 18 | orig\_ip\_bytes | Số lượng byte IP được gửi từ nguồn đến đích. |
| 8 | service | Dịch vụ liên quan đến kết nối. | 19 | resp\_pkts | Số lượng gói tin được gửi từ đích đến nguồn. |
| 9 | duration | Thời gian kết nối. | 20 | resp\_ip\_bytes | Số lượng byte IP được gửi từ đích đến nguồn. |
| **STT** | **Tên feature** | **Sự miêu tả** | **STT** | **Tên feature** | **Sự miêu tả** |
| 10 | orig\_bytes | Số byte được gửi từ nguồn đến đích. | 21 | tunnel\_parents | Chỉ ra liệu kết nối này có phải là của đường hầm hay không. |
| 11 | resp\_bytes | Số byte được gửi từ đích đến nguồn. | 22 | label | Nhãn liên quan đến kết nối ('Benign' hoặc 'Malware'). |

## 3.2. Mô hình tổng quan



## 3.3. Mô hình dự đoán

Trong phần này, trình bày phương pahsp thực hiện của 5 mô hình dự đoán trong đề tài này bao gồm XGBoost, Decision Tree, Random Forest, Naïve Bayes và GRU (Gated Recurrent Unit).

## 3.4. Tiêu chí đánh giá mô hình

### 3.4.1. Độ chính xác (Accuracy)

Độ chính xác (Accuracy) là một trong những chỉ số phổ biến nhất để đánh giá hiệu suất của mô hình học máy, đặc biệt trong các bài toán phân loại. Nó được định nghĩa là tỷ lệ giữa số lượng dự đoán đúng (bao gồm cả lớp dương và lớp âm) trên tổng số mẫu trong tập dữ liệu.

Công thức tính độ chính xác:

Trong đó:

* **TP (True Positive)**: Số mẫu dương được dự đoán đúng.
* **TN (True Negative)**: Số mẫu âm được dự đoán đúng.
* **FP (False Positive)**: Số mẫu âm bị dự đoán sai thành dương.
* **FN (False Negative)**: Số mẫu dương bị dự đoán sai thành âm.

### 3.4.2. Ma trận nhầm lẫn

Ma trận nhầm lẫn là một công cụ trực quan dùng để đánh giá hiệu suất của mô hình phân loại, thể hiện mối quan hệ giữa nhãn thực tế và nhãn dự đoán. Ma trận này đặc biệt hữu ích để phân tích chi tiết lỗi dự đoán của mô hình.

Đối với bài toán phân loại nhị phân, ma trận nhầm lẫn có dạng 2x2:

|  |  |  |
| --- | --- | --- |
|  | **Dự đoán: Dương** | **Dự đoán: Âm** |
| **Thực tế: Dương** | TP | FN |
| **Thực tế: Âm** | FP | TN |

Tùy bài toán mà ta có thể có ma trận nhầm lẫn có dạng NxN, với N là số lớp của model

Từ ma trận nhầm lẫn, có thể tính toán các chỉ số khác như:

* **Precision (Độ chính xác dương**):
* **Recall (Độ nhạy):**
* **F1-Score**:

### 3.4.3. Biểu đồ AUC - ROC

Biểu đồ ROC (Receiver Operating Characteristic) là một công cụ đánh giá hiệu suất của mô hình phân loại nhị phân, biểu diễn mối quan hệ giữa Tỷ lệ dương tính thực (True Positive Rate - TPR) và Tỷ lệ dương tính giả (False Positive Rate - FPR) ở các ngưỡng phân loại khác nhau.



Diện tích dưới đường cong ROC, gọi là AUC (Area Under the Curve), là một chỉ số tổng quát đo lường khả năng phân biệt giữa các lớp của mô hình.

Giá trị AUC nằm trong khoảng [0, 1]:

* AUC = 1: Mô hình hoàn hảo, phân biệt hoàn toàn các lớp.
* AUC = 0.5: Mô hình không tốt hơn dự đoán ngẫu nhiên.
* AUC < 0.5: Mô hình dự đoán ngược (kém hơn ngẫu nhiên).

# CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM VÀ ĐÁNH GIÁ

## 4.1. Môi trường huấn luyện

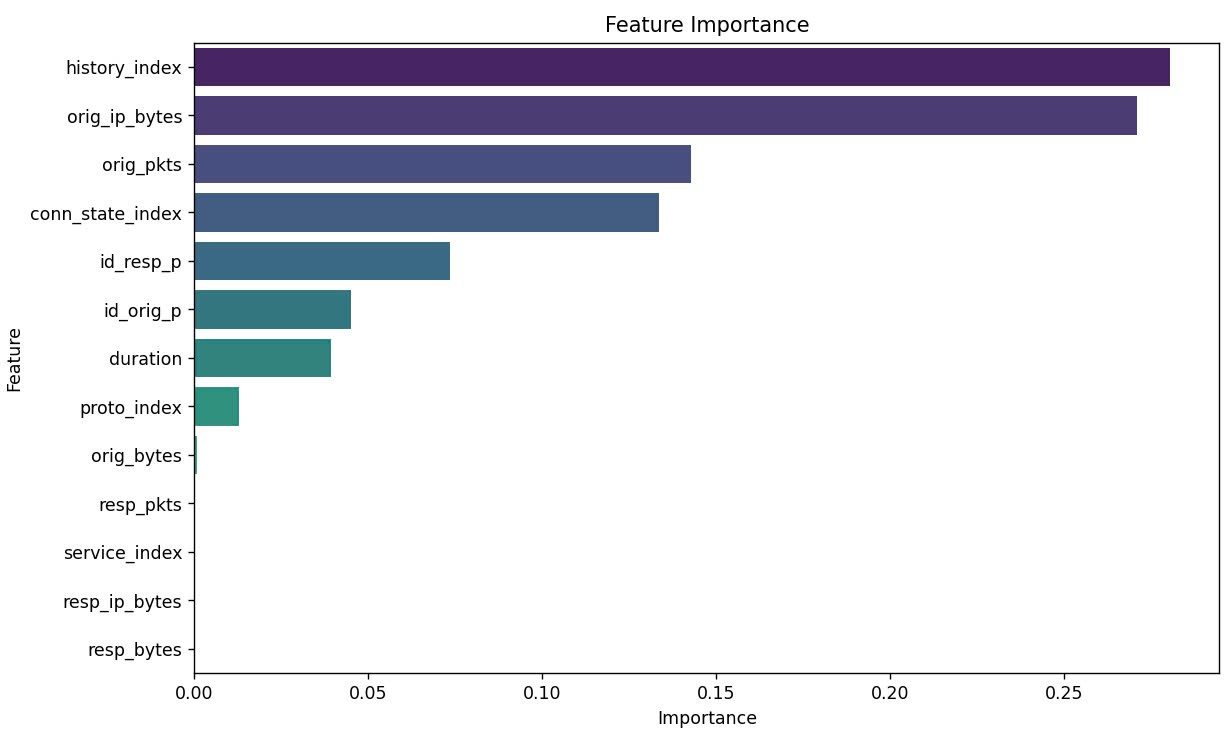
Trong nghiên cứu máy này, tất cả các thí nghiệm được thực hiện trên máy tính cá nhân chạy hệ điều hành Windows 11, bộ xử lý Intel Core i7 gen 11th, bộ nhớ 8GB. Ngôn ngữ lập trình được sử dụng để huấn luyện và kiểm tra kết quả thực nghiệm là Python.

## 4.2. Mô hình huấn luyện

Đầu tiên, chúng tôi xây dựng hai mô hình cơ sở, cả hai đều sử dụng toàn bộ đặc trưng của tập huấn luyện. Mô hình đầu tiên là phân loại nhị phân (binary classifier), tức là chúng tôi chỉ phân loại các luồng Không có tấn công (benign) và các luồng có tấn công (malicious). Mô hình thứ hai là phân loại nhiều lớp (multi-class classifier), tức là chúng tôi muốn phân loại theo từng loại tấn công riêng lẻ, cụ thể là 4 loại tấn công.

Sau đó, chúng tôi tiếp tục quá trình huấn luyện các mô hình như trên nhưng chỉ tập trung vào việc giảm bớt số lượng đặc trưng. Việc này nhằm mục tiêu tối ưu hóa hiệu suất của mô hình thông qua việc loại bỏ hoặc giảm thiểu những đặc trưng không quan trọng hoặc có sự tương quan cao. Quá trình này góp phần cải thiện khả năng tổng quan của mô hình bằng cách tạo ra một không gian đặc trưng tối ưu, từ đó giúp mô hình học tốt hơn và tránh tình trạng quá khớp (overfitting). Bằng cách tập trung vào những đặc trưng quan trọng và loại bỏ những đặc trưng không cần thiết, thời gian huấn luyện mô hình giảm đi đáng kế nhưng độ chính xác của mô hình vẫn giữ ở mức cao. Giúp mô hình có khả năng tạo ra các dự đoán chính xác và hiệu quả hơn khi phân loại một dữ liệu mới.

Chúng tôi sử dụng mô hình Random Forest để trích chọn độ quan trọng của các đặc trưng nhằm chọn ra các đặc trưng hữu ích cho việc huấn luyện. Việc này nhằm tối ưu hóa hiệu suất mô hình thông qua việc loại bỏ hoặc giảm thiểu những đặc trưng không quan trọng. Góp phần cải thiện khả năng tổng quan của mô hình từ đó giúp mô hình học tốt hơn và tránh tính trạng “học vẹt” (overfitting).



Biểu đồ biểu hiện mức độ quan trọng của các đăc trưng

## 4.3. Kết quả thực nghiệm

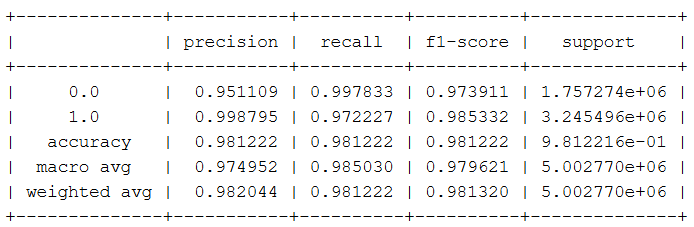
## 4.3.1. Mô hình phân lớp 2 lớp (Binary Classifier)

#### 4.3.1.1. Thực nghiệm giữ nguyên đặc trưng

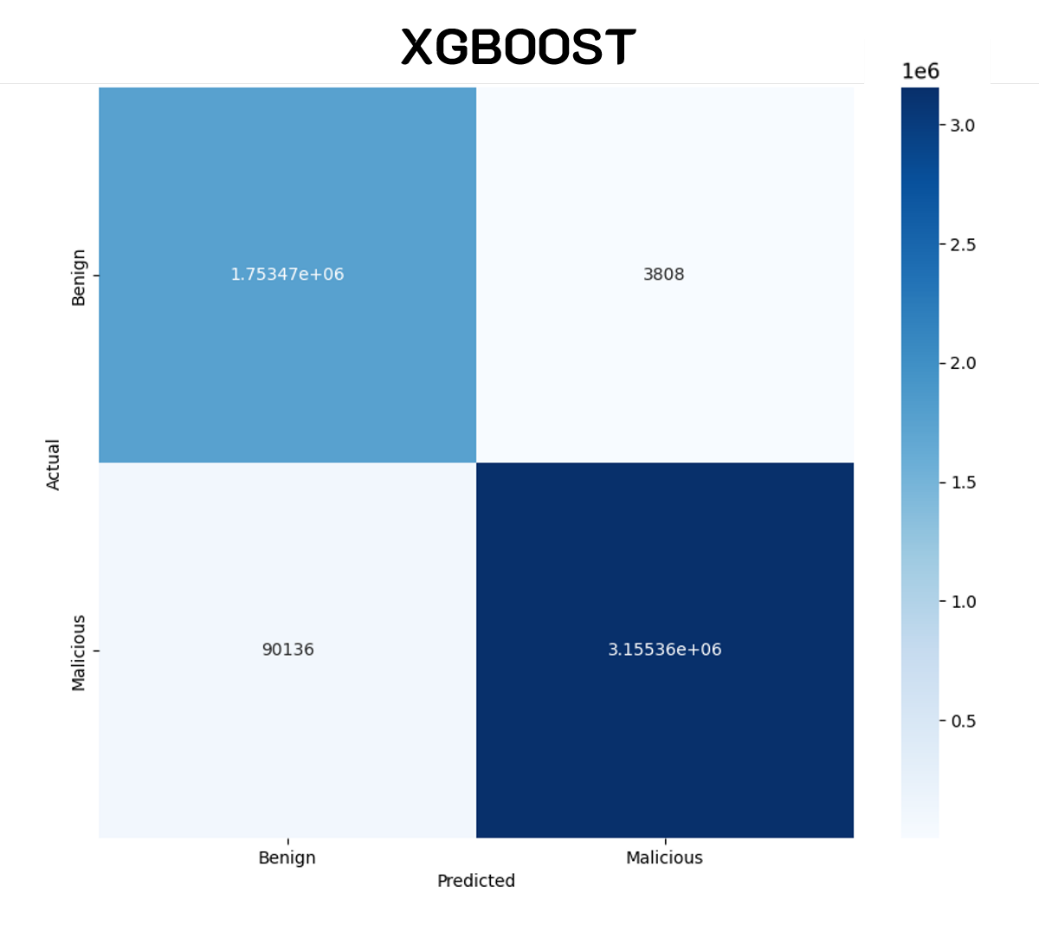
Kết quả huấn luyện của 4 mô hình được trình bày trong hình dưới

Chúng tôi trình bày kết quả thực nghiệm từ các mô hình phân lớp 2 lớp trên bộ thử nghiệm (Testing set) ở bên dưới:

