ĐẠI HỌC QUỐC GIA HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

TRẦN QUỐC ĐẠT
NGUYỄN MINH HUY
ĐOÀN QUỐC DƯƠNG

ĐỒ ÁN CHUYÊN NGÀNH

XÂY DỰNG MÔ HÌNH TƯỜNG LỬA ỨNG DỤNG WEB SỬ DỤNG MULTI-MODAL

DEVELOPING A WEB APPLICATION FIREWALL MODEL USING MULTI-MODAL APPROACH

KỸ SƯ NGÀNH AN TOÀN THÔNG TIN

TP. Hồ Chí Minh, 2023

ĐẠI HỌC QUỐC GIA HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

TRẦN QUỐC ĐẠT - 20521179
NGUYỄN MINH HUY - 20520545
ĐOÀN QUỐC DƯƠNG - 17520831

ĐỒ ÁN CHUYÊN NGÀNH

XÂY DỰNG MÔ HÌNH TƯỜNG LỬA ỨNG DỤNG WEB SỬ DỤNG MULTI-MODAL

DEVELOPING A WEB APPLICATION FIREWALL MODEL USING MULTI-MODAL APPROACH

KỸ SƯ NGÀNH AN TOÀN THÔNG TIN

GIẢNG VIÊN HƯỚNG DẪN:
ThS. Nghi Hoàng Khoa
ThS. Nguyễn Công Danh

TP.Hồ Chí Minh - 2023

LỜI CẢM ƠN

Trong quá trình nghiên cứu và hoàn thành đồ án, nhóm đã nhận được sự định hướng, giúp đỡ, các ý kiến đóng góp quý báu và những lời động viên của các giáo viên hướng dẫn và giáo viên bộ môn. Nhóm xin bày tỏ lời cảm ơn tới thầy Nguyễn Công Danh, thầy Nghi Hoàng Khoa đã tận tình trực tiếp hướng dẫn, giúp đỡ trong quá trình nghiên cứu.

Nhóm xin gửi lời cảm ơn đến gia đình và bạn bè đã động viên, đóng góp ý kiến trong quá trình làm đồ án

Nhóm cũng chân thành cảm ơn các quý thầy cô trường Đại học Công nghệ Thông tin - ĐHQG TP.HCM, đặc biệt là các thầy cô khoa Mạng máy tính và Truyền thông, các thầy cô thuộc bộ môn An toàn Thông tin đã giúp đỡ nhóm.

Trần Quốc Đạt **Nguyễn Minh Huy** **Đoàn Quốc Dương**

MỤC LỤC

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

HYDRA-WAFS	Web appliation firewall based multimodal HYDRA
ACC	Accuracy metric
PRE	Precision metric
RECALL	Recall metrics
HTTP/HTTPS	Hypertext Transfer Protocol (Secure)

DANH MỤC CÁC HÌNH VẼ

DANH MỤC CÁC BẢNG BIỂU

TÓM TẮT ĐỀ TÀI

Tính cấp thiết của đề tài nghiên cứu:

Các ứng dụng web thường phải đối mặt với các cuộc tấn công vì thông tin quan trọng và tài sản có giá trị mà chúng lưu trữ. Bảo vệ các ứng dụng web đang trở nên thách thức mỗi ngày, chủ yếu là do sự tinh vi của các cuộc tấn công, sự hiện diện toàn diện của các ứng dụng web và sự phụ thuộc quá mức vào Tường lửa ứng dụng web truyền thống (WAF). Các mô hình tường lửa ứng dụng web truyền thống hoạt động dựa trên các chính sách dần bộc lộ những yếu điểm của nó như việc phải cập nhật chính sách liên tục và liên kết với nhau để tránh việc tạo ra lỗ hổng giữa các chính sách cũng như chính sách lỏng lẻo dẫn đến việc bị khai thác dễ dàng. Khác với mô hình chính sách, mô hình học máy dựa vào các quy luật của đầu vào và đầu ra được cung cấp. Các công trình gần đây bao gồm các hệ thống phát hiện tấn công web đã cho thấy khả năng tuyệt vời để bảo vệ các ứng dụng web. Tuy nhiên, các hệ thống này đã phải đối mặt với những thách thức nghiêm trọng. Do đó, nhu cầu cấp thiết phải nghiên cứu các hệ thống tiến bộ hơn để bảo vệ các ứng dụng web khỏi các cuộc tấn công web khác nhau. Khi các cuộc tấn công web phát triển nhanh chóng về mức độ tinh vi, các nhà nghiên cứu về an ninh mạng đang tích cực khám phá các công nghệ bảo mật mới dựa trên học sâu (Deep Learning) như mạng nơ-ron tích chập (Convolutional Neural Network - CNN), mạng nơ-ron hồi quy (Recurrent Neural Network - RNN), mạng nơ-ron tái cấu trúc (Recurrent Neural Network - RNN),... Các ứng dụng học sâu, dựa trên phân tích dữ liệu lớn, cho thấy khả năng vượt trội trong việc phát hiện sự bất thường thông qua lưu lượng truy cập lớn. Các giải pháp học sâu này đã giúp thúc đẩy và tạo điều kiện phát triển bảo mật cho các ứng dụng web. Trong đồ án này, nhóm đề xuất một Hệ thống phát hiện tấn công web mới (HYDRA-WAFS), dựa trên học sâu đa phương thức

(Multimodal Deep Learning) nhằm tận dụng các kết quả từ nhiều mô hình khác nhau với mục đích nâng cao kết quả phân loại cuối cùng. Kết quả thực nghiệm cho thấy mô hình mà nhóm đề xuất có tính phân loại tốt với độ chính xác cao và mô hình hoàn toàn có khả năng ứng dụng trong môi trường thực tế.

CHƯƠNG 1. TỔNG QUAN

Chương này giới thiệu về vấn đề và các nghiên cứu liên quan. Đồng thời, trong chương này nhóm cũng trình bày phạm vi và cấu trúc của Đồ án.

1.1. Giới thiệu vấn đề

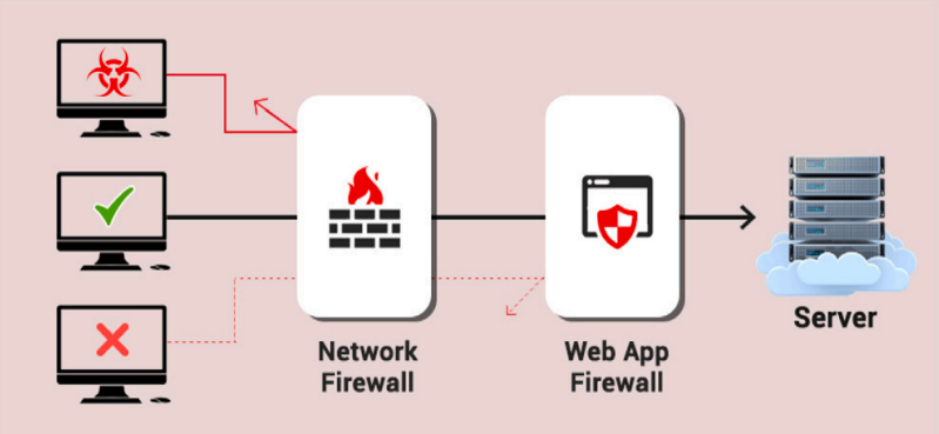
Hiện nay, một trong những đối tượng được phát triển rộng rãi và sử dụng phổ biến trên mạng internet là các nền tảng các trang web trực tuyến bởi tầm quan trọng của chúng. Vì thế, nó cũng chính là đối tượng dễ dàng cho các mối đe dọa khai thác và gây ra những thiệt hại khó có thể khôi phục đồng thời thông tin rò rỉ để lại hậu quả về sau. Các ứng dụng web thường phải đối mặt với các cuộc tấn công vì thông tin quan trọng và tài sản có giá trị mà chúng lưu trữ. Các cuộc tấn công nguy hiểm như SQL injection (SQLi), cross-site scripting (XSS) hay local file inclusion (LFI),... có thể gây hại đối với các ứng dụng web. Do các cuộc tấn công web này, dữ liệu quan trọng có thể bị lộ, hệ thống có thể bị tấn công và vi phạm đáng kể đến quyền riêng tư hoặc tổn thất tài chính, mất mát người dùng có thể xảy ra. Vì nhiều giao dịch kinh tế và quan trọng được thực hiện thông qua các ứng dụng web, số lượng tin tặc tấn công các ứng dụng này đang tăng lên hàng ngày, với các cuộc tấn công trở nên tinh vi hơn.

Các công trình gần đây bao gồm các hệ thống phát hiện tấn công web đã cho thấy khả năng tuyệt vời để bảo vệ các ứng dụng web. Tuy nhiên, các hệ thống này đã phải đối mặt với những thách thức nghiêm trọng. Do đó, nhu cầu cấp thiết phải nghiên cứu các hệ thống tiến bộ hơn để bảo vệ các ứng dụng web khỏi các cuộc tấn công web khác nhau. Khi các cuộc tấn công web phát triển nhanh chóng về mức độ tinh vi, các nhà nghiên cứu về an ninh mạng đang tích cực khám phá các công nghệ bảo mật mới dựa trên học sâu [11, 5, 9].Trong khi

các công nghệ phát hiện tấn công web truyền thống cho thấy điểm yếu trong môi trường dữ liệu lớn, sự gia tăng của học sâu cung cấp các giải pháp mới cho các vấn đề bảo mật trong các môi trường như vậy. Các ứng dụng học sâu, dựa trên phân tích lượng dữ liệu lớn, cho thấy khả năng vượt trội trong việc phát hiện sự bất thường thông qua lưu lượng truy cập lớn. Các giải pháp học sâu này đã giúp thúc đẩy và tạo điều kiện phát triển bảo mật các ứng dụng web. Môi trường dữ liệu lớn được sử dụng trong nhiều mô hình triển khai học máy cũng như kiểm thử để tái hiện lại mô hình hoặc một phần tấn công hoặc tiềm năng tấn công.

Sau khi thực nghiệm lại các nghiên cứu nêu trên, nhóm nhận thấy hầu hết các hệ thống WAF truyền thống đều tồn tại những điểm yếu nhất định ở việc chính sách phải được cập nhật liên tục và kiểm thử chặt chẽ trước khi triển khai rộng rãi. Vì vậy, nhu cầu cấp thiết đặt ra là nhóm cần phải tích hợp thêm mô hình học máy ứng dụng multimodal vào mã nguồn có sẵn của WAF như Modsecurity nhằm tăng cường thêm tính bảo mật. Việc tích hợp thêm mô hình học máy ứng dụng multimodal là tất yếu vì sự kết hợp được dữ liệu từ nhiều nguồn khác nhau để đưa ra kết quả cuối cùng.

Trong đồ án này, nhóm đề xuất một Hệ thống phát hiện tấn công web (HYDRA-WADS) mới, dựa trên học sâu đa phương thức. Cụ thể, HYDRA-WADS được đề xuất tận dụng các mô hình học sâu [3] để phân tích các trường khác nhau trong yêu cầu HTTP/HTTPS và xác định các yêu cầu bất thường trong ba nhóm thuộc tính. Theo cách tiếp cận của nhóm, ba mô hình học sâu được sử dụng để mỗi mô hình tìm hiểu các thuộc tính tương đối ẩn trong các truy vấn. Nhóm sử dụng các phương pháp khác nhau để xử lý và chuyển đổi các trường yêu cầu này thành các loại biểu diễn khác nhau để khai thác lợi thế từ nhiều mô hình học sâu khác nhau. Hơn nữa, nhóm sử dụng một bộ phân loại đa phương thức để tiến hành phân tích toàn diện kết quả của ba mô hình học sâu này. Bộ phân loại đa phương thức được thiết kế để cho phép HYDRA-WADS khắc phục điểm yếu của các bộ phân loại riêng lẻ và kết hợp các ưu điểm của



Hình 1.1: *Tường lửa ứng dụng web truyền thống.*

chúng để cải thiện hiệu suất phát hiện.

1.2. Giới thiệu những nghiên cứu liên quan

1.2.1. Những thách thức đối với WAF truyền thống

Mô hình WAF truyền thống thông thường sử dụng những chính sách (rules) đã được định nghĩa từ trước. Nó là lớp phòng thủ đứng giữa kẻ tấn công và Webserver. WAF phân tích các yêu cầu của người dùng để lọc lưu lượng truy cập bất hợp pháp ra khỏi lưu lượng truy cập thông thường. Phân tích các yêu cầu HTTP này dựa trên một bộ quy tắc được xác định trước được thiết kế để phân biệt lưu lượng độc hại. Tuy nhiên những chính sách của WAF cũng rất dễ bị bypass bởi những kẻ tấn công nguy hiểm.

1.2.2. Phương pháp học máy truyền thống trong phát hiện tấn công web

Phát hiện tấn công web dựa trên sự bất thường là một lĩnh vực trọng tâm quan trọng trong nghiên cứu bảo mật web. Một trong những nghiên cứu đầu tiên được thực hiện bởi Krueger và Vigna vào năm 2003, sử dụng các mô hình Markov. Nghiên cứu này vẫn quan trọng vì đây là nghiên cứu đầu tiên về phát

hiện tấn công web dựa trên sự bất thường trong tài liệu và nó tập trung vào việc trích xuất các thuộc tính được sử dụng trong các cuộc tấn công web

Nhóm tiếp cận và tham khảo các bài báo [19, 6, 21, 12, 14] trước trong việc triển khai mô hình học máy nhằm phát hiện tấn công web bằng nhiều thuật toán khác nhau như SVM, Random Forest, Naive Bayes, ...