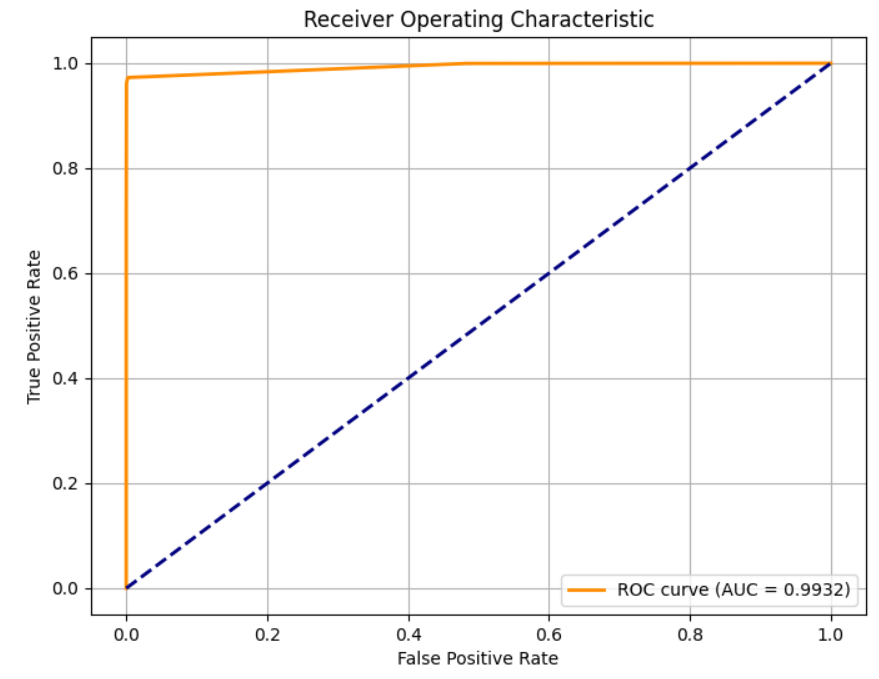
**XGBoost:** Có độ chính xác 98,13%



Đánh giá độ chính xác của mô hình XGBoost

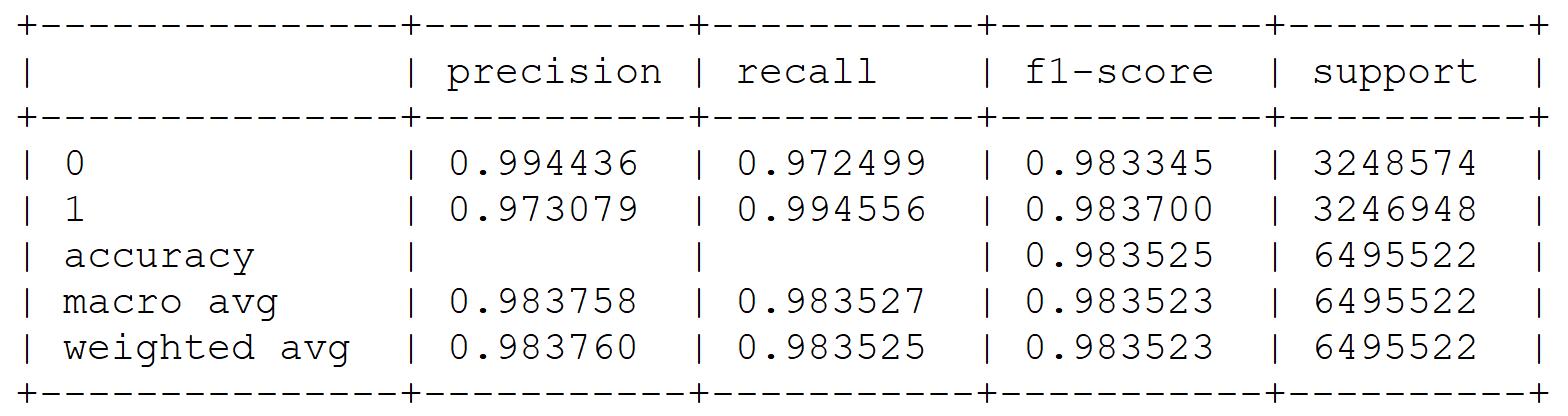


Ma trận nhầm lẫn mô hình XGBoost

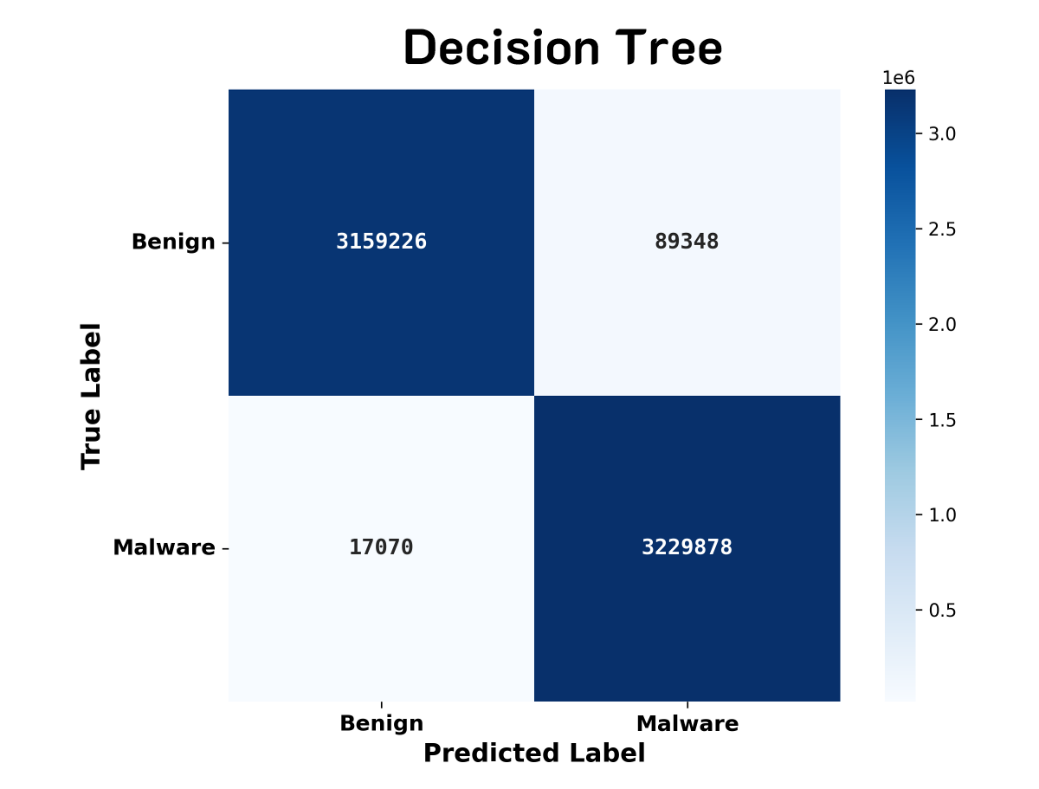


Đánh giá mô hình XGBoost qua đường cong ROC của độ đo AUC

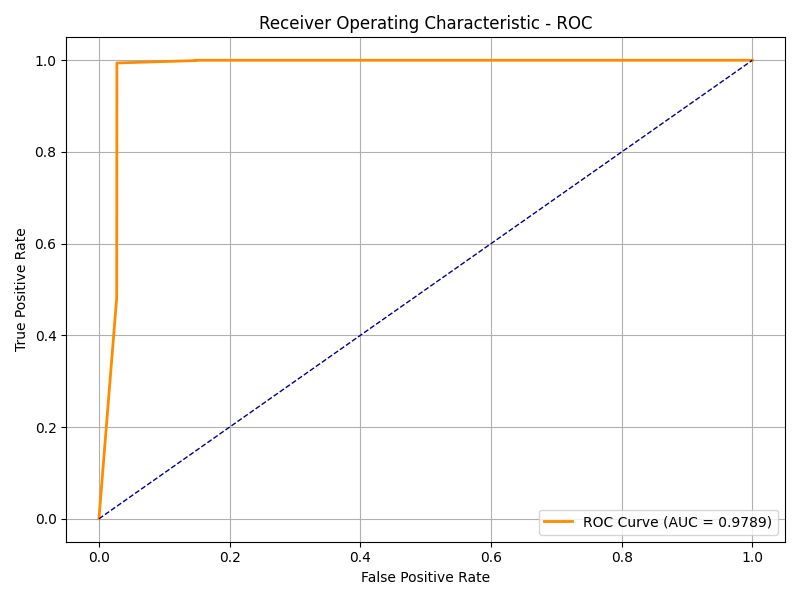
**Decision Tree:** Có độ chính xác 98.35%



Đánh giá độ chính xác của mô hình Decision Tree

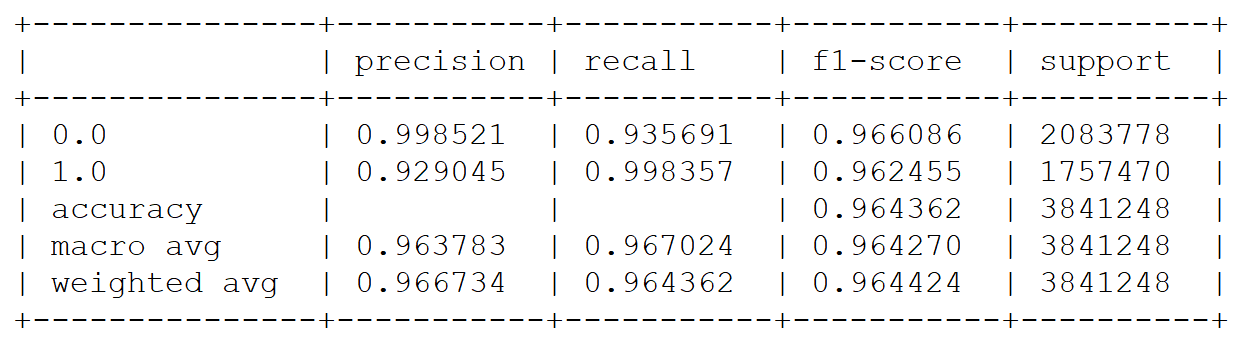


Ma trận nhầm lẫn mô hình Decision Tree



Đánh giá mô hình Decision Tree qua đường cong ROC của độ đo AUC

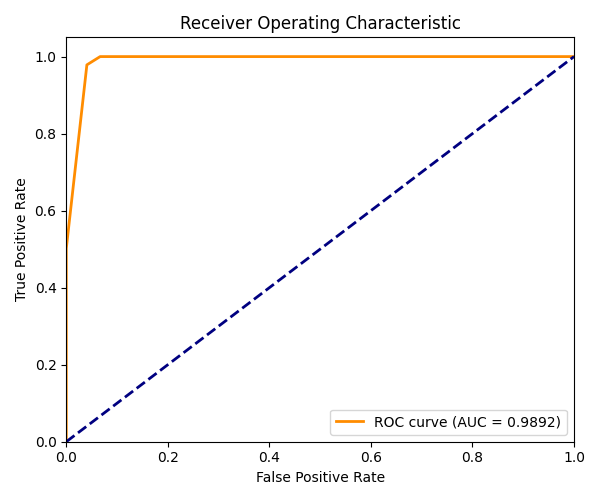
**Random Forest:** Có độ chính xác 96.43%



Đánh giá độ chính xác của mô hình Random Forest

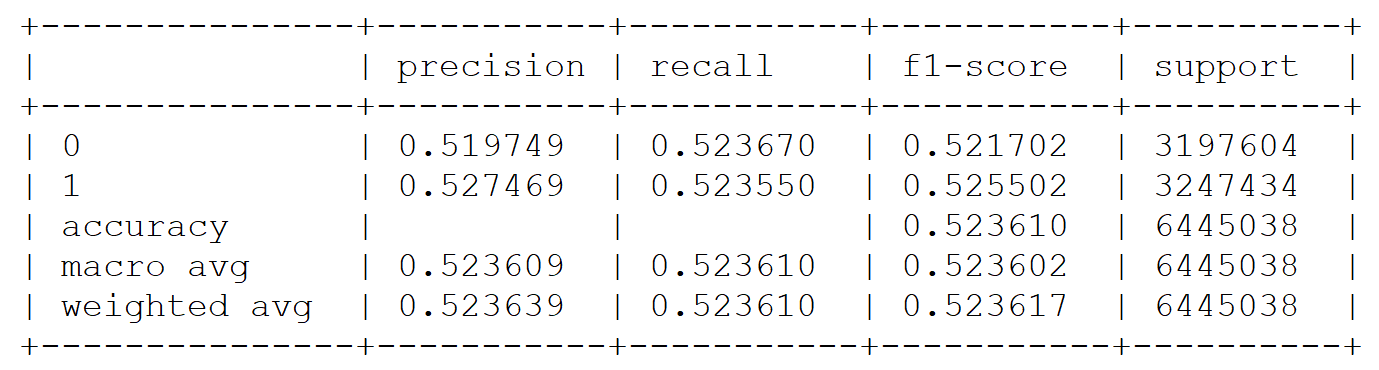


Ma trận nhầm lẫn mô hình Random Forest

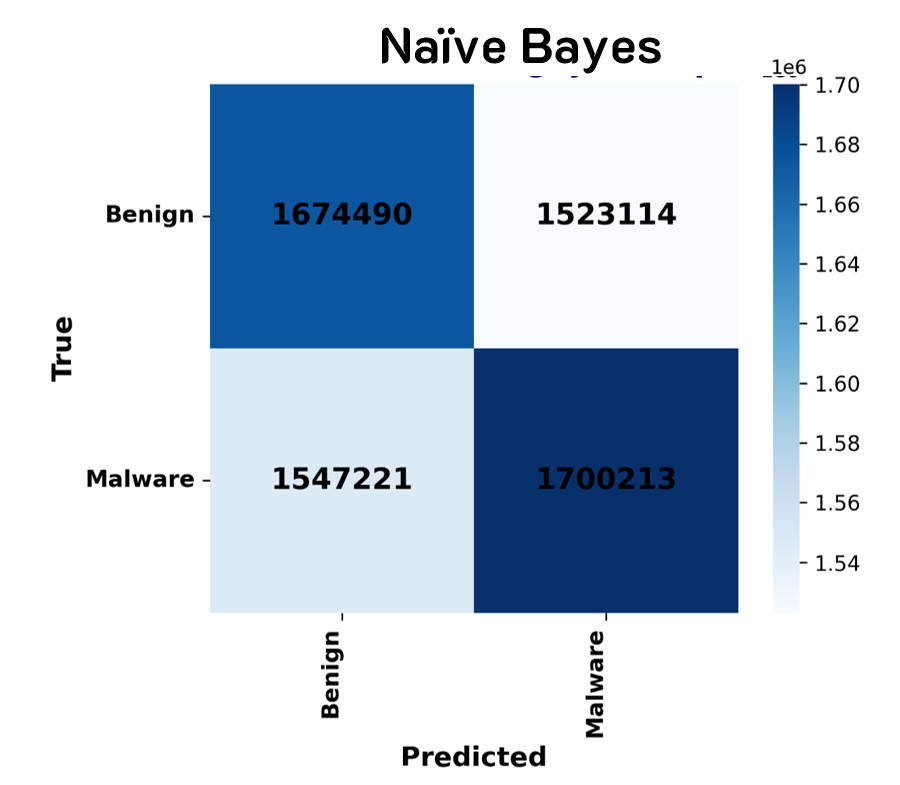


Đánh giá mô hình Random Forest qua đường cong ROC của độ đo AUC

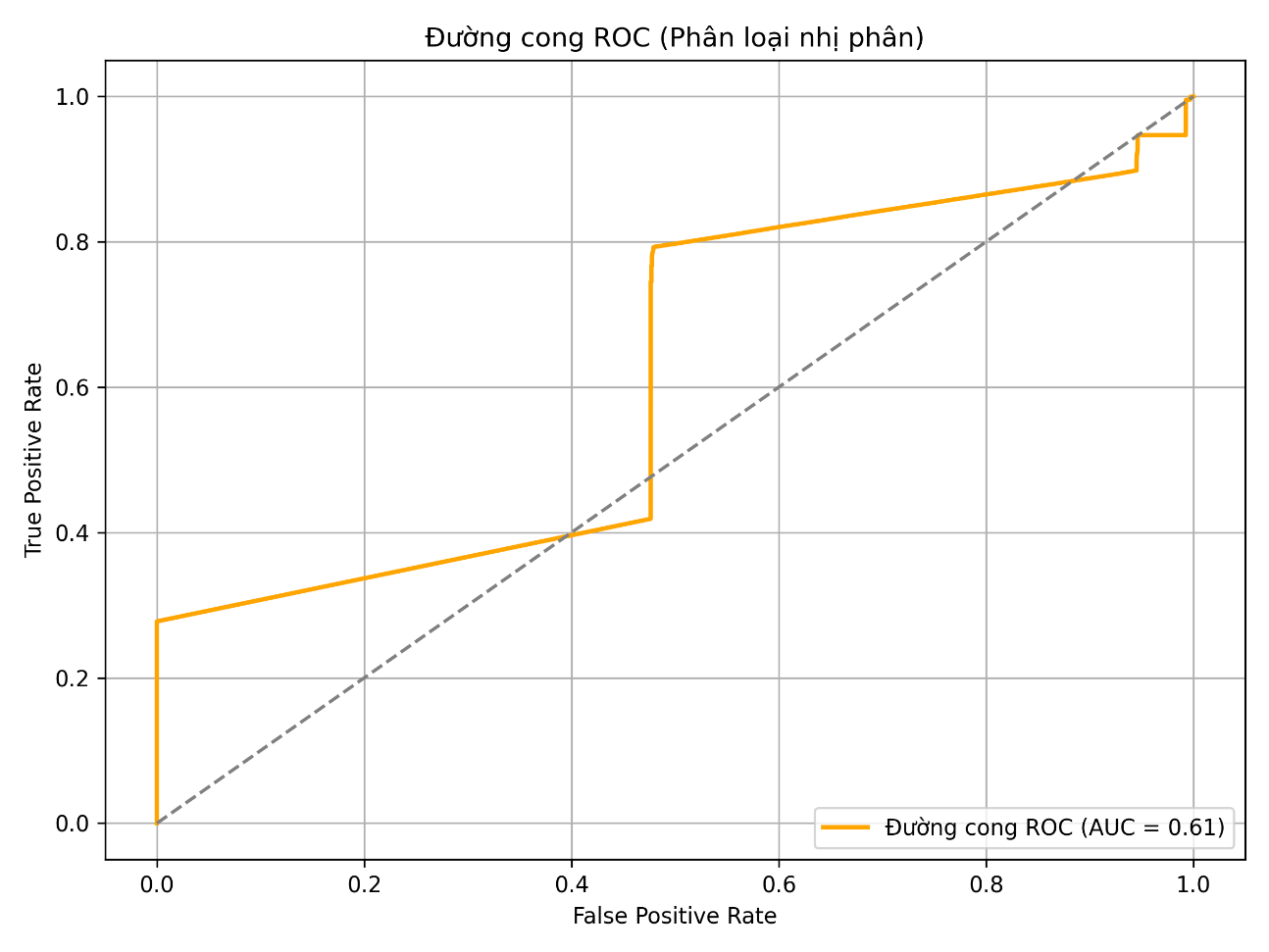
**Naïve Bayes:** Có độ chính xác 52.36%



Đánh giá độ chính xác của mô hình Naïve Bayes



Ma trận nhầm lẫn mô hình Naïve Bayes



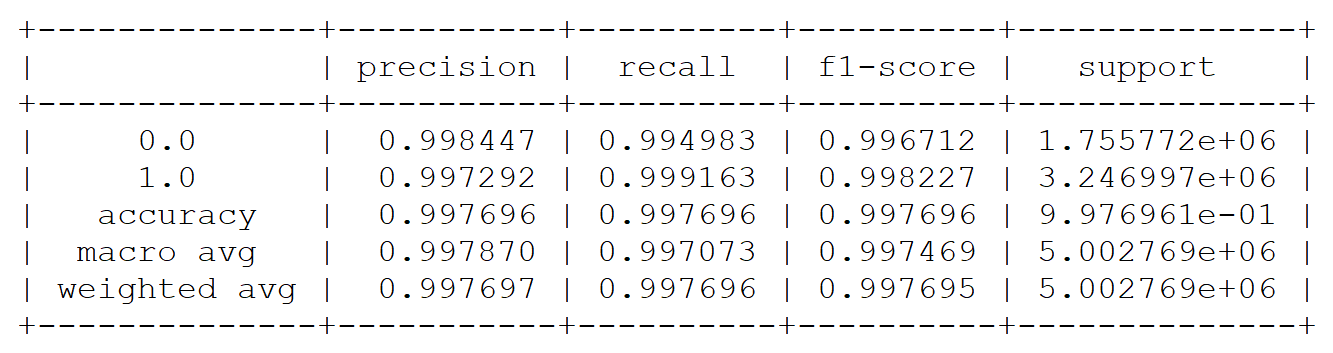
Đánh giá mô hình Naïve bayes qua đường cong ROC của độ đo AUC

#### 4.3.1.2. Thực nghiệm trích xuất đặc trưng

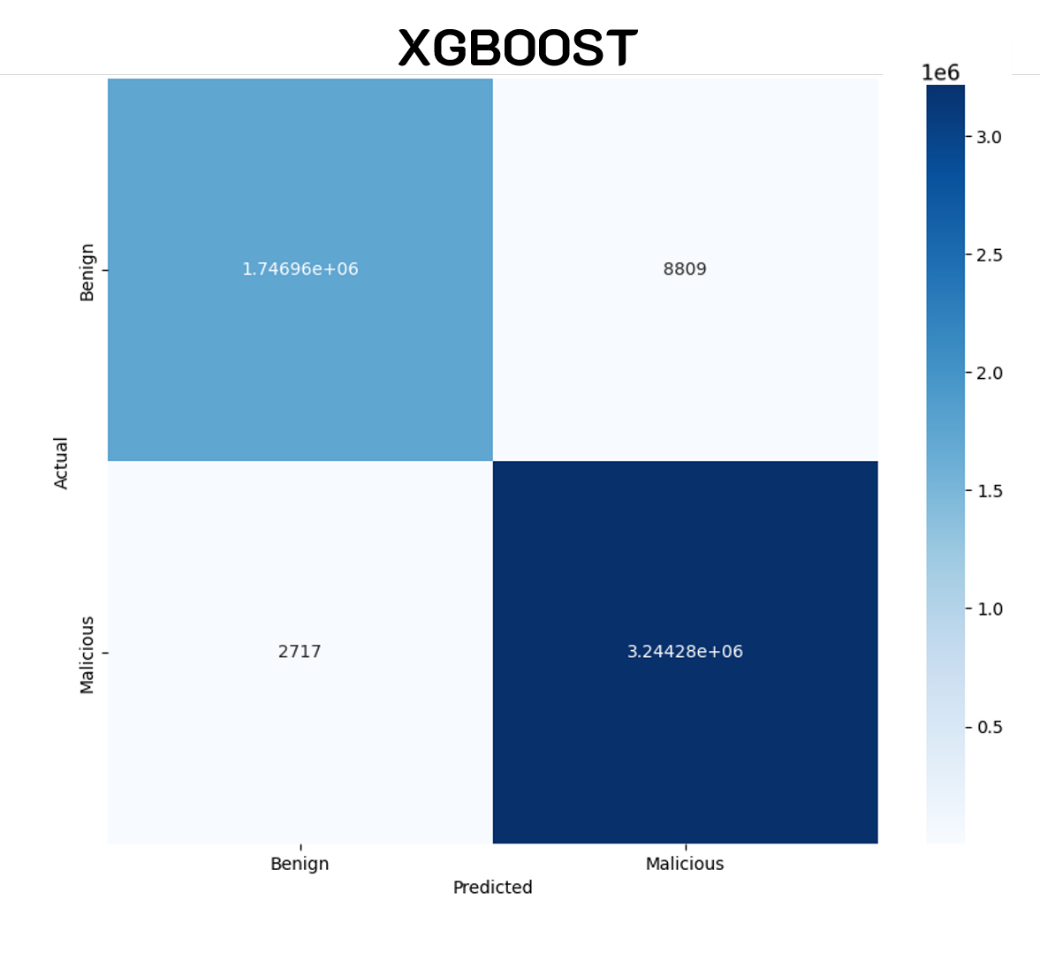
Kết quả huấn luyện của 4 mô hình được trình bày trong hình dưới

Chúng tôi trình bày ma trận nhầm lẫn của dự đoán từ các mô hình phân lớp 2 lớp trên bộ thử nghiệm (Testing set) trong các hình bên dưới

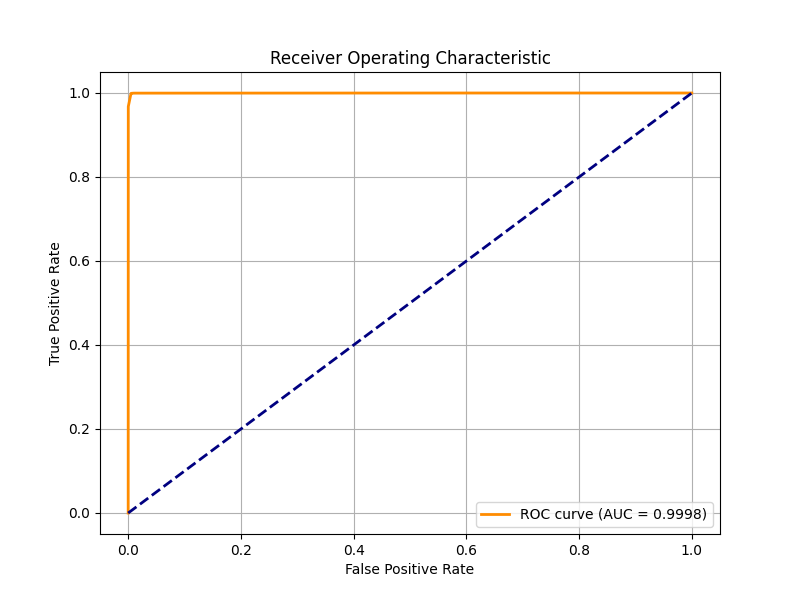
**XGBoost:** Có độ chính xác 99.76%



Đánh giá độ chính xác của mô hình XGBoost

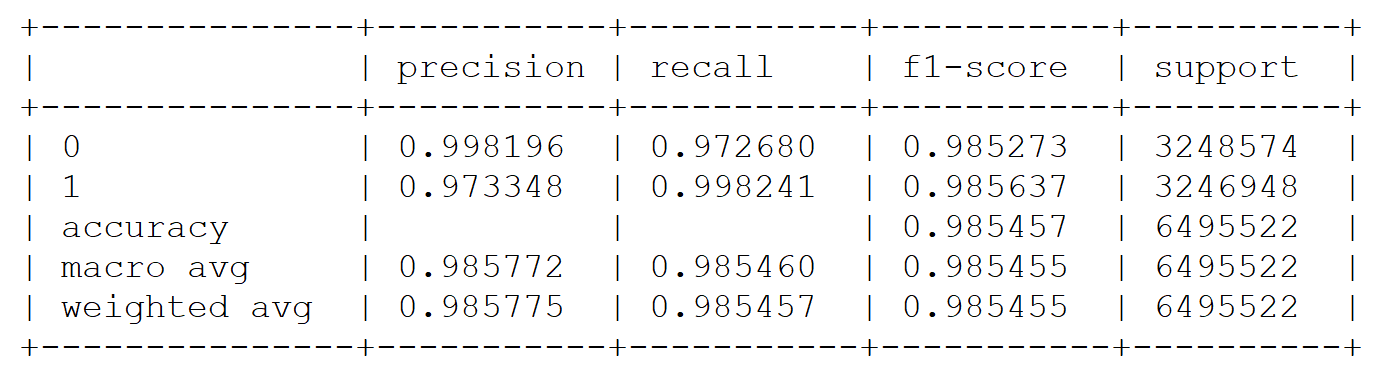


Ma trận nhầm lẫn mô hình XGBoost

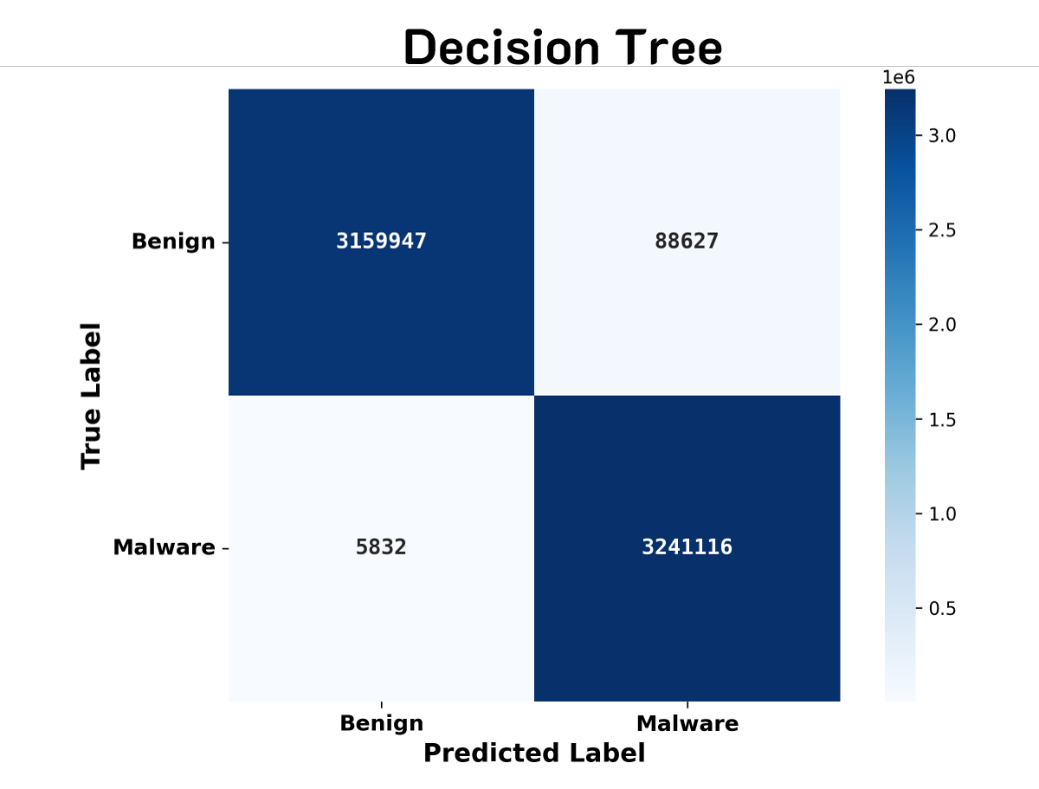


Đánh giá mô hình XGBoost qua đường cong ROC của độ đo AUC

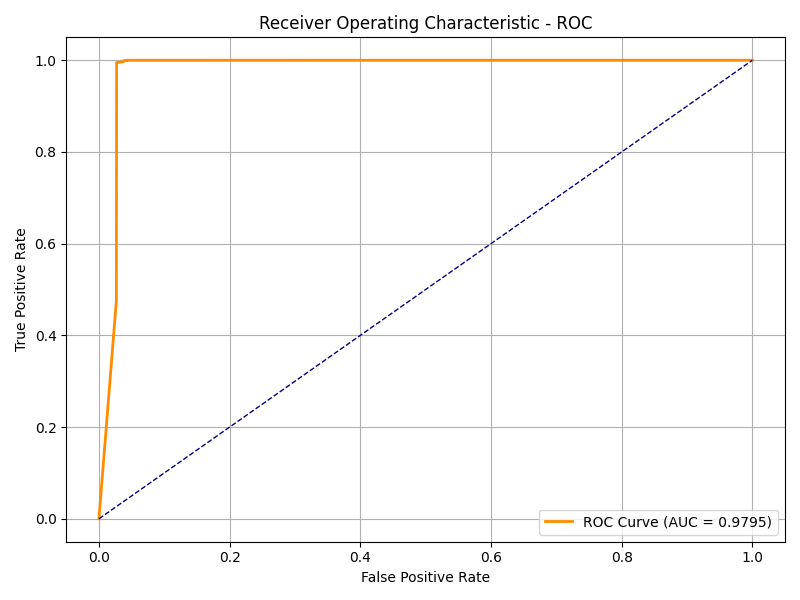
**Decision Tree:** Có độ chính xác 98.54%