Khi phân tích các nghiên cứu trên, nhóm nhận thấy một số thách thức lớn trong việc phát triển các mô hình WAF dựa trên các thuật toán học máy truyền thống. Một trong những thách thức quan trọng nhất là quá trình lựa chọn thuộc tính. Sự thành công của các thuật toán học máy truyền thống phụ thuộc vào quá trình lựa chọn thuộc tính đưa vào [16].

Các thuật toán học sâu có thể tạo ra các mô hình phân loại mạnh mẽ và hiệu quả bằng cách sử dụng các bộ dữ liệu bao gồm một lượng lớn dữ liệu mà không cần lựa chọn thuộc tính [1].

1.2.3. Phương pháp học sâu trong phát hiện tấn công web

Các nghiên cứu sử dụng phương pháp học sâu trong phát hiện tấn công Web chỉ mới được xuất bản từ năm 2015, đây là một lĩnh vực nghiên cứu mới [9]. Những nghiên cứu này cho thấy nhiều phương pháp học sâu đã được sử dụng thành công để phát hiện tấn công web. Hầu hết các nghiên cứu áp dụng phương pháp học sâu trong lĩnh vực bảo mật web đều có từ năm 2017-2020 [2].

Một số hệ thống phát hiện xâm nhập web sử dụng CNN được đề xuất trong tài liệu [1, 20, 23]. Trong các nghiên cứu trên, [20] sử dụng thuật toán CNN và word embedding, các thuộc tính được trích xuất từ payload trong các yêu cầu web, nghiên cứu [20], [23] sử dụng các phương pháp tương tự, tuy nhiên thuộc tính được trích xuất từ các phần URL. Tương tự, các tác giả trong [4] đã phát triển một mô hình WAF học sâu bằng cách sử dụng phương pháp nhúng word2vec và kiến trúc Bi-LSTM. Trong tất cả bốn nghiên cứu trong [1], [20], [23], [4], bộ dữ liệu CSIC-2010 được sử dụng và tỷ lệ phát hiện lần lượt là 96,13% [1], 97,07% [20], 96,49% [23] và 98,35% [4]. Các tác giả trong [7] sử dụng phương

pháp tăng cường dữ liệu DA-SANA đạt được độ chính xác trên tập CSIC-2010 và ECML/PKDD 2007 lần lượt là 96,95% và 99,06%. Tác giả của bài báo [18] sử dụng 2 thuật toán LightGBM và CatBoost cho kết quả phát hiện lần lượt là 88,09% và 88,44% trên tập dataset SR-BH 2020.

Nhìn vào các nghiên cứu nêu trên, cho thấy rằng các phương pháp thường đối phó với một [24], [6], [13] hoặc hai loại tấn công [21], [10] (tấn công SQLi và, hoặc XSS). Nghiên cứu [18] tuy phân loại được nhiều loại tấn công nhưng hiệu quả phát hiện còn thấp.

1.3. Mục tiêu, đối tượng, và phạm vi nghiên cứu

1.3.1. Mục tiêu nghiên cứu

Đề tài sẽ triển khai hệ thống phát hiện tấn công Web (HYDRA-WADS) ứng dụng multimodal để phân loại các kỹ thuật tấn công bao gồm SQLi, XXE, LFI, Command injection, XSS, Server Side Include (SSI),... Cụ thể, HYDRA-WADS mà nhóm đề xuất tận dụng các mô hình deep learning [3] để phân tích các đặc trưng khác nhau của các yêu cầu và xác định các yêu cầu bất thường dựa trên ba nhóm đặc trưng.

1.3.2. Đối tượng nghiên cứu

Đối tượng nghiên cứu:

- Cơ sở của HTTP Request
- Các loại tấn công Web
- Hệ thống phát hiện tấn công Web dựa trên học sâu
- Mô hình học sâu ứng dụng multimodal

1.3.3. Phạm vi nghiên cứu

Xây dựng và đào tạo mô hình học sâu dựa trên các tập dataset sẵn có và đánh giá hiệu quả của mô hình dựa trên tập dataset tương ứng.

Đào tạo các mô hình học sâu, tiến hành triển khai chúng vào môi trường thực nghiệm trên máy ảo. Giai đoạn này gồm các bước sau: 1) Trích xuất thuộc tính của 1 HTTP/HTTPS request và tiến hành xử lý, phân tích chúng tạo ra các mẫu input cho mô hình học sâu 2) Đánh giá hiệu quả phát hiện tấn công trên môi trưởng ảo

1.3.4. Cấu trúc đồ án chuyên ngành

Nhóm xin trình bày nội dung của Đồ án chuyên ngành theo cấu trúc như sau:

- Chương 1: Giới thiệu tổng quan về đề tài của Đồ án và những nghiên cứu liên quan.
- Chương 2: Trình bày cơ sở lý thuyết và kiến thức nền tảng liên quan đến đề tài.
- Chương 3: Trình bày mô hình HYDRA-WAFS ứng dụng multimodal.
- Chương 4: Trình bày thực nghiệm và đánh giá.
- Chương 5: Kết luận và hướng phát triển của đề tài.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Chương này trình bày cơ sở lý thuyết của nghiên cứu: Cấu trúc cơ bản của yêu cầu HTTP, Tổng quan về tấn công Web, Mô hình tường lửa ứng dụng web ứng dụng mô hình học máy/học sâu.

2.1. Cấu trúc cơ bản của yêu cầu HTTP trong trang Web

Yêu cầu HTTP (Hypertext Transfer Protocol) là cách mà trình duyệt web giao tiếp với máy chủ web để lấy và gửi dữ liệu. Một yêu cầu HTTP bao gồm các thành phần cơ bản sau:

1. **Phương thức (Method):** Đây là phần quan trọng nhất của yêu cầu HTTP và xác định loại hành động mà trình duyệt muốn thực hiện trên máy chủ. Một số phương thức phổ biến bao gồm:
   - GET: Lấy dữ liệu từ máy chủ.
   - POST: Gửi dữ liệu đến máy chủ để xử lý.
   - PUT: Cập nhật hoặc thay thế một tài nguyên trên máy chủ.
   - DELETE: Xóa một tài nguyên trên máy chủ.
2. **Đường dẫn (Path):** Đường dẫn xác định tài nguyên hoặc API mà trình duyệt muốn truy cập trên máy chủ. Đường dẫn bắt đầu sau URL gốc và có thể chứa các tham số và giá trị.
3. **Phiên bản giao thức (Protocol Version):** Xác định phiên bản giao thức HTTP được sử dụng trong yêu cầu. Ví dụ: HTTP/1.1.

4. **Tiêu đề (Headers):** Là các thông tin bổ sung đi kèm với yêu cầu HTTP. Các tiêu đề cung cấp thông tin như User-Agent (trình duyệt được sử dụng), Accept (định dạng dữ liệu được chấp nhận), và nhiều thông tin khác.

5. **Thân (Body):** Chỉ xuất hiện trong các yêu cầu POST hoặc PUT. Thân chứa dữ liệu cần gửi đến máy chủ, ví dụ như dữ liệu biểu mẫu được nhập bởi người dùng.

Ví dụ một yêu cầu HTTP cơ bản:

```
GET /api/products HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/88.0.4324.182 Safari/537.36
Accept: application/json
```

Trong ví dụ trên, phương thức là GET và đường dẫn là "/api/products". Phiên bản giao thức HTTP là 1.1. Các tiêu đề bao gồm Host, User-Agent và Accept.

Qua cấu trúc này, trình duyệt web có thể gửi yêu cầu đến máy chủ web và nhận phản hồi tương ứng để hiển thị nội dung trên trang.

2.2. Tổng quan về tấn công web

Tấn công web là quá trình xâm nhập vào hệ thống thông qua các lỗ hổng bảo mật trong ứng dụng web. Do các yêu cầu HTTP/HTTPS gửi bởi ứng dụng người dùng có thể bị can thiệp và chỉnh sửa bởi người dùng nên nó là đối tượng lý tưởng cho kẻ tấn công khai thác vào để chuộc lợi. Các cuộc tấn công web thường tập trung vào việc khai thác các lỗ hổng phần mềm, thiếu kiểm soát đầu vào và các vấn đề bảo mật khác. Dưới đây là một số kỹ thuật và vector tấn công phổ biến trong tấn công web:

SQL Injection: Kỹ thuật này tận dụng các lỗ hổng SQL trong ứng dụng web để chèn mã SQL độc hại vào các truy vấn SQL được thực thi trên cơ sở dữ liệu. Các từ khóa trong yêu cầu HTTP nhằm nhận biết cuộc tấn công: SELECT; UNION; FROM; ...

Cross-Site Scripting (XSS): Kỹ thuật XSS cho phép kẻ tấn công chèn mã độc hại vào các trang web hoặc ứng dụng web khác để tấn công người dùng cuối. Khi người dùng truy cập vào các trang web bị tấn công, mã độc hại sẽ được thực thi trong trình duyệt của họ, cho phép kẻ tấn công đánh cắp thông tin người dùng, thay đổi nội dung trang web hoặc thực hiện các hành động không mong muốn khác. Các từ khóa trong yêu cầu HTTP nhằm nhận biết cuộc tấn công: script; fetch; img; src;...

Cross-Site Request Forgery (CSRF): Tấn công CSRF xảy ra khi kẻ tấn công đánh lừa người dùng thực hiện các hành động không mong muốn trong ứng dụng web mà họ đã đăng nhập. Điều này thường xảy ra bằng cách chèn các yêu cầu HTTP giả mạo trong các trang web hoặc email mà người dùng truy cập.

Rò rỉ thông tin: Tấn công rò rỉ thông tin xảy ra khi kẻ tấn công tìm cách thu thập thông tin nhạy cảm từ ứng dụng web hoặc cơ sở dữ liệu bằng cách tận dụng các lỗ hổng bảo mật hoặc thiết lập không chính xác.

Remote File Inclusion (RFI) và Local File Inclusion (LFI): Cả RFI và LFI đều là các kỹ thuật tấn công cho phép kẻ tấn công chèn và thực thi mã độc hại từ các tệp tin từ xa hoặc cục bộ vào ứng dụng web. Điều này có thể cho phép kẻ tấn công đọc, sửa đổi hoặc thực thi các tệp tin trên máy chủ web. Các từ khóa trong yêu cầu HTTP nhằm nhận biết cuộc tấn công: `,; /; %2e`

Kỹ thuật chèn mã (Command Injection): Các kỹ thuật chèn mã cho phép kẻ tấn công chèn và thực thi mã độc hại vào các vùng đầu vào trong ứng dụng web, như các biểu mẫu, tham số URL hoặc các tệp tin tải lên. Điều này có thể cho phép kẻ tấn công kiểm soát ứng dụng, thực hiện các hành động không mong muốn hoặc lấy thông tin nhạy cảm từ máy chủ web. Các từ khóa trong yêu cầu HTTP nhằm nhận biết cuộc tấn công: ls; nc; whoami; ..

Các kỹ thuật và hướng khai thác này chỉ là một số ví dụ phổ biến. Tấn công web là một lĩnh vực rộng lớn và không ngừng tiến hóa, vì vậy, việc cập nhật liên tục các phương pháp phòng chống là điều cần thiết.

2.3. Mô hình tường lửa ứng dụng web ứng dụng mô hình học máy/học sâu

Mô hình tường lửa ứng dụng web với ứng dụng mô hình học máy/học sâu là một phương pháp sử dụng công nghệ trí tuệ nhân tạo để phân loại và ngăn chặn các cuộc tấn công trên ứng dụng web. Mô hình này có thể phát hiện và ngăn chặn các hành vi độc hại hoặc không mong muốn từ phía người dùng, giúp bảo vệ hệ thống và dữ liệu của ứng dụng web.

Mô hình tổng quan về quy trình hoạt động của mô hình tường lửa ứng dụng web với ứng dụng mô hình học máy/học sâu được liệt kê bên dưới:

1. Thu thập dữ liệu: Mô hình tường lửa cần có một tập dữ liệu lớn, đa dạng và đại diện cho các cuộc tấn công khác nhau trên ứng dụng web. Dữ liệu này có thể bao gồm các gói tin mạng, các yêu cầu HTTP và các thông tin khác liên quan đến hoạt động của người dùng trên ứng dụng web.

2. Tiền xử lý dữ liệu: Trước khi đưa dữ liệu vào mô hình học máy/học sâu, quá trình tiền xử lý dữ liệu là cần thiết để chuẩn hóa, rút trích đặc trưng và loại bỏ nhiễu. Các bước tiền xử lý có thể bao gồm chuyển đổi định dạng dữ liệu, mã hóa, xử lý chuỗi và rời rạc hóa dữ liệu.

3. Xây dựng mô hình học máy/học sâu: Mô hình học máy/học sâu được sử dụng để xác định các mẫu và quy tắc trong dữ liệu để phân loại các cuộc tấn công và hành vi bất thường trên ứng dụng web. Các mô hình phổ biến có thể sử dụng trong trường hợp này bao gồm Mạng nơ-ron nhân tạo (Artificial Neural Networks), Mạng nơ-ron hồi quy (Recurrent Neural Networks), Mạng nơ-ron tích chập (Convolutional Neural Networks) và Mạng nơ-ron

hệ thống (Deep Neural Networks).

4. Huấn luyện mô hình: Dữ liệu đã được tiền xử lý được sử dụng để huấn luyện mô hình học máy/học sâu. Quá trình huấn luyện bao gồm cung cấp dữ liệu đầu vào và đầu ra mong muốn cho mô hình, và điều chỉnh các tham số mô hình để tối ưu hóa hiệu suất phân loại và nhận dạng.