Đánh giá độ chính xác của mô hình Decision Tree

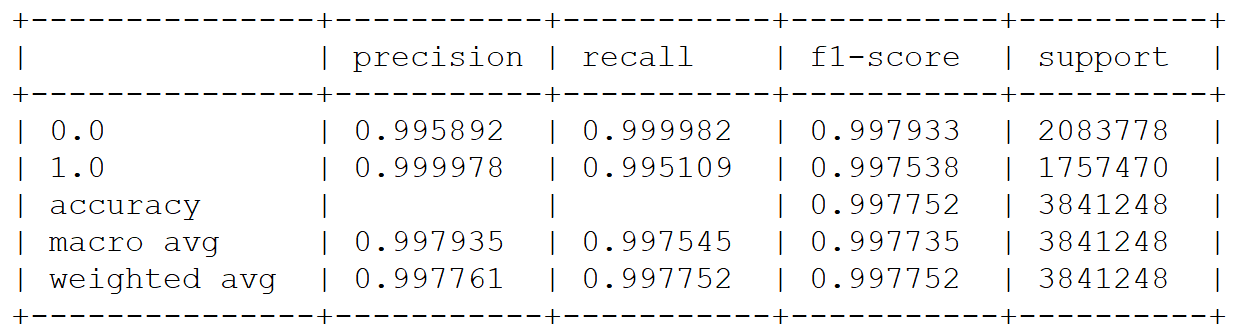


Ma trận nhầm lẫn mô hình Decision Tree

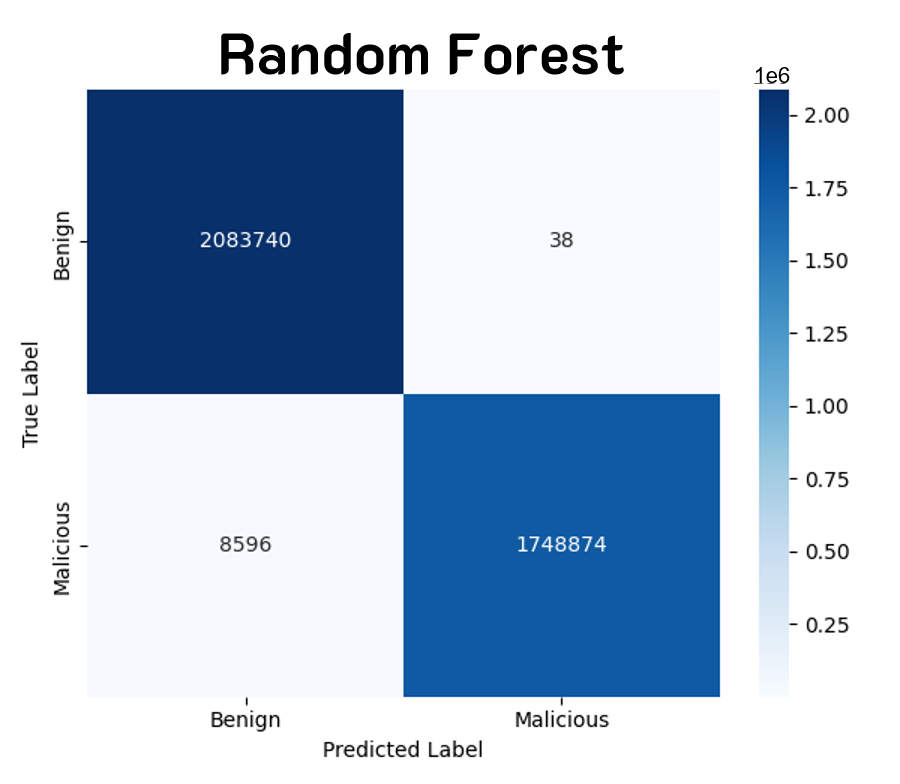


Đánh giá mô hình Decision Tree qua đường cong ROC của độ đo AUC

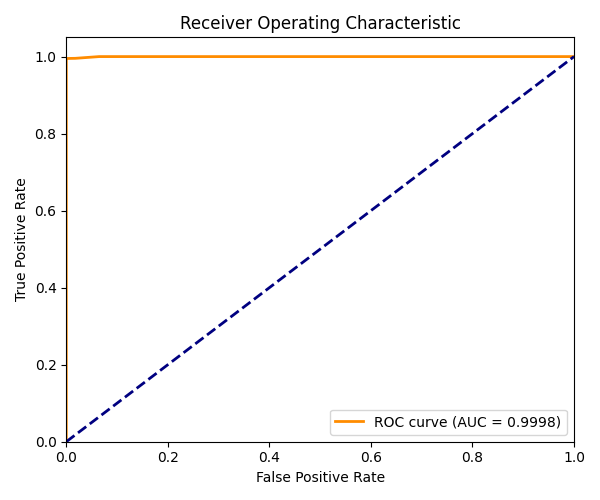
**Random Forest:** Có độ chính xác 99.77%



Đánh giá độ chính xác của mô hình Random Forest

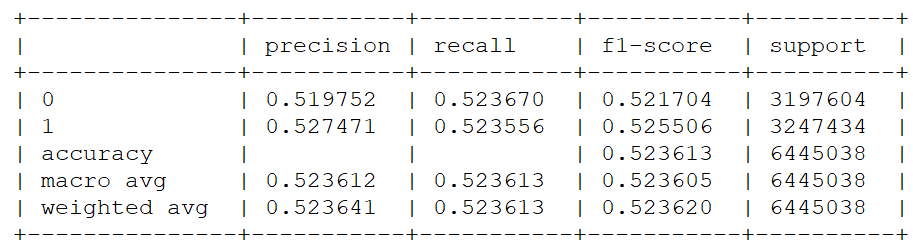


Ma trận nhầm lẫn mô hình Random Forest

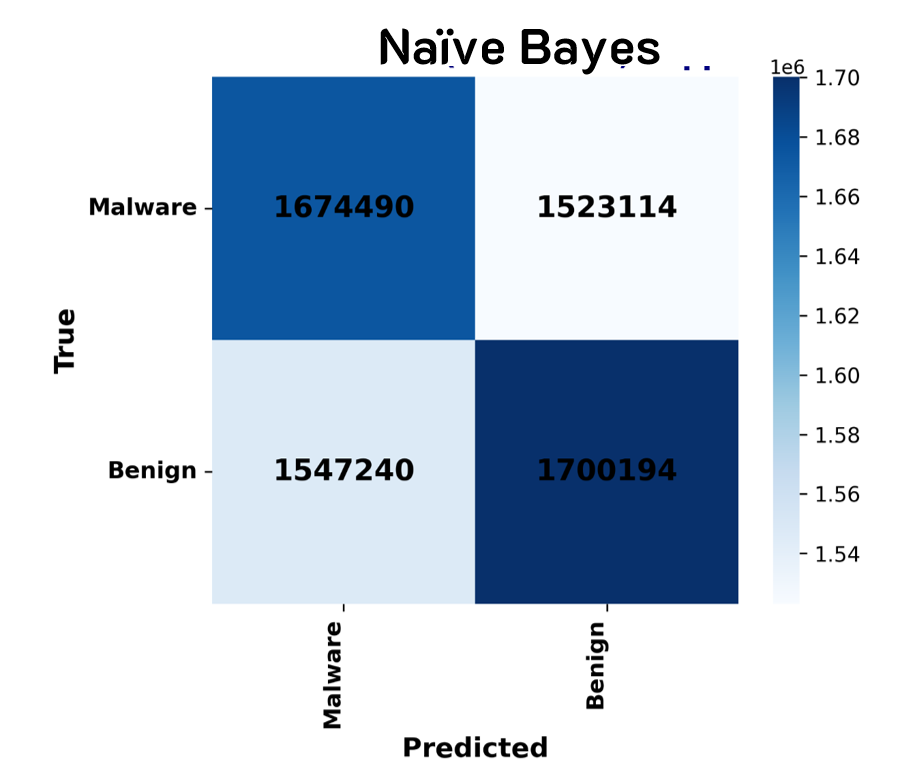


Đánh giá mô hình Random Forest qua đường cong ROC của độ đo AUC

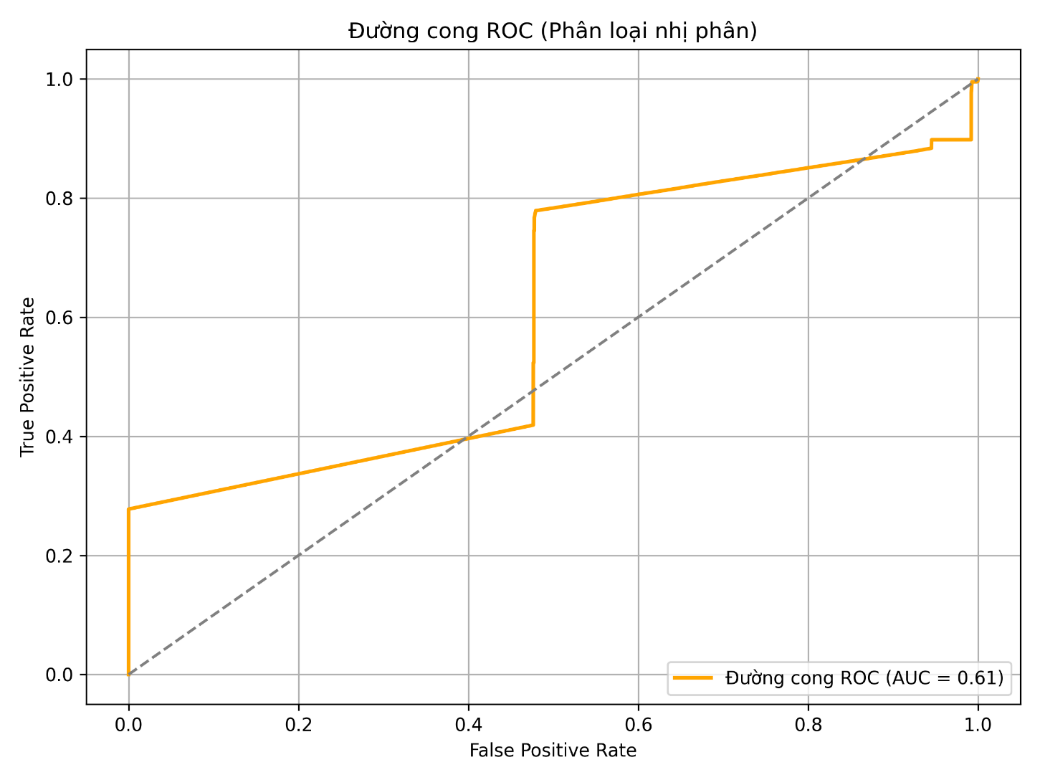
**Naïve Bayes:** Có độ chính xác 52.36%



Đánh giá độ chính xác của mô hình Naïve Bayes



Ma trận nhầm lẫn mô hình Naïve Bayes



Đánh giá mô hình Naïve bayes qua đường cong ROC của độ đo AUC

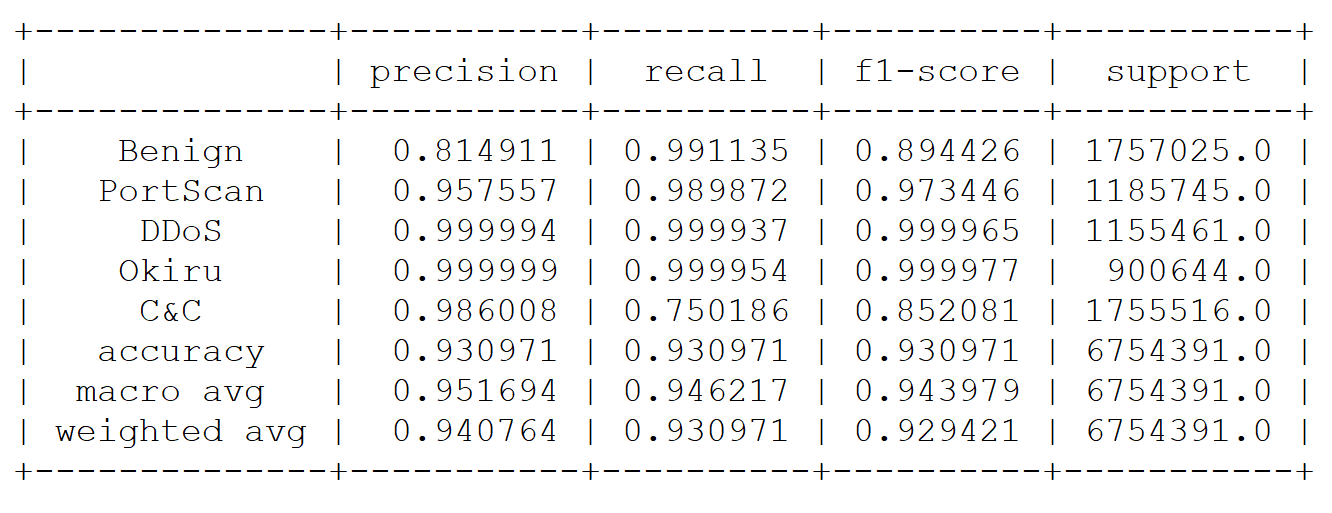
## 4.3.2. Mô hình phân lớp nhiều lớp (Multi-Class Classifier)