5. Đánh giá và tinh chỉnh: Mô hình được đánh giá bằng cách sử dụng tập dữ liệu kiểm tra độc lập. Kết quả được so sánh với nhãn thực tế để đánh giá độ chính xác và độ tin cậy của mô hình. Nếu cần thiết, mô hình sẽ được điều chỉnh và tinh chỉnh để cải thiện hiệu suất.

6. Triển khai mô hình: Sau khi mô hình đã được huấn luyện và kiểm tra, nó có thể được triển khai vào một hệ thống tường lửa ứng dụng web thực tế. Mô hình sẽ theo dõi luồng dữ liệu trên ứng dụng web và xác định các cuộc tấn công hoặc hành vi bất thường dựa trên những gì đã học được trong quá trình huấn luyện.

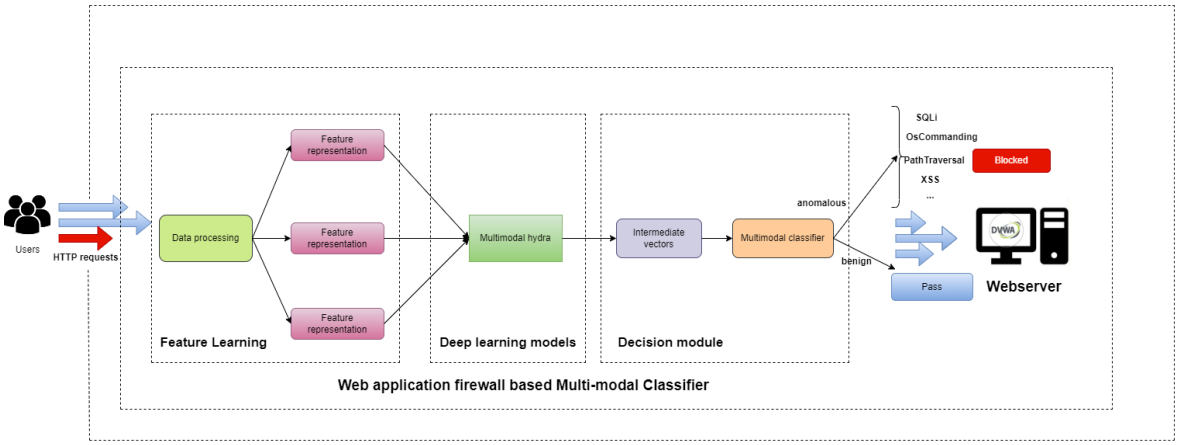
Mô hình tường lửa ứng dụng web với ứng dụng mô hình học máy/học sâu cải thiện khả năng phát hiện và ngăn chặn các cuộc tấn công trên ứng dụng web hiệu quả hơn so với các phương pháp truyền thống. Đồng thời, mô hình học và thích nghi với các dấu hiệu tấn công mới, giúp tăng cường bảo mật cho ứng dụng web.

CHƯƠNG 3. MÔ HÌNH HYDRA-WAFS ỨNG DỤNG MULTIMODAL

Ở chương này, nhóm trình bày từ tổng quan đến chi tiết mô hình tường lửa ứng dụng web (HYDRA-WAFS) cũng như mô hình học sâu được sử dụng trong hệ thống HYDRA-WAFS.

3.1. Mô hình tường lửa ứng dụng web (HYDRA-WAFS)

3.1.1. Mô hình đề xuất



Hình 3.1: *Tổng quan mô hình HYDRA-WAFS*

Cụ thể các thành phần và cơ chế hoạt động của mô hình được đề xuất như sau:

- Khi người dùng gửi các yêu cầu HTTP/HTTPS truy cập đến Web server, các yêu cầu đó phải đi qua WAF trước khi đến Web server xử lý và trả lời phản hồi cho người dùng. WAF như một lớp trung gian giữa người dùng và máy chủ web để bảo vệ ứng dụng web khỏi các cuộc tấn công và mối đe

dọa an ninh.

- Đầu tiên các yêu cầu HTTP/HTTPS (bao gồm cả độc hại và lành tính) sẽ được ghi nhận lại tại mô-đun học đặc trưng (Feature Learning) để tiến hành xử lý cũng như trích xuất các đặc trưng quan trọng. Sau đó những đặc trưng này sẽ được dùng làm đầu vào của mô-đun mô hình học sâu (Deep Learning Model). Cuối cùng, mô-đun quyết định (Decision) dựa vào kết quả phân loại từ mô-đun mô hình học sâu để đưa ra quyết định ngăn chặn (nếu yêu cầu được dự đoán là độc hại) hay cho yêu cầu đi qua đến Web server (nếu yêu cầu được dự đoán là lành tính).

Hệ thống Phát hiện Tấn công Web ứng dụng multimodal Deep Learning (HYDRA-WADS) gồm ba mô-đun:

- Mô-đun học đặc trưng (Feature Learning): được áp dụng để phân tích các đặc trưng của các yêu cầu HTTP/HTTPS và biến đổi chúng thành các vectơ có đặc trưng bất thường.
- Mô-đun mô hình học sâu (Deep Learning Model): bao gồm ba mô hình deep learning độc lập và được kết hợp lại tạo ra vector phân loại cuối cùng.
- Mô-đun quyết định (Decision): được sử dụng để kết hợp các kết quả từ mô hình học sâu để thu được kết quả cuối cùng cho việc phát hiện.

3.1.2. Mô-đun học đặc trưng (Feature Learning)

Các yêu cầu HTTP/HTTPS mà người dùng gửi đến Web server sẽ được bắt lại tại đây để trích xuất ra các thuộc tính cần thiết như "URI", "Method", "User-Agent", "Accept", "Request Body",... Các trường thuộc tính này sẽ chia thành 3 nhóm (mỗi nhóm sẽ từ 2 đến 3 thuộc tính). Mỗi nhóm sẽ được xử lý theo 1 trong 2 phương pháp xử lý ngôn ngữ tự nhiên Natural Language Processing (NLP) sau:

1. **Phương pháp 1: Word Tokenizer**

   Trong phương pháp này, các chuỗi được tạo từ các yêu cầu HTTP ở dạng text, sử dụng Keras word tokenizer và Keras word embedding functions. Trong giai đoạn token hóa, số nguyên đưa ra sau khi xác định các số hạng thường gặp nhất bằng cách tạo tất cả các vectơ thuật ngữ có thể có trong các yêu cầu web.

   Trong quá trình mã hóa, một phân tích hiệu quả được thực hiện bằng cách sử dụng một số thuật ngữ khác nhau để xác định có bao nhiêu thuật ngữ sẽ đủ để đảm bảo sự thành công của việc phân loại. Post-padding được xử lý bằng cách thêm các số không vào đuôi của vectơ để đảm bảo cùng kích thước. Do đó, tất cả các mẫu tấn công có thể có trong các yêu cầu HTTP có thể được xử lý bất kể độ dài thuật ngữ. Word tokens được tạo theo tần suất và phân tích liên kết của các biến thể phụ của các nhóm ký tự trong các yêu cầu web trong tập dữ liệu. Những từ thường gặp nhất trong tập dữ liệu được xác định bằng cách xem xét các liên kết của các ký tự và tần số của chúng trong toàn bộ tập dữ liệu.

2. **Phương pháp 2: Quy về dạng chung và map index**

   Đầu tiên, nhóm đã phát triển một từ điển như trong Hình 3.2.

   Từ điển bao gồm các từ khóa trong HTML, JavaScript, SQL, Linux, Window, v.v. và một bảng để chuyển đổi. Sau đó, nhóm chuẩn hóa các thuộc tính trong yêu cầu HTTP bằng cách giữ lại các từ trong từ điển và thay thế các thuộc tính còn lại trong yêu cầu HTTP theo bảng chuyển đổi. Quy tắc quy đổi về dạng chuẩn: 1) Ưu tiên giữa lại những từ trong dictionary. 2) Nếu từ nào không nằm trong dictionary sẽ dùng regex để kiểm tra và thay bằng ‘numbers’, ‘unistring’, ‘hexstring’, ‘purestring’, hoặc ‘mixstring’. Ưu điểm nổi bật của phương pháp này là có thể phát hiện các đặc trưng bất thường dựa vào các từ lẫn các ký tự đặc biệt. Phương pháp này này cũng sẽ bao quát được tất cả các trường hợp trong thực tế khi quy đổi những từ

không có trong từ điển về dạng chung. Cụ thể, từ điển được thể hiện trong Bảng 3.1 và bảng biến đổi được trình bày trong Bảng 3.2. Ví dụ: các thuộc tính trong yêu cầu HTTP được mã hóa bằng cách xử lý dữ liệu trông giống như:

/ bbiq / usershome / userinformation . php ? union select 1 , 2 , (select loadfile (' / var / www / html / sql-connections / db-creds . inc ')) - - +

Nó sẽ được chuyển đổi thành một biểu thức tiêu chuẩn như sau:

/ bbiq / usershome / userinformation . php ? union select Numbers , Numbers , (select loadfile (' / Purestring / Purestring / html / PureString / PureString . inc ')) - - +

Bảng 3.1: *Ví dụ về từ khóa được định nghĩa*

Description	**Keywords**
HTML	script a abbr b big body br img src href ...
Javascript	await const fetch alert async case catch char ...
SQL	Union select and or if order by limit concat create table from...
Punctuation	! @ ; ' " < > / ? * ( )

Bảng 3.2: *Những từ không có trong từ điển được chuyển đổi về dạng chung*

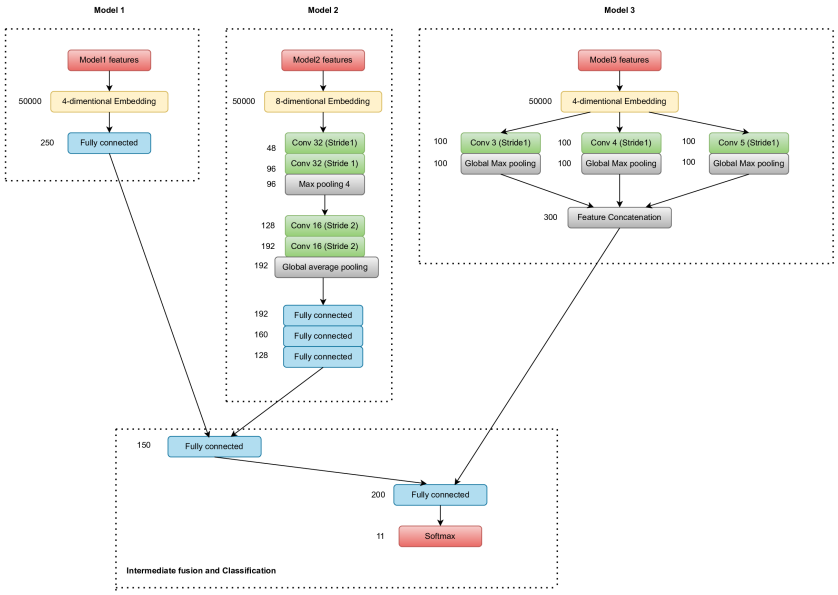
Transformation	**Description**
Numbers	Represent numbers
UniString	Represent Unicode strings
HexString	Represent Hex strings
PureString	Represent strings which are made up of a-z and '-'
MixString	Represent of other characters beside a-z and '-'

```
purestring
/
=
pathstring
.
numbers
&
)
?
(
mixstring
'
+
,
\
"
|
jsp
>
<
;
select
hexstring
$
unistring
:
@
null
cp
```

Hình 3.2: *File* ***dictionary.txt*** *được định nghĩa*

3.1.3. Mô-đun mô hình học sâu (Multimodal deep neural network)

Trong phần này nhóm sẽ trình bày tổng quan và chi tiết về mô hình học sâu multimodal deep neutral network (HYDRA)



Hình 3.3: *Tổng quan kiến trúc mô hình multimodal deep neural network*

3.1.3.1. Tổng quan mô hình

Nhóm có tham khảo các mô hình ứng dụng multimodal trong lĩnh vực phát hiện các loại tấn công khác như Malware, Network... và cuối cùng chọn ra được mô hình phù hợp [3] để triển khai và tận dụng lại làm mô hình học sâu trong hệ thống HYDRA-WAFS.

Tổng quan có thể thấy mô hình được tạo thành từ 3 model riêng biệt. Mỗi model này sẽ lần lượt nhận một nhóm thuộc tính từ Mô-đun học đặc trưng làm input đầu vào. Cụ thể, Model 1 sẽ nhận input là dữ liệu sau khi tiền xử lý như "Accept", "User-Agent"; Model 2 sẽ nhận input là dữ liệu sau khi tiền xử lý

như "Referer", "Cookie"; Model 3 sẽ nhận input là dữ liệu sau khi tiền xử lý như "Host", "URI". Dữ liệu sau khi đưa vào 3 model sẽ đi qua các lớp như Embedding, Conv1D, GlobalMaxPooling, Fully Connected,... Cuối cùng, nhóm dùng hàm kích hoạt Softmax kết hợp các output từ các model lại với nhau thành một vector duy nhất phục vụ cho quá trình phân loại.