#### 4.3.2.1. Thực nghiệm giữ nguyên xuất đặc trưng

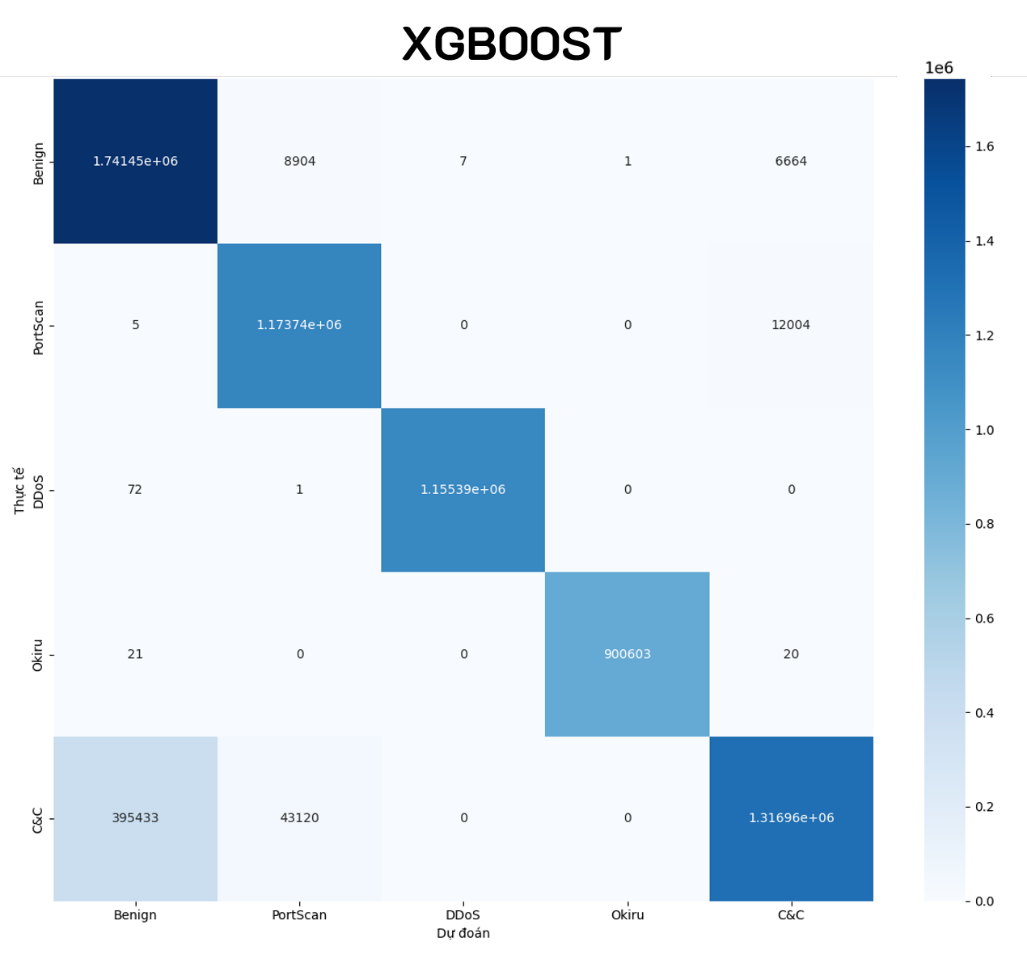
Kết quả huấn luyện của 4 mô hình được trình bày trong hình dưới

Chúng tôi trình bày ma trận nhầm lẫn của dự đoán từ các mô hình phân lớp đa lớp trên bộ thử nghiệm (Testing set) trong các hình bên dưới

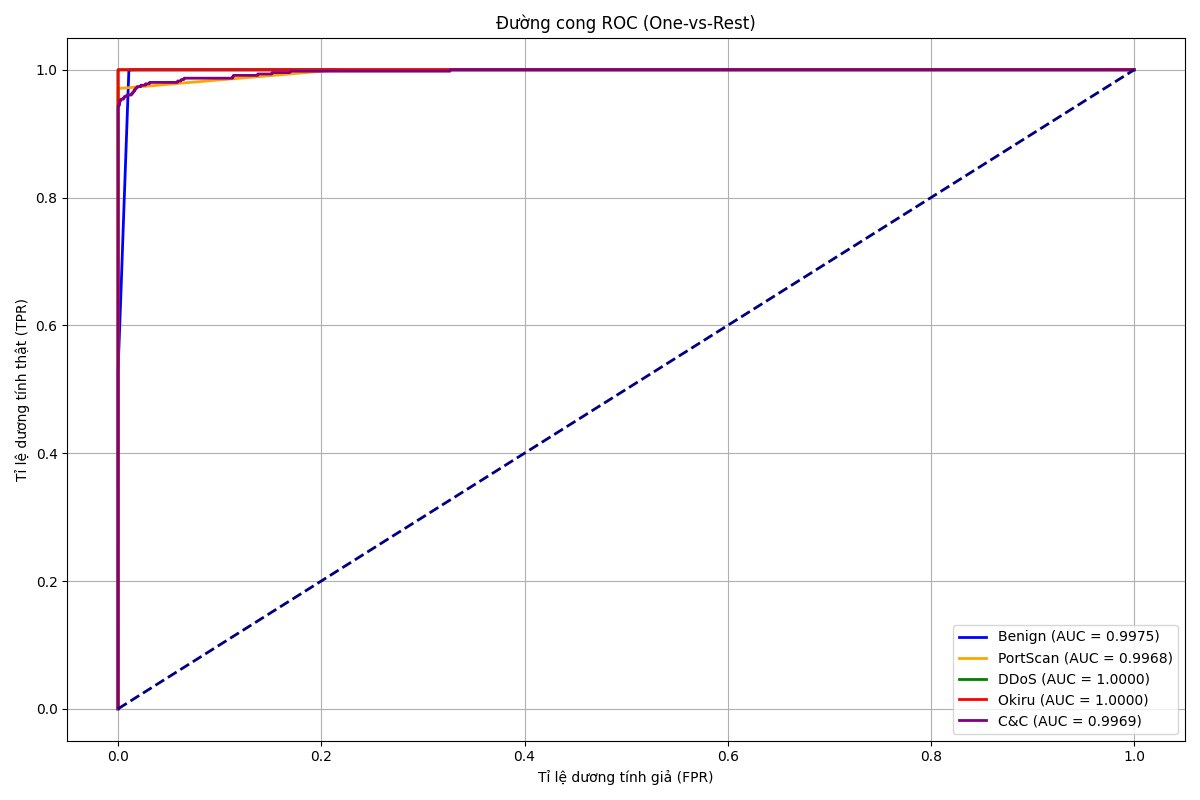
**XGBoost:** Có độ chính xác 93.09%



Đánh giá độ chính xác của mô hình XGBoost

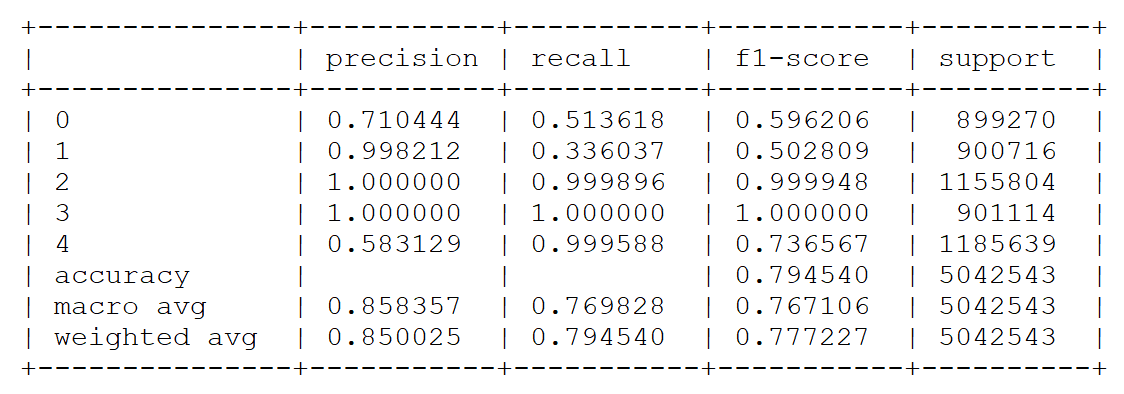


Ma trận nhầm lẫn mô hình XGBoost

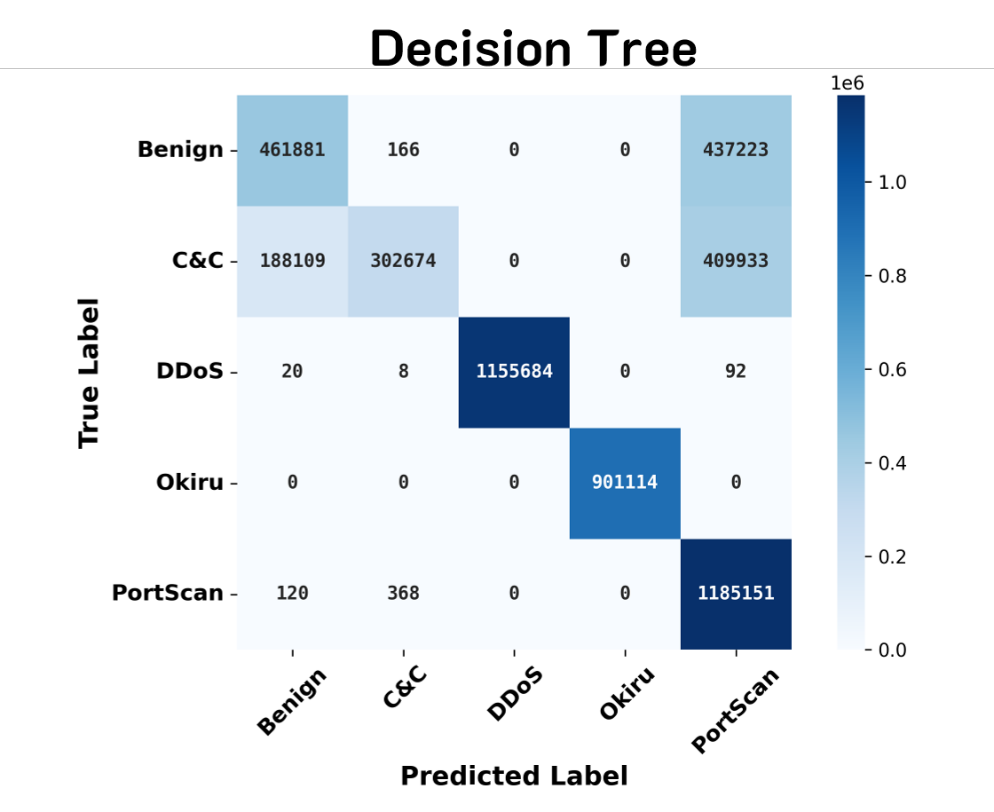


Đánh giá mô hình XGBoost qua đường cong ROC của độ đo AUC

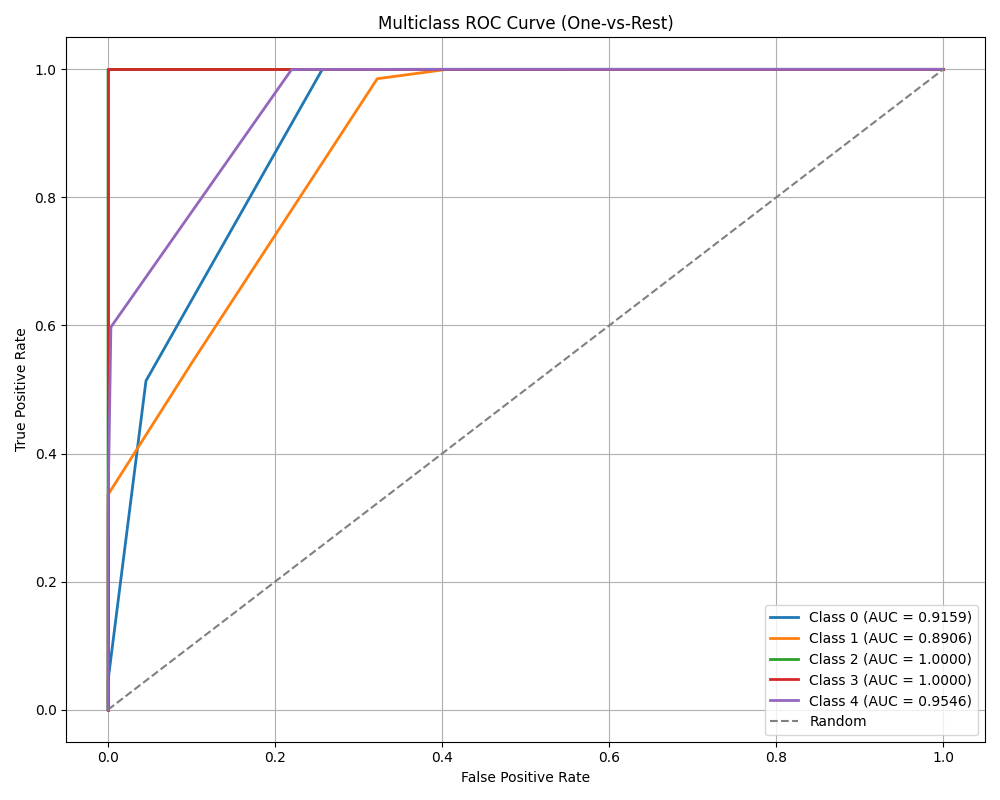
**Decision Tree:** Có độ chính xác 79.45%



Đánh giá độ chính xác của mô hình Decision Tree

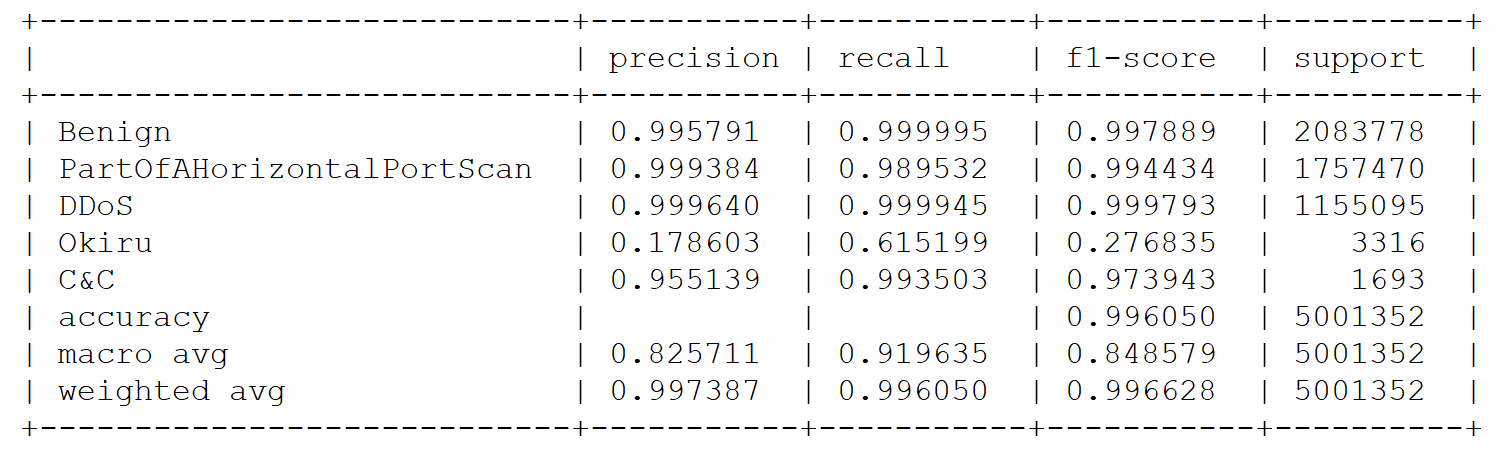


Ma trận nhầm lẫn mô hình Decision Tree

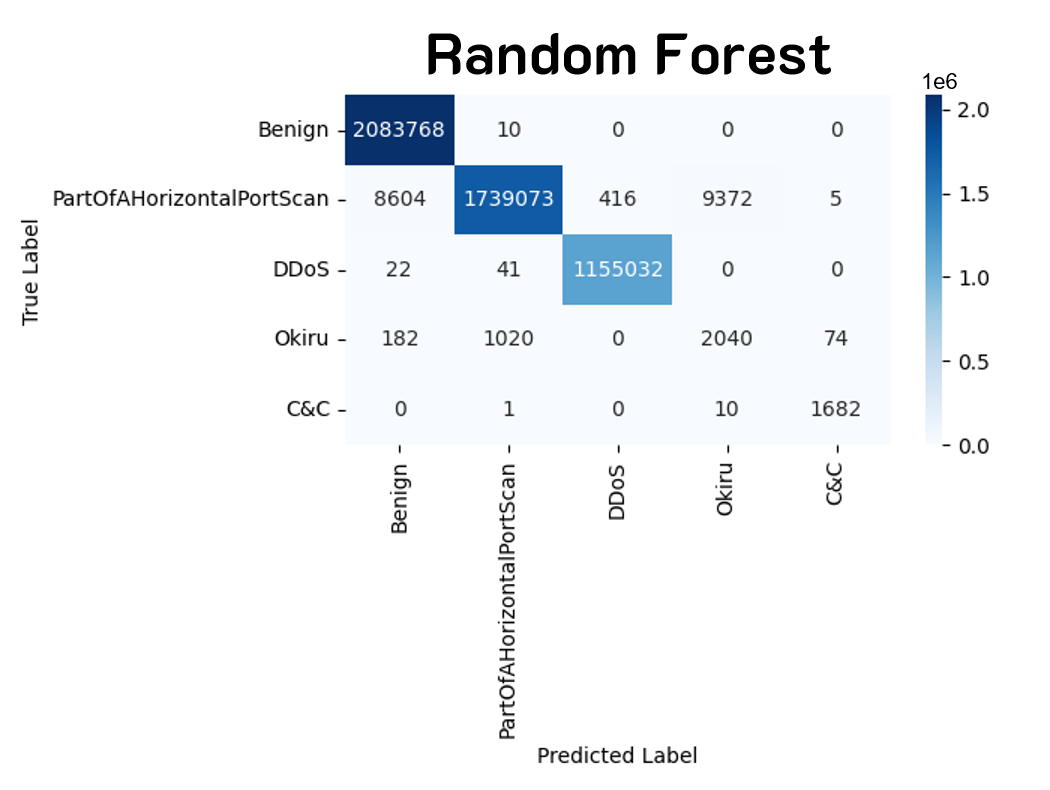


Đánh giá mô hình Decision Tree qua đường cong ROC của độ đo AUC

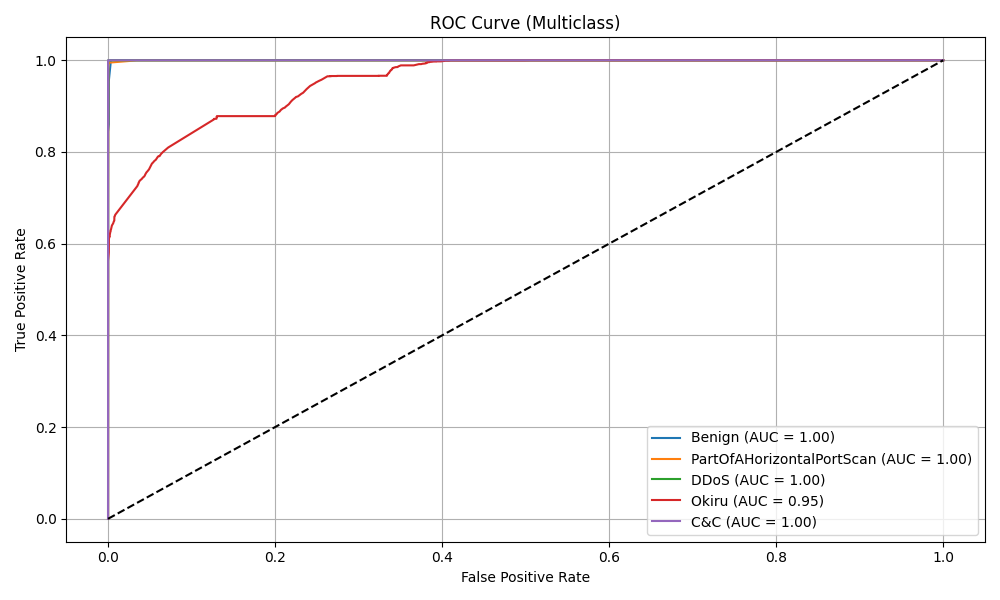
**Random Forest:** Có độ chính xác 99.60%



Đánh giá độ chính xác của mô hình Random Forest

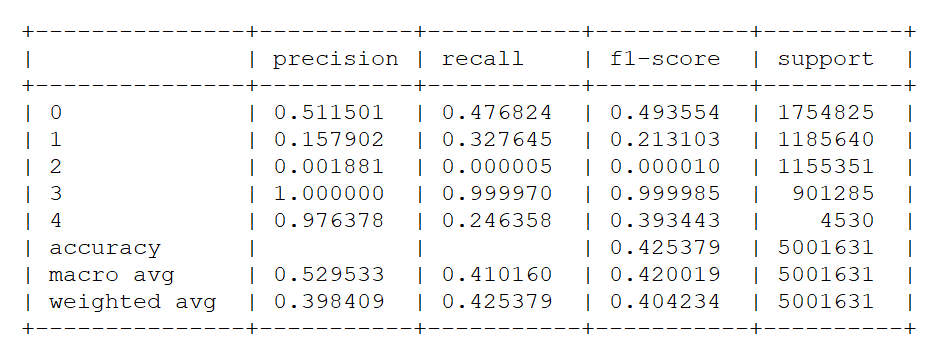


Ma trận nhầm lẫn mô hình Random Forest

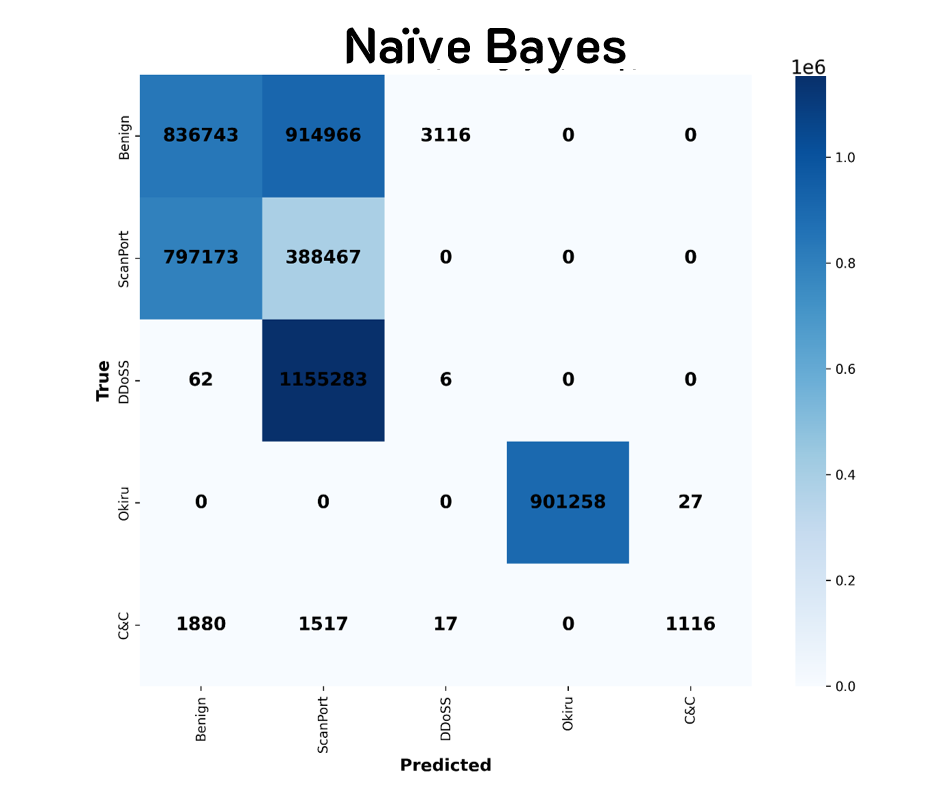


Đánh giá mô hình Random Forest qua đường cong ROC của độ đo AUC

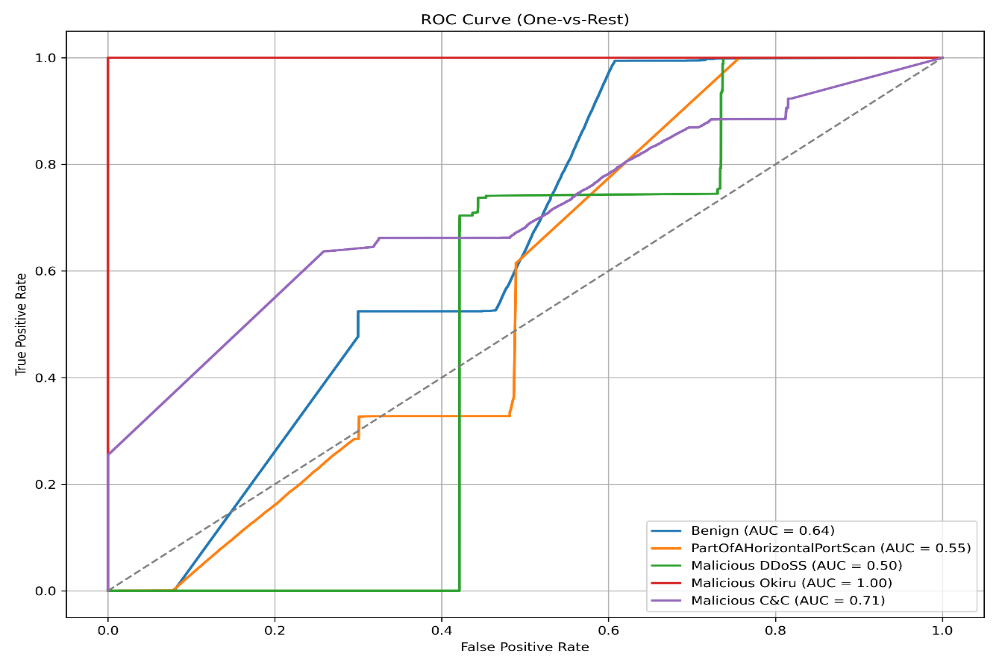
**Naïve Bayes:** Có độ chính xác 42.53%



Đánh giá độ chính xác của mô hình Naïve Bayes



Ma trận nhầm lẫn mô hình Naïve Bayes



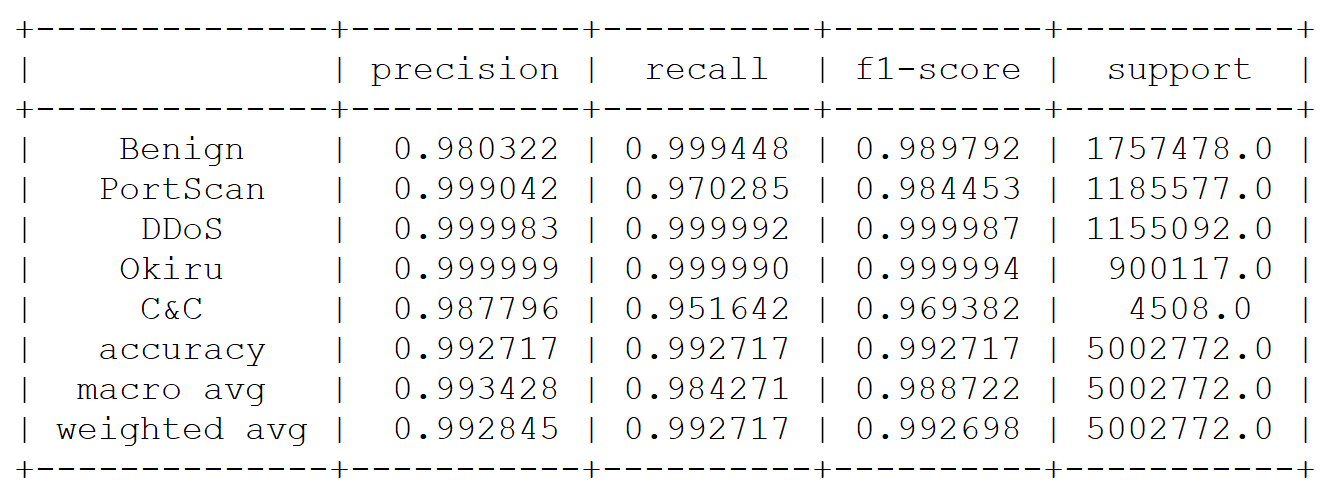
Đánh giá mô hình Naïve Bayes qua đường cong ROC của độ đo AUC