3.1.3.2. Chi tiết mô hình

Hình 3.3 minh họa kiến trúc của HYDRA (multimodal deep neural network) để phân loại các loại tấn công Web. Mô hình gồm 4 thành phần chính gồm Model 1, Model 2, Model 3 và Feature fusion and classification. Ba thành phần đầu tiên không được kết nối với nhau. Mỗi thành phần lần lượt nhận một nhóm thuộc tính từ Mô-đun học đặc trưng làm input đầu vào. Thành phần cuối cùng chịu trách nhiệm hợp nhất các thuộc tính được học bởi mỗi thành phần thành một biểu diễn chung và tạo ra các dự đoán phân loại. Cụ thể các thành phần như sau:

Model 1

Về đầu vào, các thuộc tính đưa vào Model 1 bao gồm "Accept", "User-Agent",... đã được tiền xử lý về dạng mảng số.

Về cấu trúc mô hình, Model 1 gồm các lớp Input Layer là Embedding Layer để biểu diễn số lượng số nguyên đầu cũng là số lượng từ phân biệt của mô hình Word2vec vào thành các vector số có 4 chiều, sau đó đưa vào Hidden Layer sử dụng Flatten để xử lý chiều ma trận và output để xử lý đưa vào kích thước ma trận mong muốn để xử lý nhằm liên kết các feature vector với các model kế tiếp trước khi thực hiện connected.

Model 2

Về đầu vào, các thuộc tính đưa vào Model 2 bao gồm "Referer" và "Cookie",... đã được tiền xử lý về dạng mảng số.

Về mô hình, lớp hidden trong Model 2 bao gồm các lớp Conv1D, MaxPool-

ing1D và Dense. Lớp Conv1D áp dụng các bộ lọc tích chập để tìm ra các đặc trưng cục bộ và mối quan hệ giữa các thành phần trong mảng input đầu vào. Lớp MaxPooling1D được sử dụng để giảm kích thước của đầu ra và tạo ra các biểu diễn tóm tắt của dữ liệu. Cuối cùng, các lớp Dense trong Hidden Layer tạo ra các biểu diễn đặc trưng phức tạp hơn từ dữ liệu đầu vào bằng cách kết hợp các thông tin đã học từ các lớp trước đó. Việc sử dụng nhiều lớp Dense cho phép mô hình học các mức độ biểu diễn trừu tượng và tăng cường khả năng mô hình hóa dữ liệu do đầu vào Model 2 có phần phức tạp và đa dạng hơn. Các lớp Dense có thể học các đặc trưng cấp cao và tạo ra các biểu diễn đặc trưng phức tạp hơn từ dữ liệu đầu vào. Điều này giúp mô hình có khả năng biểu diễn và học được các mối quan hệ phức tạp giữa các đặc trưng và đầu ra mong muốn.

Model 3.

Về đầu vào, các thuộc tính đưa vào Model 3 bao gồm "Host", "URI",... đã được tiền xử lý về dạng mảng số.

Về mô hình, mô hình sử dụng các lớp Conv1D với các kích thước bộ lọc khác nhau để trích xuất đặc trưng từ mảng input đầu vào. Tiếp theo, lớp GlobalMax-Pooling1D được sử dụng để lấy giá trị lớn nhất trên mỗi chiều từ mảng input đầu vào đã được xử lý. Cuối cùng, đầu ra từ lớp GlobalMaxPooling1D được sử dụng trong các bước tiếp theo của mô hình hoặc làm đầu vào cho các lớp mạng nơ-ron khác. Mô hình Model 3 giúp trích xuất thông tin quan trọng từ văn bản và tạo ra một biểu diễn tổng quan của dữ liệu đầu vào.

Feature fusion and classification. Các biểu diễn ("Accept", "User-Agent"), ("Referer", "Cookie") và ("Host", "URI") đã học của các loại tấn công Web được hợp nhất lặp đi lặp lại trên nhiều lớp hợp nhất trong quá trình huấn luyện và kết hợp thành một đại diện đa phương thức. Quá trình này được gọi là phản ứng tổng hợp trung gian (Intermediate fusion). Đầu tiên, vectơ output của Model 1 (Ký hiệu A) và vectơ output của Model 2 (Ký hiệu B) được hợp nhất thành

vectơ C có kích thước 150. Sau đó, vectơ C và vectơ output của Model 3 (Ký hiệu M) được kết hợp trong một vectơ 1 chiều có kích thước 200, được gọi là P. Biểu diễn đa phương thức chung P này sau đó được sử dụng để phân loại các loại tấn công Web thành họ tương ứng, như sau:

$$\text{p} = \text{softmax}(W_c P + b_c)$$

Trong đó p là một vectơ có kích thước C (C = <Số lượng loại tấn công> + 1), và W_c và b_c là trọng số và độ thiên vị của lớp. Hàm softmax xuất ra xác suất của một HTTP/HTTPS request thuộc về là "Normal" hay thuộc về bất kỳ họ tấn công Web nào trong tập huấn luyện.

3.1.4. Mô-đun quyết định (Decision)

Mô-đun này có nhiệm phụ phân loại một request HTTP/HTTPS là "Normal" hay thuộc vào các kỹ thuật tấn công dựa trên nhãn có xác xuất cao nhất trong các họ tấn công web khác nhau của vector p. Dựa vào kết quả này, nếu request được dự đoán là "Normal", HYDRA-WAFS sẽ cho phép request đó đi qua để truy cập đến Webserver. Ngược lại, nếu request được dự đoán thuộc một trong các loại tấn công trong tập huấn luyện, HYDRA-WAFS sẽ đưa ra cảnh báo loại tấn công đó cho người dùng thông qua nhật ký (logs) của HYDRA-WAFS và đưa ra ngăn chặn.

CHƯƠNG 4. THÍ NGHIỆM VÀ ĐÁNH GIÁ

Ở chương này nhóm tiến tạo môi trường, cài đặt và đưa ra các tiêu chí đánh giá về mức độ hiệu quả của mô hình. Sau đó, triển khai mô hình trong môi trường ảo hóa đánh giá hiệu quả.

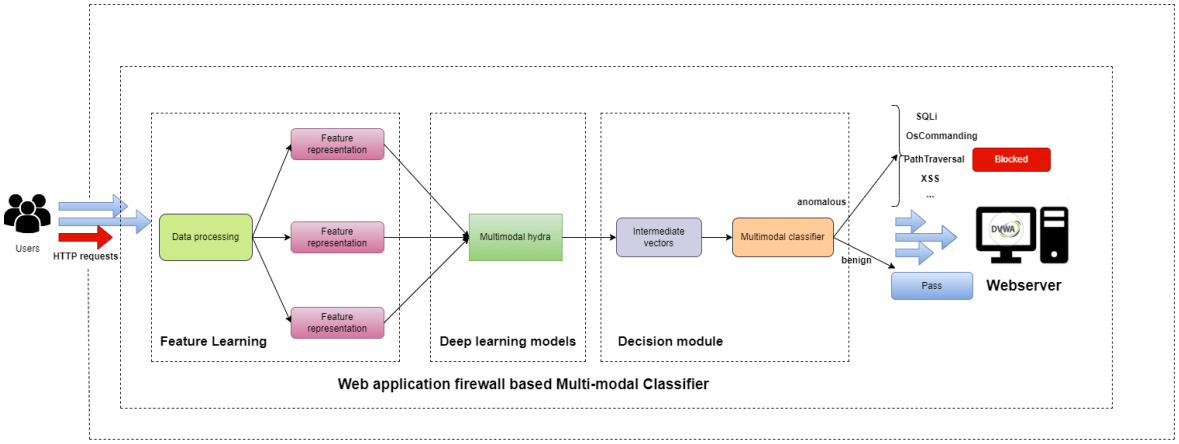
4.1. Thiết lập thí nghiệm

1. **Giai đoạn 1:** Đào tạo mô hình như đã đề cập ở phần 3.1.3
   Ở giai đoạn này, nhóm sẽ tiến hành xử lý dữ liệu từ các tập dataset và đưa vào đào tạo mô hình học sâu. Nhóm sẽ sử dụng các thư viện có sẵn như Scikit-learn, Keras, Tensorflow,... trên nền tảng Google Colab 12GB RAM, sử dụng Python 3.10.

2. **Giai đoạn 2:** Triển khai mô hình học sâu sau khi đào tạo ở Giai đoạn 1 vào mã nguồn của ModSecurity.
   ModSecurity là module tường lửa thông dụng thường được triển khai nhằm bảo vệ và kiểm soát các truy cập ra vào Web server.

   Hệ thống HYDRA-WAFS được triển khai trên môi trường ảo sử dụng VMWare Workstation Pro 17 gồm 3 máy: 1) Kalilinux 2022.3 (2G RAM, 2 processors, 2 core per processor) làm máy attacker 2) Ubuntu 20.04 (3G RAM, 2 processors, 1 core per processor) làm máy WAF 3) Ubuntu 18.04 (2G RAM, 2 processors, 1 core per processor) làm máy chủ Web.

   1. Ngôn ngữ lập trình: C/C++.
   2. Nền tảng: C_API Tensorflow.



Hình 4.1: *Mô hình triển khai*

4.2. Tập dữ liệu

4.2.1. CSIC-2010

Bộ dữ liệu này được tạo ra tại Consejo Superior de Investigaciones Científicas (CSIC) [22]. Bắt nguồn từ các yêu cầu web mô phỏng đến một ứng dụng web thương mại điện tử, bộ dữ liệu này bao gồm 36.000 yêu cầu bình thường và hơn 25.000 yêu cầu bất thường.

4.2.2. ECML/PKDD 2007

Bộ dữ liệu này đã được công bố tại hội nghị *ECML/PKDD* năm 2007 [8], như một phần của Thử thách khám phá Phân tích lưu lượng truy cập web *ECML/PKDD 2007*, bộ dữ liệu này bao gồm 35.006 bản ghi bình thường và 15.110 bản ghi được gắn nhãn là tấn công (bao gồm 7 loại tấn công LdapInjection, OsCommanding, PathTransversal, SSI, SqlInjection, XPathInjection, XSS). Chi tiết phân phối các mẫu dữ liệu được biểu diễn qua Bảng 4.1.

Bảng 4.1: *Phân phối các mẫu trong tập dữ liệu ECML/PKDD 2007*

No.	Category	Number of Samples
1	LdapInjection	2279
2	OsCommanding	2302
3	PathTransversal	2295
4	SSI	1856
5	SqlInjection	2274
6	XPathInjection	2279
7	XSS	1825
8	Normal	35,006
9	Total	50,116

4.2.3. SR-BH 2020 multilabel dataset [18]

Tập dữ liệu bao gồm các yêu cầu web được thu thập trong 12 ngày của tháng 7 năm 2020 bởi một máy chủ web (Wordpress) được cài đặt trên máy ảo và tiếp xúc với Internet. Trên máy chủ này, Modsecurity phiên bản 2.9.2 cho Apache, với Core Rule Set (CRS) phiên bản 3.3.0 đã được cài đặt ở chế độ D̈etection Only¨; để tất cả các yêu cầu (hợp pháp và độc hại) được ghi lại trong nhật ký do ModSecurity tạo ra, nhưng không bị chặn. Hàng ngày, các bản ghi do ModSecurity tạo ra đã được thu thập và máy ảo được khôi phục về trạng thái sạch.

Khi thời gian tiếp xúc với máy chủ web kết thúc, các bản ghi được thu thập được xử lý thủ công và bán tự động bởi một trong các tác giả để xem xét việc gắn thẻ yêu cầu web do Modsecurity thực hiện, sửa chữa khi cần thiết việc gán bình thường/tấn công cho yêu cầu web tương ứng và đảm bảo gán phân loại CAPEC thích hợp.

Kết quả cuối cùng là một bộ dữ liệu đa nhãn đặc biệt nhằm phát hiện tấn công web và bao gồm 907.814 yêu cầu, trong đó 525.195 là yêu cầu bình thường và 382.619 là yêu cầu bất thường, trong đó mỗi bản ghi có 24 tính năng khác nhau và một bộ 13 nhãn.

Chi tiết phân phối các mẫu dữ liệu được biểu diễn qua Bảng 4.2.

Bảng 4.2: *Phân phối các mẫu trong tập dữ liệu SR-BH 2020*

No.	Category	Number of Samples
1	000 - Normal	525,195
2	272 - Protocol Manipulation	9153
3	242 - Code Injection	15,827
4	88 - OS Command Injection	7482
5	126 - Path Traversa	20,992
6	66 - SQL Injection	250,311
7	16 - Dictionary-based Password Attack	1847
8	310 - Scanning for vulnerable software	2718
9	153 - Input Data Manipulation	2272
10	274 - HTTP Verb Tampering	5437
11	194 - Fake the source of data	56,145
12	34 - HTTP Response Splitting	19,738
13	33 - HTTP Request Smuggling	1059
14	Total	918,176

4.3. Chỉ số đánh giá

Nhóm đã sử dụng các chỉ số hiệu suất tiêu chuẩn để đánh giá hiệu quả phân loại của mô hình như Confusion Matrix (CM), Accuracy (ACC), Precision (PRE), Recall Score (RECALL), F1-Score (F1), Đường cong ROC và Đồ thị mất mát (Log Loss) cho mô hình học sâu (HYDRA) trên cả ba tập dữ liệu.

Ma trận nhầm lẫn là một kỹ thuật để tóm tắt hiệu suất của thuật toán phân loại báo cáo số lượng dương tính thật (TP), âm tính thật (TN), dương tính giả (FP) và âm tính giả (FN).

Độ chính xác (ACC) được tính bằng tổng số hai dự đoán đúng chia cho tổng số tập dữ liệu.