#### 4.3.2.2. Thực nghiệm trích xuất đặc trưng

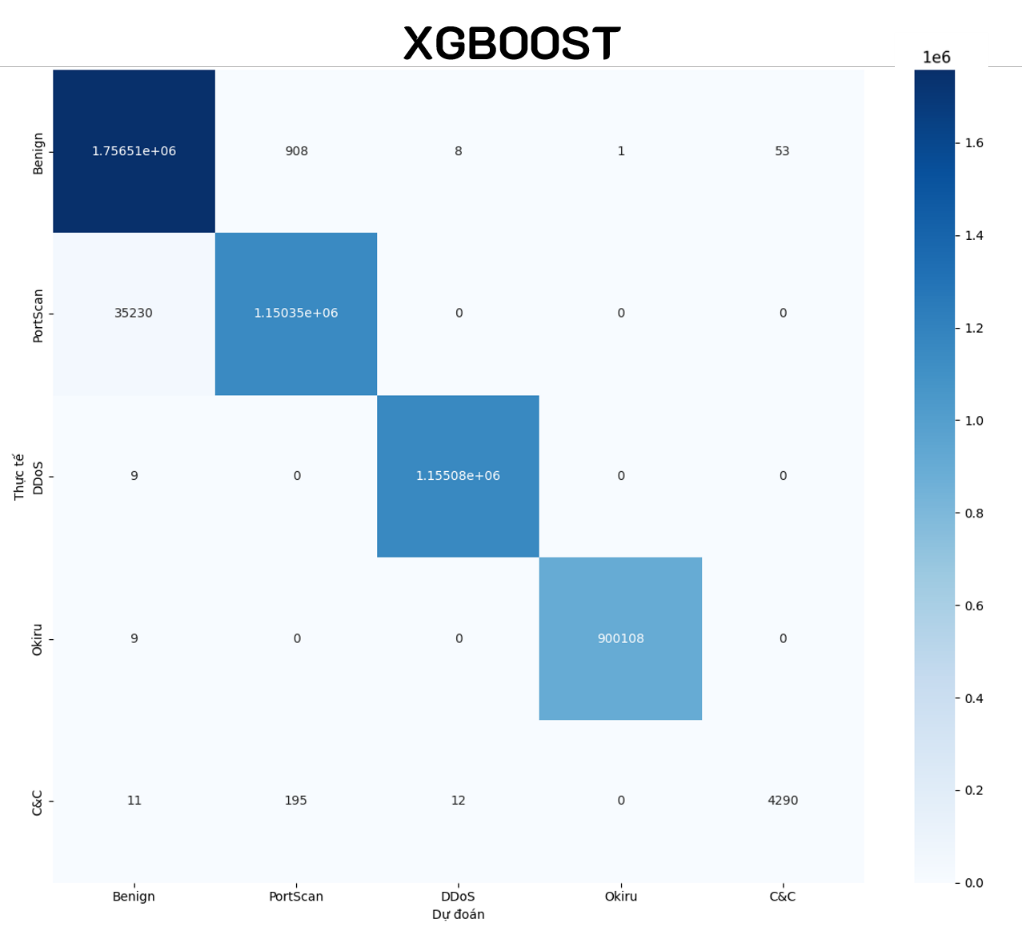
Kết quả huấn luyện của 4 mô hình được trình bày trong hình dưới

Chúng tôi trình bày ma trận nhầm lẫn của dự đoán từ các mô hình phân lớp 2 lớp trên bộ thử nghiệm (Testing set) trong các hình bên dưới

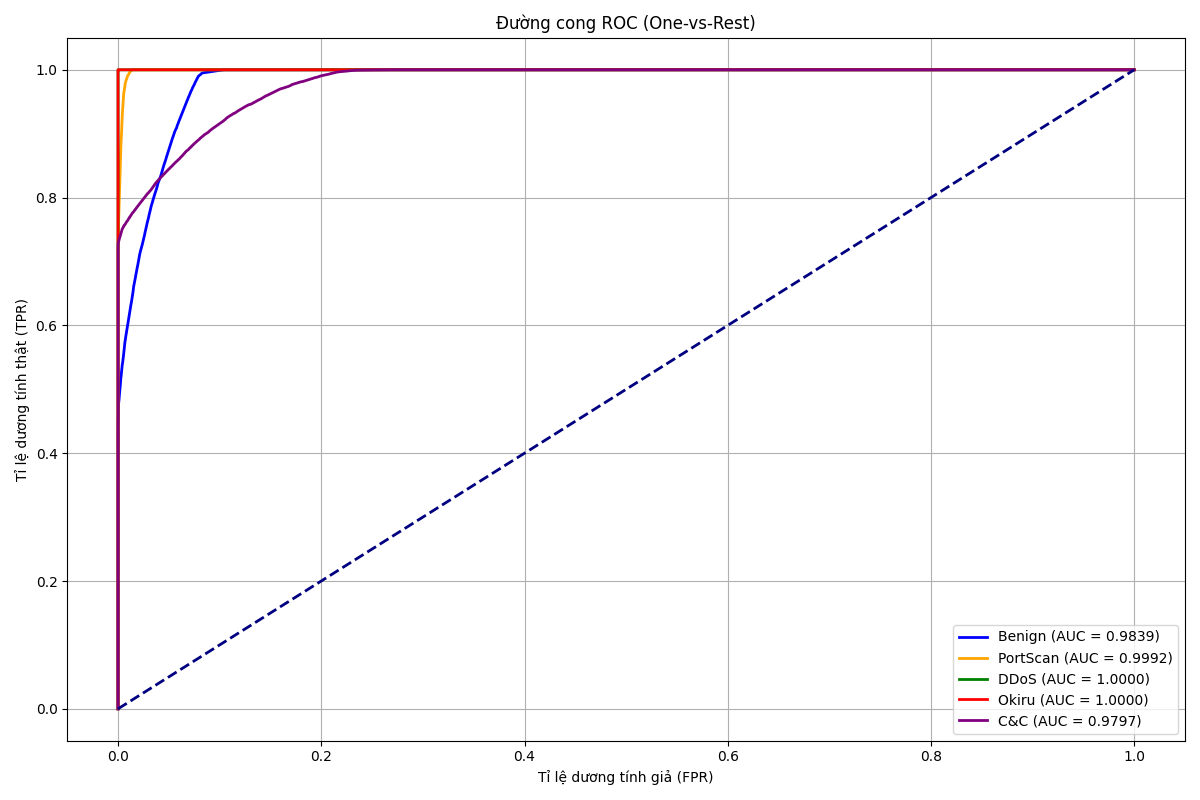
**XGBoost:** Có độ chính xác 99.27%



Đánh giá độ chính xác của mô hình XGBoost

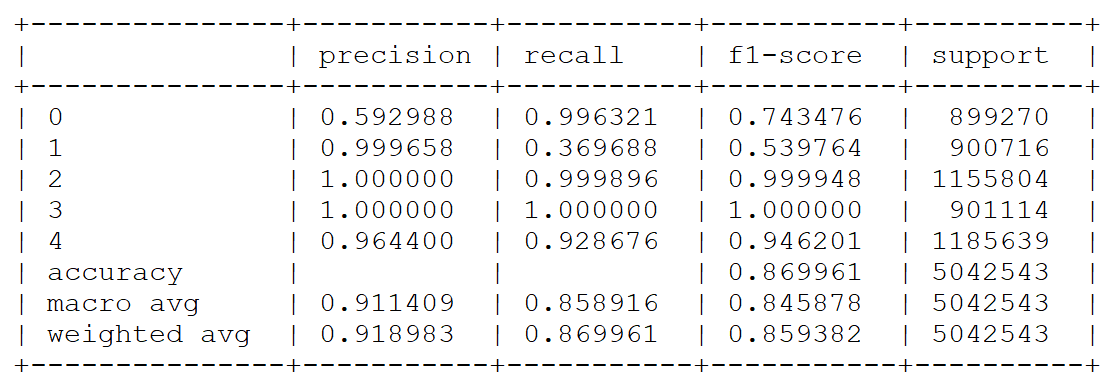


Ma trận nhầm lẫn mô hình XGBoost

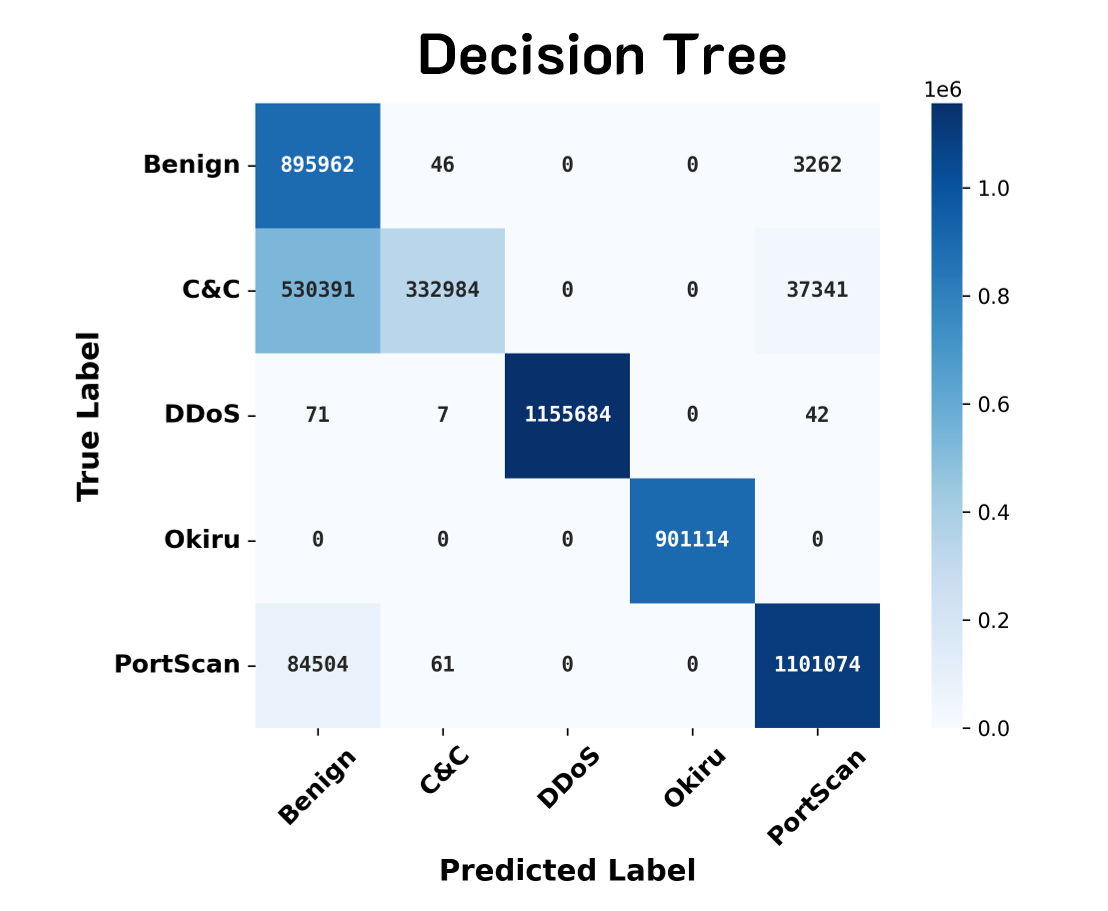


Đánh giá mô hình XGBoost qua đường cong ROC của độ đo AUC

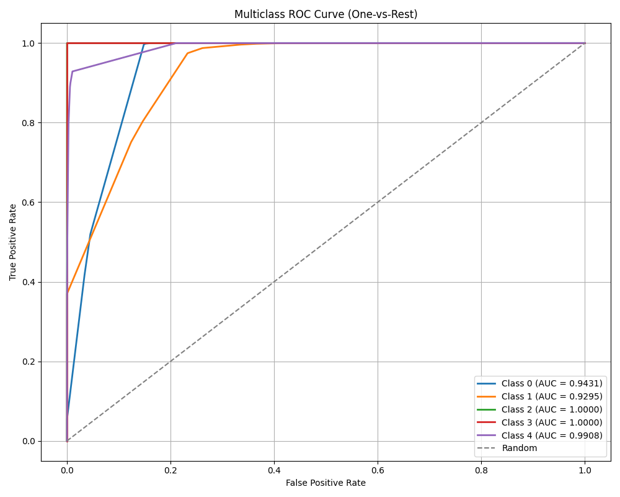
**Decision Tree:** Có độ chính xác 86.99%