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Độ chính xác dương (PRE) là khả năng của một mô hình để tìm tất cả các trường hợp dương tính dự đoán được phân loại chính xác.

$$\text{PRE} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Độ phủ (RECALL) là tỷ lệ các trường hợp dương tính được xác định chính xác.

$$\text{RECALL} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1-Score (F1) là thước đo của lớp dương, kết hợp độ chính xác và khả năng nhớ lại của một bộ phân loại thành một số liệu duy nhất bằng cách lấy trung bình điều hòa của chúng.

$$\text{F1-Score} = \frac{2*\text{Precision}*\text{Recall}}{\text{Precision} + \text{Recall}}$$

Đường cong ROC là một biểu đồ sử dụng trong đánh giá hiệu suất của một mô hình phân loại. Nó biểu thị mối quan hệ giữa tỷ lệ True Positive (TPR) và tỷ lệ False Positive (FPR) trên toàn bộ các ngưỡng quyết định có thể áp dụng cho mô hình. Đường cong ROC cũng được sử dụng để tính toán AUC (Area Under the Curve), diện tích nằm dưới đường cong ROC. Giá trị AUC càng gần 1, tức diện tích càng lớn, thì mô hình càng tốt trong việc phân loại các lớp.

Log loss là một hàm mất phân loại được sử dụng cho mô hình. Nó cho biết xác suất dự đoán gần với giá trị thực tế tương ứng như thế nào.

4.4. Kết quả thí nghiệm

Ở mục này, nhóm sẽ trình bày các kết quả thực nghiệm và đưa ra đánh giá.

Nhóm tập trung trả lời hai câu hỏi sau:

Câu hỏi 1: Các chỉ số đánh giá khác nhau như thế nào với các tập dữ liệu khác nhau trong bài toán phân loại nhị nhân (binary classification) và bài toán phân loại đa lớp (multi classification)?

Câu hỏi 2: Chọn ra mô hình phân loại đa lớp có kết quả tốt để triển khai vào mã nguồn của Modsecurity?

4.4.1. Câu hỏi 1: Các chỉ số đánh giá khác nhau như thế nào với các tập dữ liệu khác nhau trong bài toán phân loại nhị nhân (binary classification) và bài toán phân loại đa lớp (multi classification)?

Trong tất cả các kết quả thí nghiệm bên dưới, các tập dataset đều được chia theo tỷ lệ 70% dùng để train và 30% còn lại dùng để đánh giá hiệu quả mô hình phân loại.

Bảng 4.3 biểu thị kết quả phân loại nhị phân trên cả 3 tập dataset với 2 phương pháp xử lý dữ liệu được sử dụng. Có thể thấy rằng tập dataset SR-BH 2020 cho độ chính xác ACC cũng như các chỉ số khác cao nhất khoảng 99% với cả 2 cách tiếp cận. Khi sử dụng phương pháp NLP1, kết quả trên tập CSIC-2010 cho kết quả khá ấn tượng với ACC và F1 lần lượt là 97,45% và 97,44% tuy nhiên lại khá thấp với phương pháp NLP2. Các chỉ số ACC, PRE, RECALL, F1 đều khá tương đồng ở mức 95% trên tập *ECML/PKDD 2007*. Kết quả này tương đối cạnh tranh so với độ chính xác trên tập CSIC-2010 và ECML/PKDD 2007 lần lượt là 96,95% và 99,06% trong [7].

Dataset	ACC (NLP1, NLP2)	PRE (NLP1, NLP2)	RECALL (NLP1, NLP2)	F1 (NLP1, NLP2)	AUC (NLP1, NLP2)
CSIC 2010	0.9745,0.9188	0.9748,0.9251	0.9745,0.9188	0.9744,0.9174	0.9911,0.9718
ECML/PKDD 2007	0.9497,0.9457	0.9530,0.9477	0.9497,0.9457	0.9503,0.9443	**0.9920**,0.9573
SR-BH 2020	**0.98**,0.9761	**0.9882**,0.9763	**0.9881**,0.9761	**0.9881**,0.9762	0.9887,0.9759

Bảng 4.3: *Chỉ số đánh giá trung bình trên 3 tập dataset với 2 phương pháp NLP được đề xuất (Phân loại nhị phân)*

Dataset	ACC (NLP1, NLP2)	PRE (NLP1, NLP2)	RECALL (NLP1, NLP2)	F1 (NLP1, NLP2)	AUC (NLP1, NLP2)
ECML/PKDD 2007	0.8911,0.9264	0.9169,0.9292	0.8911,0.9264	0.8982,0.9232	0.9727,0.9533
SR-BH 2020	**0.9850**,0.9708	**0.9847**,0.9713	**0.9850**,0.9708	**0.9845**,0.9681	**0.9985**,0.9968

Bảng 4.4: *Chỉ số đánh giá trung bình trên 3 tập dataset với 2 phương pháp NLP được đề xuất (Phân loại đa lớp)*

Dataset	Thời gian training/step (s) (NLP1 - NLP2)
CSIC-2010	**1,054** - 1,562
ECML/PKDD 2007	1,060 - **1,103**
SR-BH 2020	1,194 - 1,153

Bảng 4.5: *Thời gian training trên 3 tập dataset với 2 phương pháp NLP được đề xuất (Phân loại nhị phân)*

Dataset	Thời gian training/step (s) (NLP1 - NLP2)
ECML/PKDD 2007	**1,153** - **1,158**
SR-BH 2020	1,194 - 1,160

Bảng 4.6: *Thời gian training trên 2 tập dataset với 2 phương pháp NLP được đề xuất (Phân loại đa lớp)*

Bảng 4.5 cho thấy thời gian trung bình phục vụ cho quá trình training trên 3 tập dataset trong phân loại nhị phân. Có thể thấy, mô hình yêu cầu thời gian training thấp nhất trên tập CSIC-2010 là 1,054s với phương pháp NLP1. Trong khi đó, trên tập CSIC-2010 cũng yêu cầu thời gian training lâu nhất là 1,562s khi sử dụng phương pháp NLP2. Tập dataset ECML/PKDD 2007 và SR-BH 2020 không có sự thay đổi nhiều giữa 2 phương pháp ở mức 1,1-1,2s.

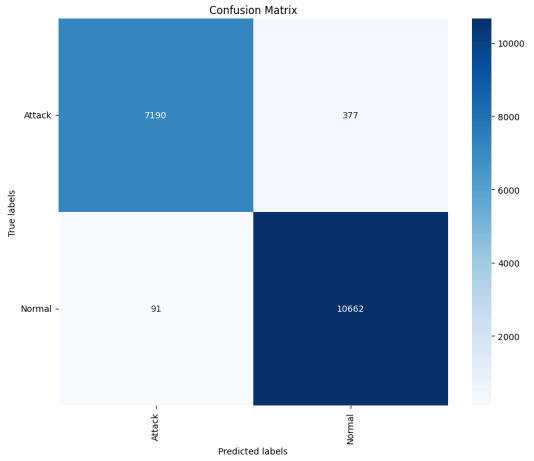
Bảng 4.6 cho thấy thời gian trung bình phục vụ cho quá trình training trên 2 tập dataset trong phân loại nhị phân. Có thể thấy, mô hình yêu cầu thời gian training trên tập ECML/PKDD 2007 thấp hơn một chút so với SR-BH 2020.

Bảng 4.4 biểu thị kết quả phân loại đa lớp trên 2 tập dataset (ECML/PKDD 2007 và SR-BH 2020) với 2 phương pháp xử lý dữ liệu được sử dụng. Có thể thấy rằng tập dataset SR-BH 2020 cho độ chính xác ACC cũng như các chỉ số khác cao nhất khoảng 98,5% với cả 2 cách tiếp cận. Kết quả này cao hơn so với [18] khi độ chính xác chỉ ở mức 88%. Ở phía ngược lại, kết quả phân loại tập ECML/PKDD 2007 tuy không cho kết quả quá cao nhưng các chỉ số đều khá tương đồng dao động từ 90-93% và kết quả này hơi nhỉnh hơn về phía phương pháp NLP2. Kết quả này khá tương đồng so với 93% trong [7].

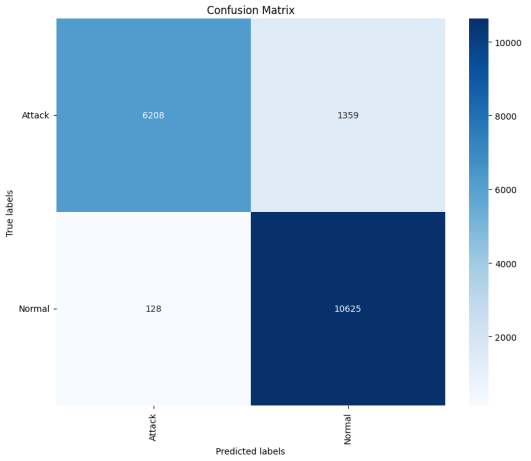
Hình 4.2, Hình 4.3 và Hình 4.5 là ma trận nhầm lẫn trên cả 3 tập dataset cho bài toán phân loại nhị phân (Attack hay Normal). Nhìn chung số lượng dự đoán đúng (TP và TN) đều ở mức cao và tỷ lệ dự đoán sai ở mức chấp nhận được. Tuy nhiên, số lượng nhầm lẫn ở mức tương đối cao (FN = 1359) có thể thấy ở Hình 4.2b.

Hình 4.4 là ma trận nhầm lẫn trên tập ECML/PKDD 2007 trong bài toán phân loại đa lớp. Có thể thấy phương pháp NLP1 như trong hình 4.4a cho kết quả nhầm lẫn tương đối nhiều và đồng đều ở tất cả các nhãn. Trong khi đó, Hình 4.4b phân loại khá tốt ở các nhãn ngoại trừ nhãn "Normal".

Hình 4.6 là ma trận nhầm lẫn trên tập SR-BH 2020 trong bài toán phân loại đa lớp. Có thể thấy phương pháp NLP1 như trong hình 4.6a và phương pháp NLP2 như trong Hình 4.6b cho kết quả phân loại khá tốt và tương đồng giữa các nhãn.



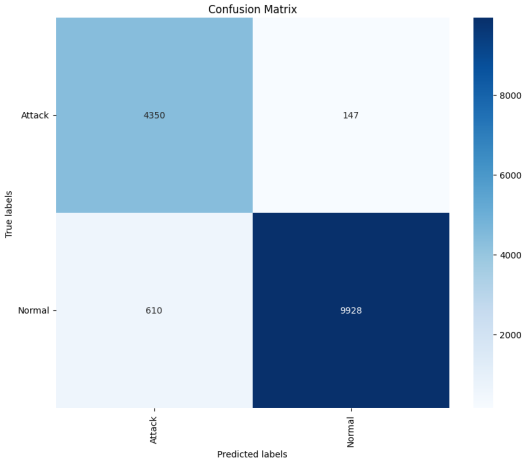
(a) *Confusion matrix trên tập CSIC 2010 (NLP1)*



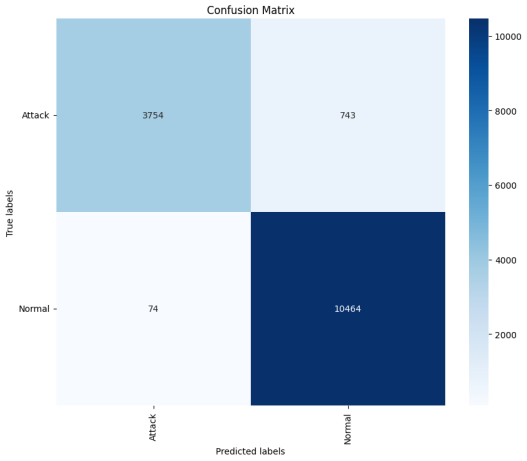
(b) *Confusion matrix trên tập CSIC 2010 (NLP2)*

Hình 4.2: *Confusion matrices trên tập CSIC 2010 (Phân loại nhị phân)*

Nhìn vào các đường cong ROC trong các hình Hình 4.7, Hình 4.8 và Hình 4.10 cho thấy kết quả phân loại nhị phân khi sử dụng phương pháp NLP1 cho hiệu quả phân loại tốt hơn phương pháp NLP2 với diện tích dưới đường cong

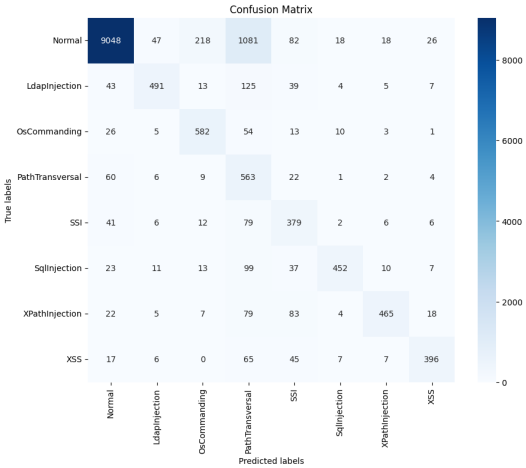


(a) *Confusion matrix trên tập ECML/PKDD 2007 (NLP1)*



(b) *Confusion matrix trên tập ECML/PKDD 2007 (NLP2)*

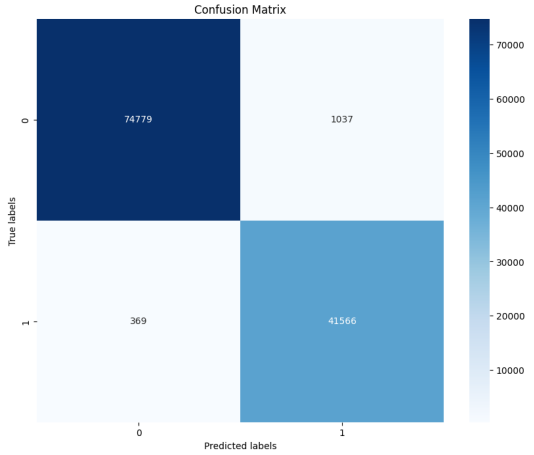
Hình 4.3: *Confusion matrices trên tập ECML/PKDD 2007 (Phân loại nhị phân)*



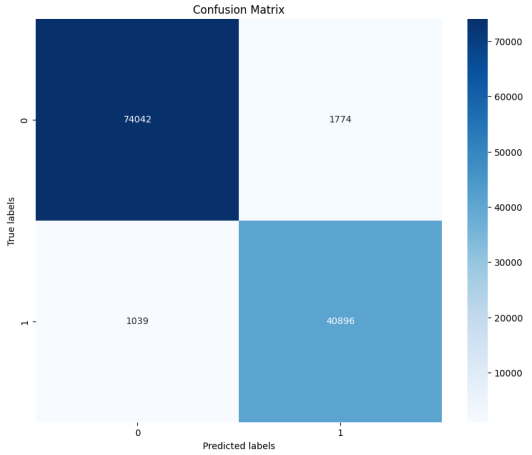
(a) *Confusion matrix trên tập ECML/PKDD 2007 (NLP1)*

(b) *Confusion matrix trên tập ECML/PKDD 2007 (NLP2)*

Hình 4.4: *Confusion matrices trên tập ECML/PKDD 2007 (Phân loại đa lớp)*



(a) *Confusion matrix trên tập SR-BH 2020 (NLP1)*



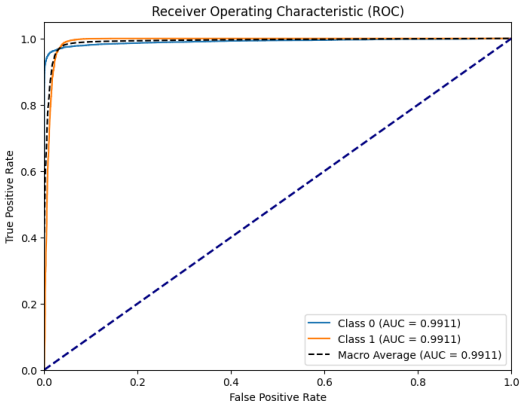
(b) *Confusion matrix trên tập SR-BH 2020 (NLP2)*

Hình 4.5: *Confusion matrices trên tập SR-BH 2020 (Phân loại nhị phân)*

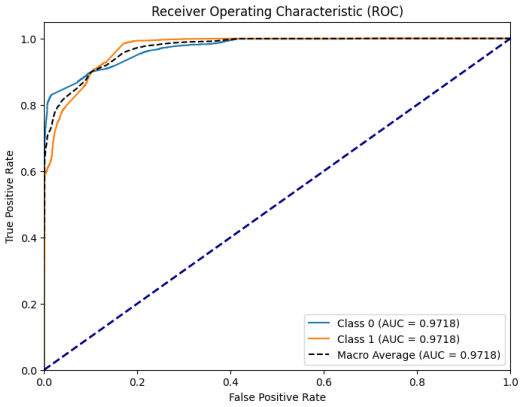
(a) *Confusion matrix trên tập SR-BH 2020 (NLP1)*

(b) *Confusion matrix trên tập SR-BH 2020 (NLP2)*

Hình 4.6: *Confusion matrices trên tập SR-BH 2020 (Phân loại đa lớp)*

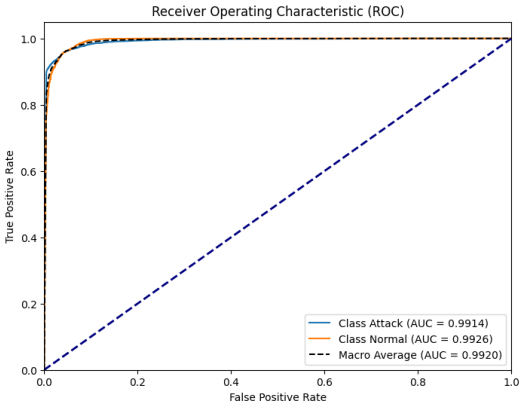


(a) *ROC Curve trên tập CSIC 2010 (NLP1)*

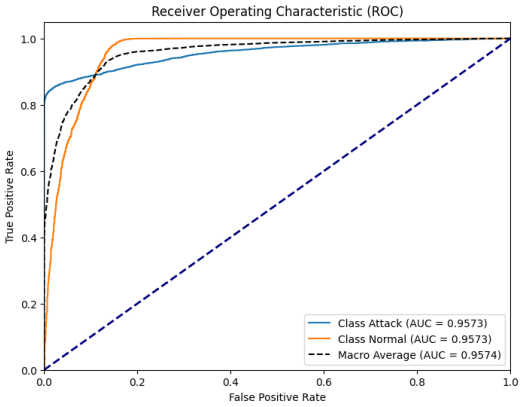


(b) *ROC Curve trên tập CSIC 2010 (NLP2)*

Hình 4.7: *ROC Curves trên tập CSIC 2010*

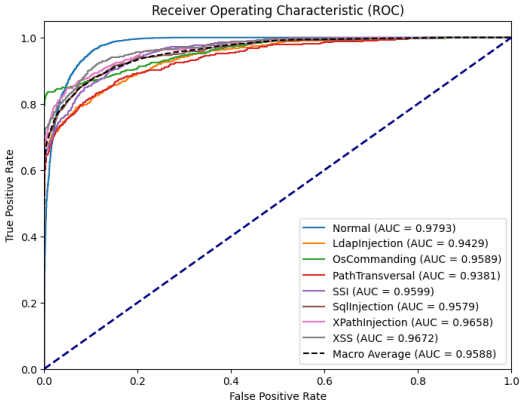


(a) *ROC Curve trên tập ECML/PKDD 2007 (NLP1)*

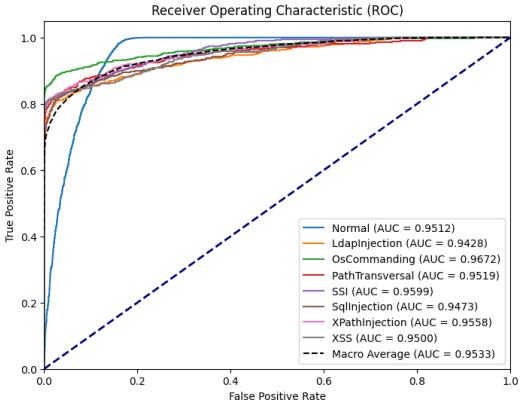


(b) *ROC Curve trên tập ECML/PKDD 2007 (NLP2)*

Hình 4.8: *ROC Curves trên tập ECML/PKDD 2007 (Phân loại nhị phân)*

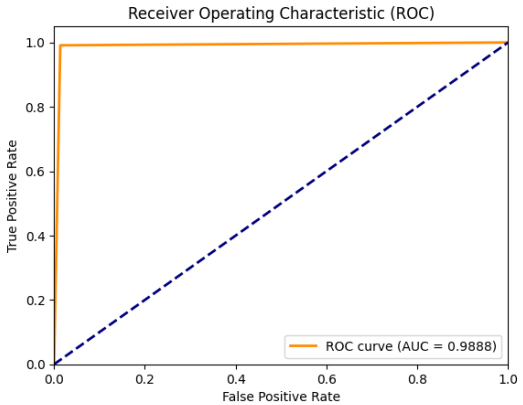


(a) *ROC Curve trên tập ECML/PKDD 2007 (NLP1)*

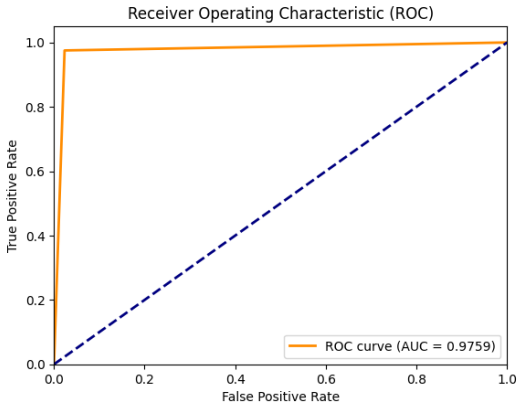


(b) *ROC Curve trên tập ECML/PKDD 2007 (NLP2)*

Hình 4.9: *ROC Curves trên tập ECML/PKDD 2007 (Phân loại đa lớp)*

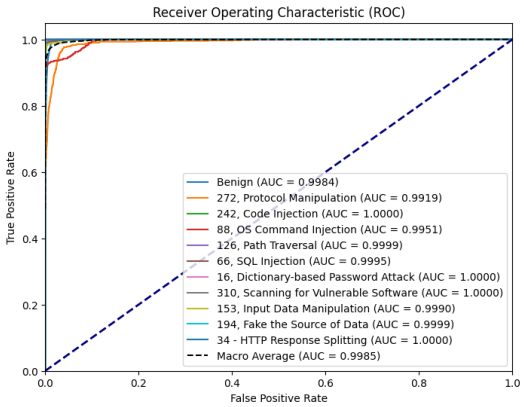


(a) *ROC Curve trên tập SR-BH 2020 (NLP1)*

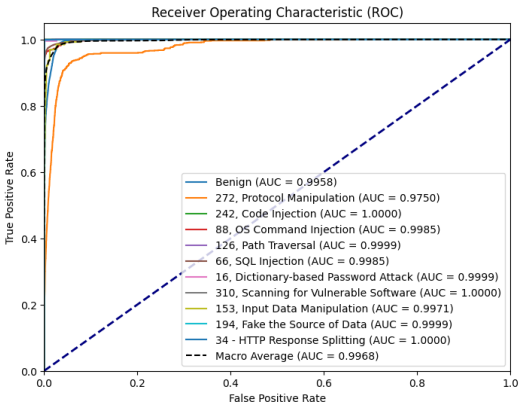


(b) *ROC Curve trên tập SR-BH 2020 (NLP2)*

Hình 4.10: *ROC Curves trên tập SR-BH 2020 (Phân loại nhị phân)*



(a) *ROC Curve trên tập SR-BH 2020 (NLP1)*

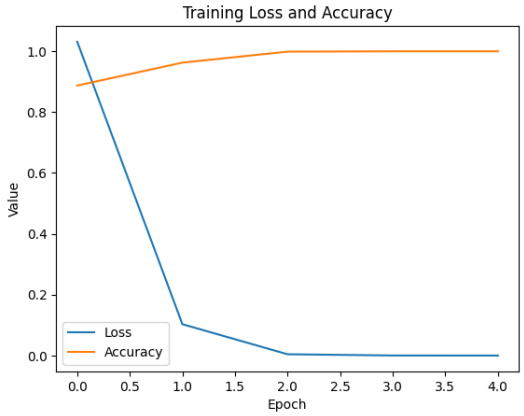


(b) *ROC Curve trên tập SR-BH 2020 (NLP2)*

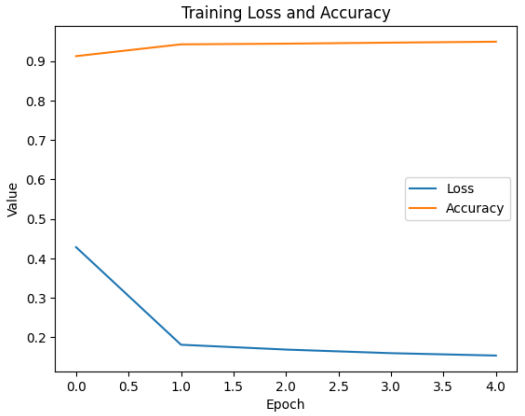
Hình 4.11: *ROC Curves trên tập SR-BH 2020 (Phân loại đa lớp)*

(Macro-Average AUC) lần lượt là 0.9911, 0.9920 và 0.9888 trong khi phương pháp NLP2 chỉ ở mức 0.9718, 0.9574, 0.9759.

Các đường cong ROC trong các hình Hình 4.9 và Hình 4.11 cho thấy kết quả phân loại đa lớp khá ấn tượng. Cụ thể, Hình 4.11a cho kết quả AUC đạt được 0.9793, 0.9672, 0.9658 ở các lớp Normal, XSS, XPathInjection và gần như tương đồng ở mức cao với các lớp còn lại. Hình 4.11 cho kết quả cao nhất ở cả 2 phương pháp với Macro-Average AUC lần lượt là 0.9985 và 0.9968.



(a) *Log loss vs acc trên tập ECML/PKDD 2007 (NLP1)*



(b) *Log loss vs acc trên tập ECML/PKDD 2007 (NLP2)*

Hình 4.12: *Log loss vs acc trên tập ECML/PKDD 2007.*

Đồ thị mất mát và accuracy trong quá trình huấn huyện với epochs=5 được thể hiện qua Hình 4.12.