Đánh giá độ chính xác của mô hình Decision Tree

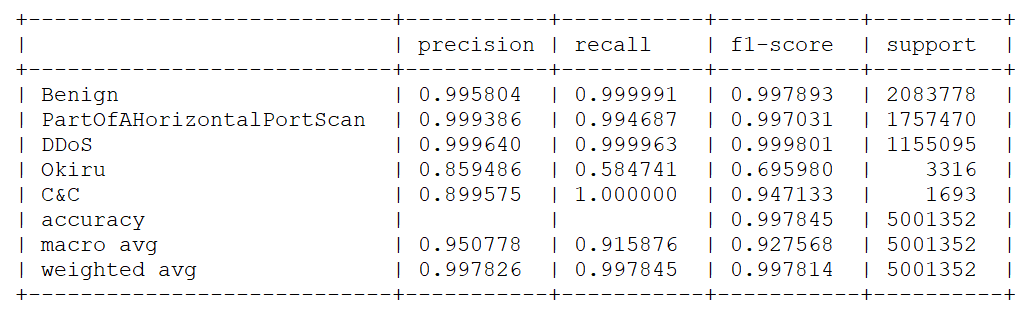


Ma trận nhầm lẫn mô hình Decision Tree

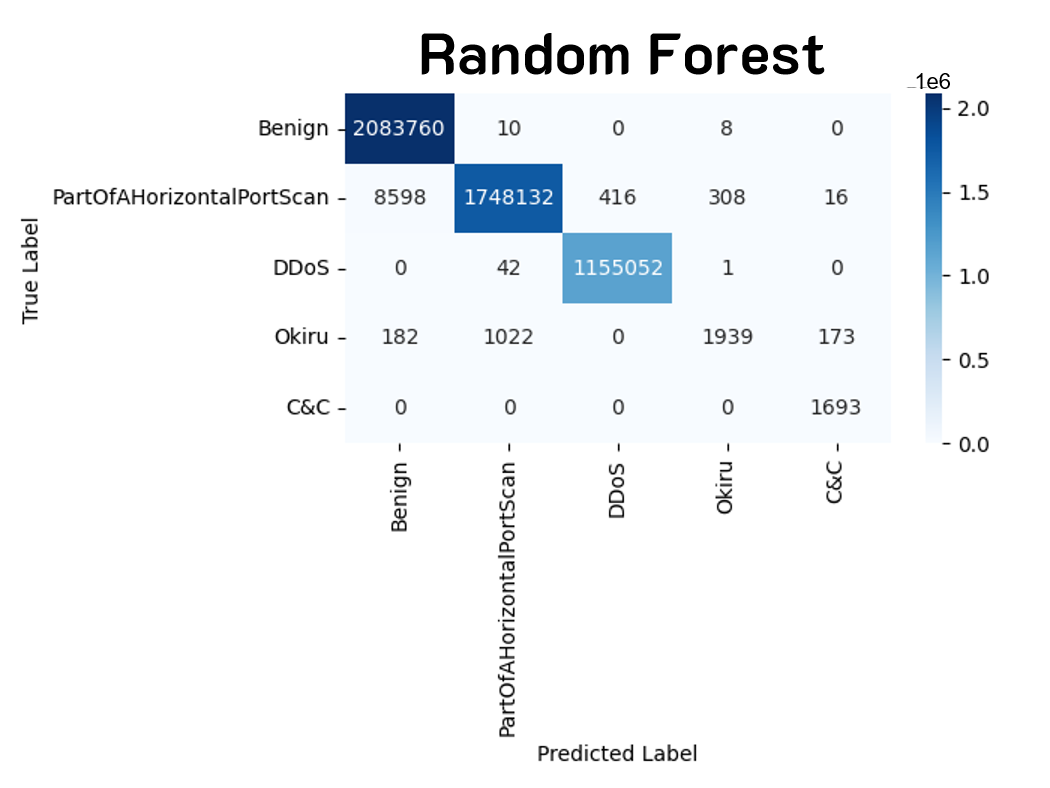


Đánh giá mô hình Decision Tree qua đường cong ROC của độ đo AUC

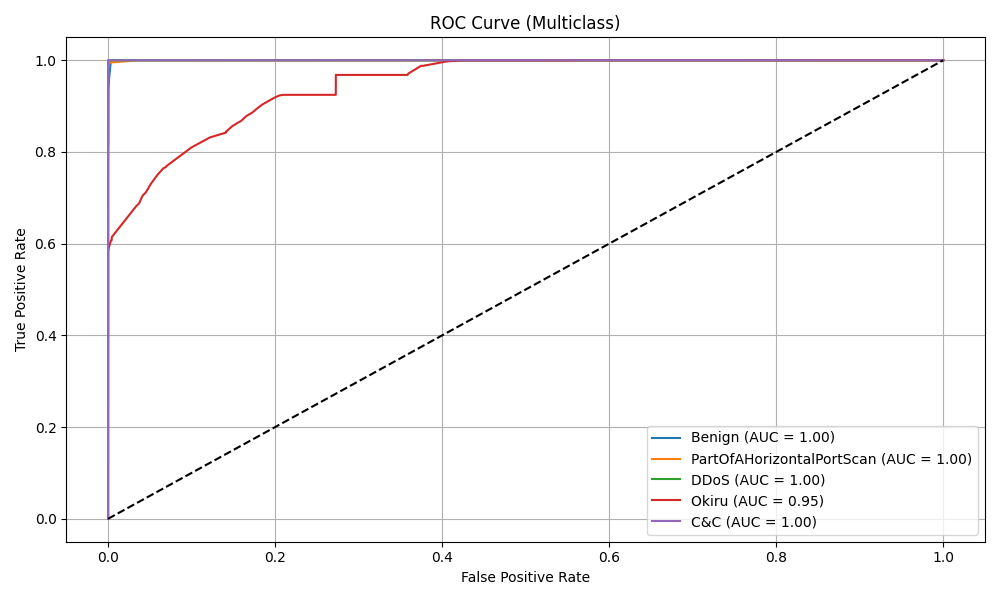
**Random Forest:** Có độ chính xác 99.78%



Đánh giá độ chính xác của mô hình Random Forest

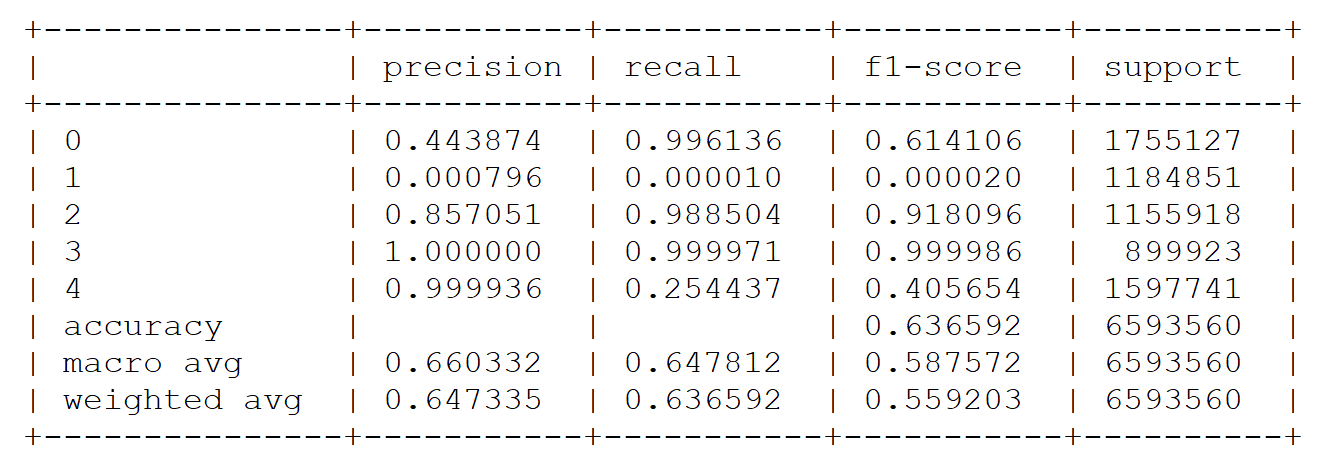


Ma trận nhầm lẫn mô hình Random Forest

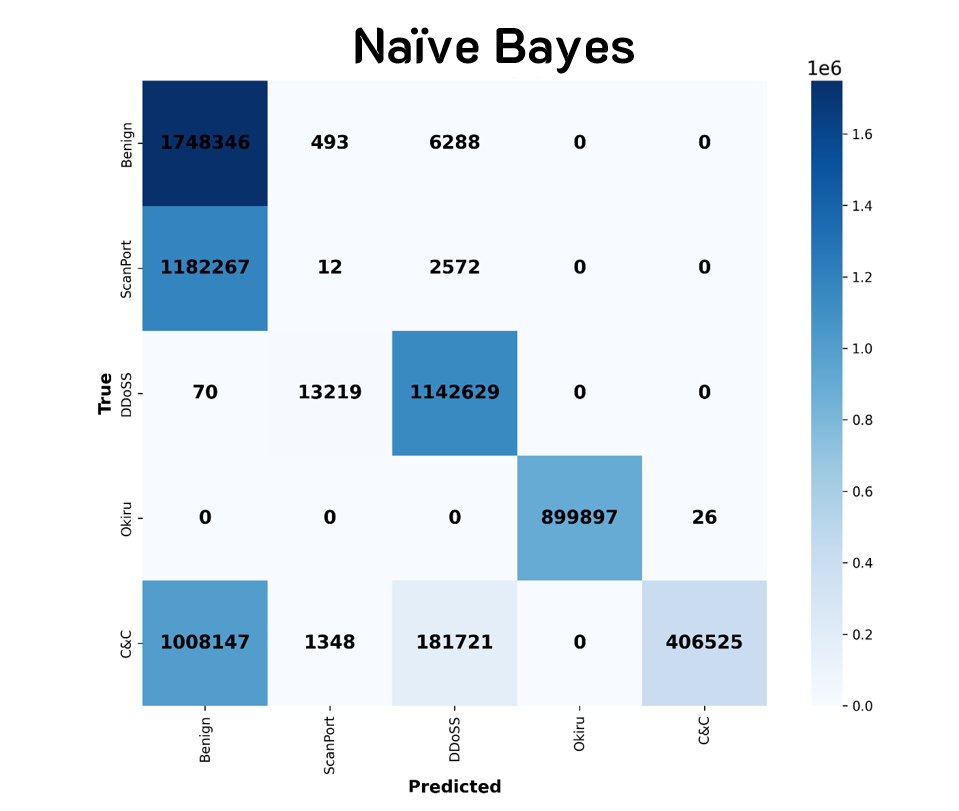


Đánh giá mô hình Random Forest qua đường cong ROC của độ đo AUC

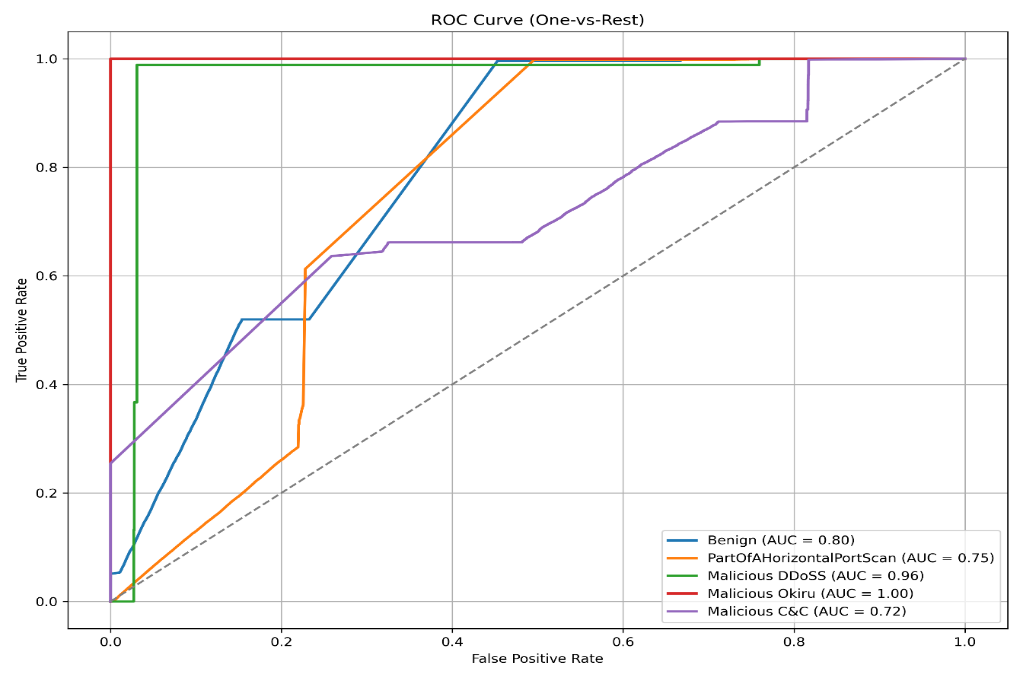
**Naïve Bayes:** Có độ chính xác 63.65%



Đánh giá độ chính xác của mô hình Naïve Bayes



Ma trận nhầm lẫn mô hình Naïve Bayes



Đánh giá mô hình Naïve bayes qua đường cong ROC của độ đo AUC

**4.3.3. Mô hình học sâu GRU (Gated Recurrent Unit)**

# CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 5.1. Kết quả đạt được

Chúng tôi thực hiện một loạt các thử nghiệm trên các mô hình khác nhau, bao gồm XGBoost, Decision Tree, Random Forest và Naïve Bayes. Chúng tôi chia thành hai nhóm thừ nghiệm.

Nhóm thứ nhất tập trung vào huấn luyện các mô hình phân loại nhị phân với hai lớp chính: "Benign" và "Malware". Sau đó, chúng tôi tiến hành trích chọn các đặc trưng quan trọng đối với quyết định của mô hình và tiến hành huấn luyện lại các mô hình trên tập các đặc trưng này.

Nhóm thứ hai thực hiện huẩn luyện các mô hình phân loại đa lớp với một lớp “Benign” và 4 lớp khác tương ứng với từng loại tấn công. Cũng như trên, chúng tôi tiếp tục thực hiện huấn luyện lại các mô hình trên tập các đặc trưng quan trọng được trích lọc.

Kết quả từ hai nhóm thử nghiệm đã thế hiện rằng mô hình được huấn luyện dựa trên XGBoost đạt dược kết quả tốt nhất. Dựa trên mô hình này, chúng tôi tiến hành sử dụng hai phương pháp diễn giải là LIME và SHAP (Shapley Additive exPlanations) để giải thích các kết quả dự đoán trên từng luồng mạng trong tập test, nhằm hiểu rõ hơn về quyết định của mô hình và cách nó dự đoán các sự kiện xâm nhập.

## 5.2. Hướng phát triển

Dựa trên kết quả đã trình bày, dưới đây là một số hạn chế quan trọng của nghiên cứu:

Sự hạn chế về Dữ Liệu: Hiệu suất của mô hình phân loại và khả nãng diễn giải phụ thuộc rất nhiều vào chất lượng và tính đại diện của dữ liệu huấn luyện. Trong nghiên cứu này, hiệu suất mô hình và khả năng diễn giải được đánh giá trên một tập dữ liệu cụ thể. Khả năng áp dụng nghiên cứu này cho các tập dữ liệu khác có thể bị hạn chế và yêu cầu xem xét kỹ lưỡng. Đặc biệt trong kỷ nguyên số lượng dữ liệu được tạo ra mỗi giây rất lớn cùng với đó là sự đa dạng trong các đợt tấn công.

Sự hạn chế về Mô Hình: Dù đã thử nghiệm nhiều mô hình khác nhau, nghiên cứu này chỉ tập trung vào một số lượng hữu hạn các mô hình phân loại. Sự lựa chọn của mô hình có thể không áp dụng hoàn toàn cho mọi tình huống và cần cân nhắc kỹ lưỡng khi áp dụng cho các tình huống khác.

Những hạn chế được trình bày ở phần trên cần được xem xét khi áp dụng và hiểu kết quả của nghiên cứu này, và nên đặt ra cơ hội cho các nghiên cứu tương lai để giải quyết những thách thức này và mở rộng sự hiểu biết về các mô hình học máy.

# TÀI LIỆU THAM KHẢO

[1] **Piragash Maran, Timothy Tzen Vun Yap, JiJianChin, HuNg, VikTorGoh, and Thiam Yong Kuek**. *Comparison of Machine Learning Models for IoT Malware Classification*

[2] **Nguyen Van Binh.**  *Phát hiện xâm nhập mạng bằng phương pháp học máy có thể giải thích.* Luận văn thạc sĩ ngành Khoa học máy tính, Trường Đại học Cần Thơ (2023)

[3] **S Lagraa, J François**. *Knowledge discovery of port scans from darknet*. 2017 IFIP/IEEE Symposium on…, 2017.

[4] **SK Patel, A Sonker**. *Rule-based network intrusion detection system for port scanning with efficient port scan detection rules using Snort*. International Journal of Future Generation, 2016.

[5] **Wickramasinghe, I., & Kalutarage, H. (2021).** Naïve Bayes: Applications, variations, and vulnerabilities: A review of literature with code snippets for implementation. *Soft Computing*.

[6] **Prokhorov, S., & Safronov, V. (2019).** AI for AI: What NLP techniques help researchers find the right articles on NLP. *IEEE Conference on Artificial Intelligence Applications*.

[7] **Bitdefender Vietnam**. *What is Malware ?* – Bitdefender Vietnam, n.d.

[8] **MS Kumar, J Ben-Othman**. *Artificial intelligence managed network defense system against port scanning outbreaks*. IEEE Conference on Vision Towards..., 2019.