Tác giả	Mô hình	CSIC(Binary)	ECML/PKDD(Binary)	ECML/PKDD(Multi)
Nguyen and Franke [14]	Naive Bayes	72,78%	85,12%	N/A
Nguyen and Franke [14]	Bayes Network	82,79%	86,95%	N/A
Nguyen and Franke [14]	RBF Network	82,79%	86,95%	N/A
Nguyen and Franke [14]	Hedge Boosting	82,10%	82,36%	N/A
Nguyen et al. [15]	C4.5	94,49%	96,37%	N/A
Nguyen et al. [15]	CART	94,12%	96,11%	N/A
Nguyen et al. [15]	Random Tree	92,30%	96,89%	N/A
Nguyen et al. [15]	Random Forest	93,71%	98,80%	N/A
Tekerek et al. [17]	Static Method	N/A	87,93%	**69,00%**
Hao et al. [4]	Bi-LSTM	**98,35%**	N/A	N/A
Liang et al. [10]	RNN	85,15%	N/A	N/A
KARACAN and SEVRİ [7]	Bi-LSTM	**98,35%**	**98,94%**	87,66%
Mô hình đề xuất (HYDRA)	**CNN-HYDRA**	**97,45%**	**94,97%**	**92,64%**

Bảng 4.7: *So sánh kết quả phát hiện với các công trình nghiên cứu liên quan*

Approach	Accuracy			Precision			Recall			F1			Ghi chú
	1	2	3	1	2	3	1	2	3	1	2	3	batch_size=32
MODEL1	0.7046	0.7043	0.7016	0.6789	0.6447	0.5979	0.7046	0.7043	0.7016	0.5867	0.5881	0.5877	Epoch 5/10/15
MODEL2	0.7074	0.7167	0.7387	0.5715	0.6964	0.7318	0.7074	0.7167	0.7387	0.6042	0.6203	0.6809	
MODEL3	0.909	0.9071	0.9095	0.9182	0.9151	0.9171	0.909	0.9071	0.9095	0.9032	0.9013	0.9042	
Multimodal 1	0.7006	0.7208	0.7133	0.6418	0.6561	0.6639	0.7006	0.7208	0.7133	0.6198	0.6613	0.6777	MODEL1 - MODEL2
Multimodal 2	0.914	0.9035	0.8994	0.9198	0.9053	0.8997	0.914	0.9035	0.8994	0.9089	0.899	0.8949	MODEL1 - MODEL3
Multimodal 3	0.923	0.9201	0.9206	0.9245	0.9242	0.9246	0.923	0.9201	0.9206	0.919	0.9183	0.9164	MODEL2 - MODEL3
Multimodal 4	0.926	0.9217	0.9137	0.9276	0.9216	0.9187	0.926	0.9217	0.9137	0.9227	0.9203	0.9137	MODEL1 - MODEL2 - MODEL3

Hình 4.13: *So sánh kết quả khi xây dựng mô hình học sâu theo nhiều cách kết hợp khác nhau trên tập ECML/PKDD 2007 (Phân loại đa lớp).*

Hình 4.13 cho thấy các chỉ số ACC, PRE, RECALL, F1 qua các epochs (5/10/15) khi xây dựng mô hình học sâu theo nhiều hướng kết hợp khác nhau. Có thể thấy rằng khi chỉ sử dụng riêng lẻ từng model (MODEL1, MODEL2, MODEL3) sẽ không cho kết quả cao. Điều này chủ yếu là do dữ liệu đưa vào model chưa đủ đặc trưng, đa dạng để mô hình có thể phân loại tốt. Tuy nhiên, bằng cách kết hợp từ 2 đến 3 model với nhiều input đầu vào khác nhau (multi approach), ta sẽ cải thiện độ chính xác ACC cũng như các tiêu chí còn lại lên đến 92%.

4.4.2. Câu hỏi 2: Chọn ra mô hình phân loại đa lớp có kết quả tốt để triển khai vào mã nguồn của Modsecurity?

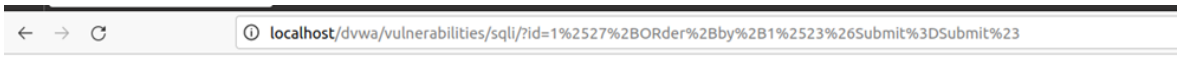
Model được nhóm sử dụng được train trên tập dataset ECML/PKDD 2007 (bao gồm 7 loại tấn công LdapInjection, OsCommanding, PathTransversal, SSI, SqlInjection, PathInjection, XSS). Dữ liệu đưa vào model được xử lý theo phương pháp NLP2 cho độ chính xác 93% trên tập ECML/PKDD 2007. Model sau khi train được lưu dưới dạng tensor sau đó được triển khai vào mô-đun Modsecurity nginx. Để thuận tiện thì ta sẽ tắt những rules của Modsecurity và chỉ cho HTTP/HTTPS request đi qua mô-đun học sâu HYDRA. Lúc này tường lửa ứng dụng web chỉ là mô hình HYDRA-WAFS mà nhóm đề xuất.

Quy trình hoạt động của hệ thống HYDRA-WAFS như sau:

1. Người dùng gửi HTTP/HTTPS request đến Web server bằng nhiều cách thức khác nhau

```
(base) modsecurity@ubuntu:~$ curl -i http://localhost/dvwa/vulnerabilities/sqli/?id=1%2527%2BORder%2Bby%2B1%2523%26Submit%3DSubmit
%23
```

Hình 4.14: *Curl request*



Hình 4.15: *Request từ trình duyệt*

2. HTTP/HTTPS request được bắt lại tại HYDRA-WAFS để xử lý dữ liệu, dữ liệu được xử lý sau đó đưa vào mô hình học sâu để dự đoán.

```
Request User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/114.0
Request Referer: Don't have Referer
Request Cookie: security=impossible; PHPSESSID=roe1o90pfk242tc9bj89oen1ho
Request Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Request HTTP Method: GET
Request Host: localhost
Request Body (from char*): (null)
Request HTTP Request: /dvwa/vulnerabilities/sqli/?id=1%2527%2BORder%2Bby%2B1%2523%26Submit%3DSubmit%23
Request HTTP Request (from char*): /dvwa/vulnerabilities/sqli/?id=1%2527%2BORder%2Bby%2B1%2523%26Submit%3DSubmit%23
Response HTTP Status Message:  OK
Response Content Length: 1370
```

Hình 4.16: *Dùng mô-đun nginx tự động bắt lại các trường cần thiết*

```
texts1:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/114.0

texts2:
(null)
security=impossible; PHPSESSID=roe1o90pfk242tc9bj89oen1ho

texts3:
localhost
/dvwa/vulnerabilities/sqli/?id=1%2527%2BORder%2Bby%2B1%2523%26Submit%3DSubmit%23
```

Hình 4.17: *Ba input đầu vào của mô hình bao gồm: texts1 (accept+user-agent), texts2(referer+cookie), texts3(host+url)*

```
Concatenated Text1: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 Mozilla/5.0 (X11; Ubuntu
; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/114.0
Concatenated Text2: security=impossible; PHPSESSID=roe1o90pfk242tc9bj89oen1ho
Concatenated Text3: localhost /dvwa/vulnerabilities/sqli/?id=1%2527%2BORder%2Bby%2B1%2523%26Submit%3DSubmit%23

Concatenated Text1 after decode: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 Mozilla/5.0
 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/114.0
Concatenated Text2 after decode: security=impossible; PHPSESSID=roe1o90pfk242tc9bj89oen1ho
Concatenated Text3 after decode: localhost /dvwa/vulnerabilities/sqli/?id=1'+ORder+by+1#&Submit=Submit#
```

Hình 4.18: *Nối chuỗi và decode các ký tự hex*

```
parse_concatenated_text1:  text / html , application / xhtml + xml , application / xml ; q = 0 . 9 , image / avif , image / webp ,
  *  /  *  ; q = 0 . 8 Mozilla / 5 . 0  ( X11 ;  Ubuntu ;  Linux x86_64 ;  rv : 109 . 0 )  Gecko / 20100101 Firefox / 114 . 0
parse_concatenated_text2:  security = impossible ;  PHPSESSID = roe1o90pfk242tc9bj89oen1ho
parse_concatenated_text3:  localhost  / dvwa / vulnerabilities / sqli /  ? id = 1 '  + ORder + by + 1 #  & Submit = Submit #
```

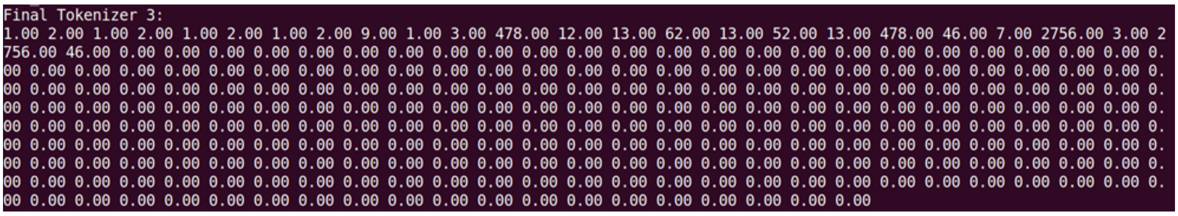
Hình 4.19: *Thêm khoảng trắng " " trước và sau ký tự đặc biệt. Sau đó dùng* ***strtok()*** *để tách các từ và ký tự đặc biệt dựa trên khoảng trắng và map index của từng từ với vị trí của nó trong file* ***dictionary.txt****.*

```
Final Tokenizer 1:
943.00 2.00 68.00 14.00 2198.00 2.00 440.00 13.00 71.00 14.00 2198.00 2.00 71.00 21.00 1057.00 3.00 458.00 5.00 520.00 14.00 302.0
0 2.00 1.00 14.00 302.00 2.00 1.00 14.00 30.00 2.00 30.00 21.00 1057.00 3.00 458.00 5.00 588.00 1.00 2.00 502.00 5.00 458.00 10.00
 11.00 21.00 1.00 21.00 1.00 11.00 21.00 1.00 26.00 6.00 5.00 458.00 8.00 1.00 2.00 6.00 1.00 2.00 6.00 5.00 458.00 0.00 0.00 0.00
 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

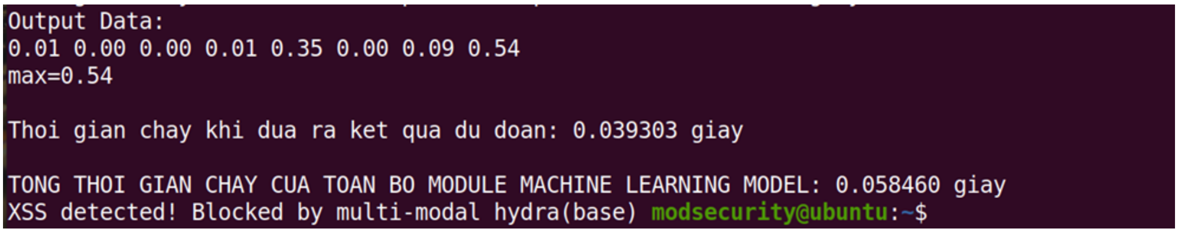
Hình 4.20: *Input thứ nhất của model*

```
Final Tokenizer 2:
1.00 3.00 1.00 21.00 1.00 3.00 11.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

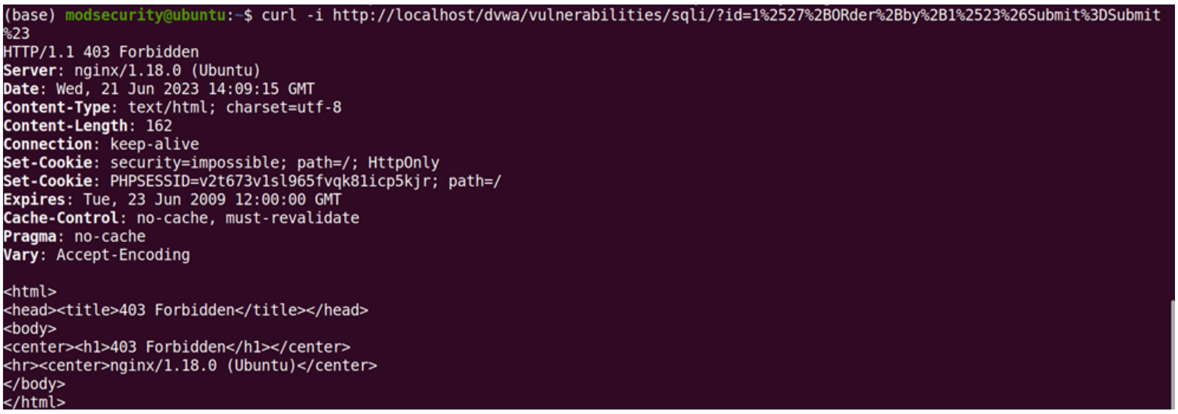
Hình 4.21: *Input thứ hai của model*



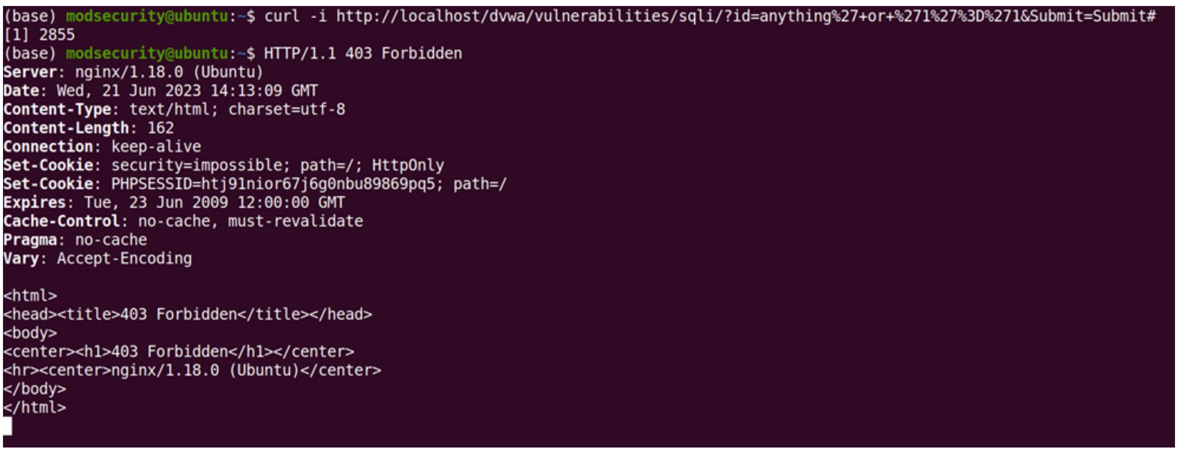
Hình 4.22: *Input thứ ba của model*



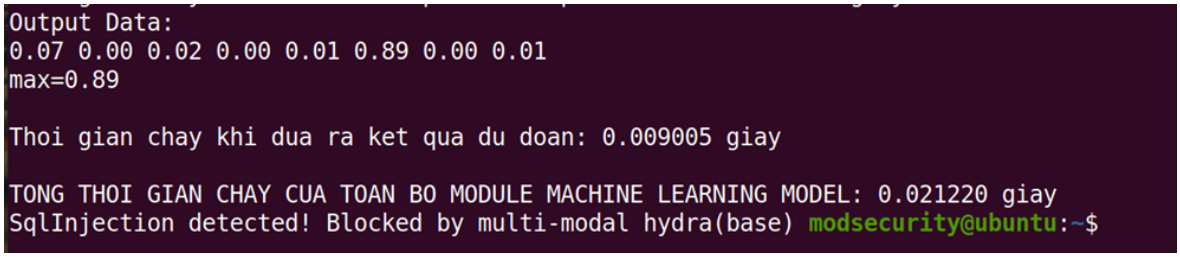
Hình 4.23: *Phát hiện, phân loại tấn công dựa trên ma trận xác xuất và đưa ra thông báo ngăn chặn.*



Hình 4.24: *Request curl bị chặn*

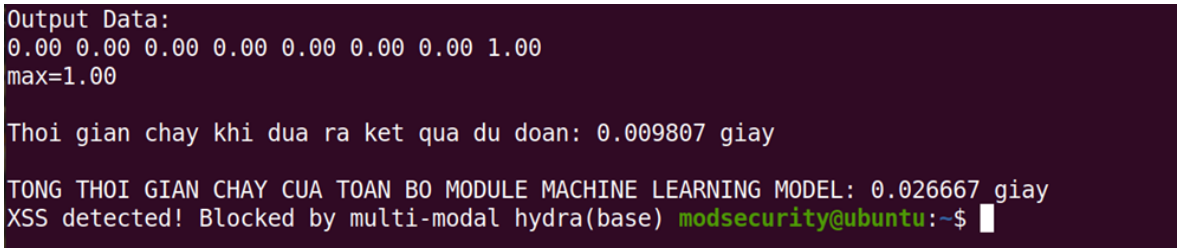


Hình 4.25: *Ngăn chặn tấn công SQLi*



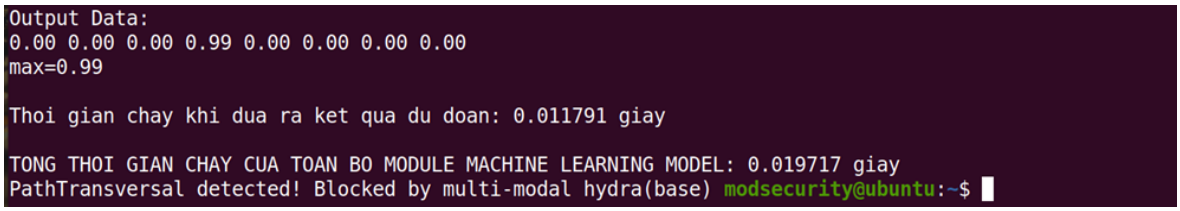
Hình 4.26: *Kết quả dự đoán tấn công SQLi (89%) với tổng thời gian đưa ra dự đoán là 0.0212s*

Hình 4.27: *Ngăn chặn tấn công XSS*



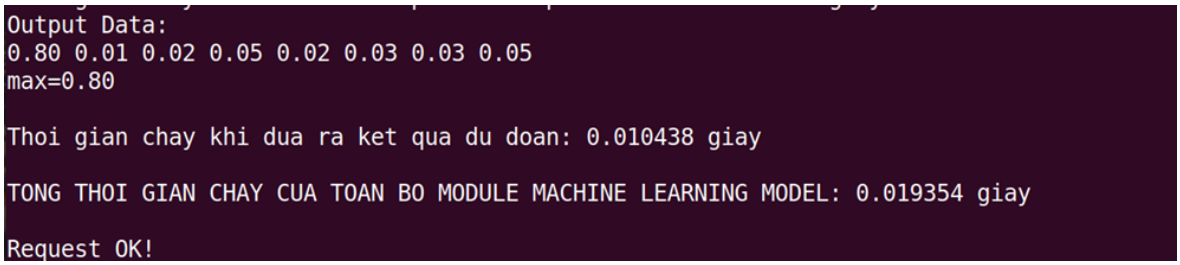
Hình 4.28: *Kết quả dự đoán tấn công XSS (100%) với tổng thời gian đưa ra dự đoán là 0.0266s*

Hình 4.29: *Ngăn chặn tấn công PathTraversal*



Hình 4.30: *Kết quả dự đoán tấn công PathTraversal (99%) với tổng thời gian đưa ra dự đoán là 0.0197s*

Hình 4.31: *Cho phép request bình thường đi qua.*



Hình 4.32: *Kết quả dự đoán bình thường (80%) với tổng thời gian đưa ra dự đoán là 0.0193s*

CHƯƠNG 5. KẾT LUẬN

Ở chương này, nhóm đưa ra những kết luận về nghiên cứu, những hạn chế, và đồng thời đưa ra hướng cải thiện và phát triển.

5.1. Kết luận

Nghiên cứu về áp dụng mô hình học máy vào tường lửa ứng dụng Web. Nhóm nhận thấy đa phần các nghiên cứu chỉ tập trung vào phân loại nhị phân hơn là phân loại đa lớp. Trong Đồ án, nhóm đã đề xuất ra hệ thống tường lửa ứng dụng Web có ứng dụng mô hình học sâu đa phương thức (HYDRA-WAFS).

Qua việc xây dựng hệ thống HYDRA-WAFS này có thể giúp tăng cường thêm tính bảo mật của WAF truyền thống bằng cách phát hiện tấn công trên nhiều thuộc tính khác nhau của một request. Đồ án này đã đạt được một số kết quả như sau:

- Tìm hiểu về cơ chế hoạt động của tường lửa ứng dụng Web truyền thống và tường lửa ứng dụng Web có ứng dụng mô hình học máy/học sâu.
- Tìm hiểu về những công trình nghiên cứu liên quan và những phương pháp tiếp cận khác nhau.
- Xây dựng hệ thống HYDRA-WAFS phát hiện tấn công Web.
- Minh chứng hiệu quả của mô hình khi phát hiện tấn công Web trong môi trường ảo hóa.

Kết quả mà nhóm thu được quả thực nghiệm cho thấy hệ thống HYDRA-WAFS hoàn toàn có khả năng ứng dụng trong mô hình thực tế trong việc phát hiện đa dạng các loại tấn công Web. Đồng thời, nhóm cũng chỉ ra hạn chế của các

nghiên cứu trước đó: việc phát hiện tấn công web chỉ dựa vào trường URL là không đủ khi những kẻ tấn công có thể thay đổi tùy thích các trường khác để đạt được mục đích xấu.

5.2. Hướng phát triển

So sánh các nghiên cứu trước đây trong cùng lĩnh vực, tuy mô hình đưa ra có thể cải thiện các nhược điểm hiện có (thông qua việc phân loại dựa trên nhiều đầu vào khác nhau) song kết quả thí nghiệm cho thấy chưa thật sự ấn tượng.

Ngoài ra, phương pháp xử lý dữ liệu NLP2 tuy có thể áp dụng trong điều kiện thực tế nhưng hiệu quả khi đưa vào mô hình chưa được như mong đợi. Dự kiến, nhóm sẽ cải tiến phương pháp này bằng cách tăng cường dữ liệu để mô hình có thể học được những đặc trưng tốt hơn.

Dự kiến, nhóm có thể xây dựng thêm 1 đến 2 mô hình học sâu như LSTM, RNN,... để kết hợp với kết quả của mô hình đề xuất xây dựng mô hình Ensemble Classifier để cải thiện kết quả phân loại cuối cùng.

TÀI LIỆU THAM KHẢO

Tiếng Anh:

[1] Joanne Bechta Dugan, Salvatore J Bavuso, and Mark A Boyd, "Fault trees and sequence dependencies", in: *Annual Proceedings on Reliability and Maintainability Symposium*, IEEE, 1990, pp. 286–293.

[2] Mohamed Amine Ferrag et al. (2020), "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study", *Journal of Information Security and Applications*, 50, p. 102419.

[3] Daniel Gibert, Carles Mateu, and Jordi Planes (2020), "HYDRA: A multimodal deep learning framework for malware classification", *Computers & Security*, 95, p. 101873.

[4] Saiyu Hao, Jun Long, and Yingchuan Yang, "Bl-ids: Detecting web attacks using bi-lstm model based on deep learning", in: *Security and Privacy in New Computing Environments: Second EAI International Conference, SPNCE 2019, Tianjin, China, April 13–14, 2019, Proceedings 2*, Springer, 2019, pp. 551–563.

[5] Debabrata Kar, Suvasini Panigrahi, and Srikanth Sundararajan, "SQLiDDS: SQL injection detection using query transformation and document similarity", in: *Distributed Computing and Internet Technology: 11th International Conference, ICDCIT 2015, Bhubaneswar, India, February 5-8, 2015. Proceedings 11*, Springer, 2015, pp. 377–390.

[6] Debabrata Kar, Suvasini Panigrahi, and Srikanth Sundararajan (2016), "SQLiGoT: Detecting SQL injection attacks using graph of tokens and SVM", *Computers & Security*, 60, pp. 206–225.

[7] Hacer Karacan and Mehmet Sevri (2021), "A Novel Data Augmentation Technique and Deep Learning Model for Web Application Security", *IEEE Access*, 9, pp. 150781–150797.

[8] Joost N Kok et al. (2007), *Knowledge Discovery in Databases: PKDD 2007: 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, Proceedings*, vol. 4702, Springer.

[9] Anh Le, Athina Markopoulou, and Michalis Faloutsos, "Phishdef: Url names say it all", in: *2011 Proceedings IEEE INFOCOM*, IEEE, 2011, pp. 191–195.

[10] Jingxi Liang, Wen Zhao, and Wei Ye, "Anomaly-based web attack detection: a deep learning approach", in: *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, 2017, pp. 80–85.

[11] Min-Sheng Lin et al., "Malicious URL filtering—A big data application", in: *2013 IEEE international conference on big data*, IEEE, pp. 589–596.

[12] Chao Liu, Jing Yang, and Jinqiu Wu (2020), "Web intrusion detection system combined with feature analysis and SVM optimization", *EURASIP Journal on Wireless Communications and Networking*, 2020, pp. 1–9.

[13] Paul R McWhirter et al. (2018), "SQL Injection Attack classification through the feature extraction of SQL query strings using a Gap-Weighted String Subsequence Kernel", *Journal of information security and applications*, 40, pp. 199–216.

[14] Hai Thanh Nguyen and Katrin Franke, "Adaptive Intrusion Detection System via online machine learning", in: *2012 12th international conference on hybrid intelligent systems (HIS)*, IEEE, 2012, pp. 271–277.

[15] Hai Thanh Nguyen et al. (2013), "Enhancing the effectiveness of web application firewalls by generic feature selection", *Logic Journal of IGPL*, 21 (4), pp. 560–570.

[16] Seungyoung Park, Myungjin Kim, and Seokwoo Lee (2018), "Anomaly detection for http using convolutional autoencoders", *IEEE Access*, 6, pp. 70884–70901.

[17] Chedy Raıssi et al., "Web analyzing traffic challenge: description and results", in: *Proceedings of the ECML/PKDD*, 2007, pp. 47–52.

[18] Tomás Sureda Riera et al. (2022), "A new multi-label dataset for Web attacks CAPEC classification using machine learning techniques", *Computers & Security*, 120, p. 102788.

[19] Bayu Adhi Tama et al. (2020), "An enhanced anomaly detection in web traffic using a stack of classifier ensemble", *IEEE Access*, 8, pp. 24120–24134.

[20] Adem Tekerek (2021), "A novel architecture for web-based attack detection using convolutional neural network", *Computers & Security*, 100, p. 102096.

[21] Nguyen Manh Thang (2020), "Improving efficiency of web application firewall to detect code injection attacks with random forest method and analysis attributes http request", *Programming and Computer Software*, 46, pp. 351–361.

[22] Camen Torrano-Giménez, Alejandro Perez-Villegas, and Gonzalo Alvarez Maranón (2010), "An anomaly-based approach for intrusion detection in web traffic".

[23] Ming Zhang et al., "A deep learning method to detect web attacks using a specially designed CNN", in: *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14–18, 2017, Proceedings, Part V 24*, Springer, 2017, pp. 828–836.

[24] Yun Zhou and Peichao Wang (2019), "An ensemble learning approach for XSS attack detection with domain knowledge and threat intelligence", *Computers & Security*, 82, pp. 261–269.