

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

NGUYỄN ĐÌNH KHA

KHÓA LUẬN TỐT NGHIỆP  
MỘT NGHIÊN CỨU VỀ HONEYPOD THÍCH ỦNG  
DỰA TRÊN HỌC TĂNG CUỜNG VÀ PHƯƠNG PHÁP  
DEEFDIG ĐỂ PHÁT HIỆN  
CÁC CUỘC TẤN CÔNG WEB NÂNG CAO

A STUDY ON AN ADAPTIVE HONEYPOD  
BASED ON REINFORCEMENT LEARNING AND THE  
DEEFDIG METHOD FOR DETECTING  
ADVANCED WEB ATTACKS

CỦ NHÂN NGÀNH AN TOÀN THÔNG TIN

TP. HỒ CHÍ MINH, 2024

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

NGUYỄN ĐÌNH KHA – 20520562

**KHÓA LUẬN TỐT NGHIỆP**  
**MỘT NGHIÊN CỨU VỀ HONEYBOT THÍCH ỦNG**  
**DỰA TRÊN HỌC TĂNG CƯỜNG VÀ PHƯƠNG PHÁP**  
**DEEFDIG ĐỂ PHÁT HIỆN**  
**CÁC CUỘC TẤN CÔNG WEB NÂNG CAO**  
**A STUDY ON AN ADAPTIVE HONEYBOT**  
**BASED ON REINFORCEMENT LEARNING AND THE**  
**DEEFDIG METHOD FOR DETECTING**  
**ADVANCED WEB ATTACKS**

CỦ NHÂN NGÀNH AN TOÀN THÔNG TIN

GIẢNG VIÊN HƯỚNG DẪN  
TS. NGUYỄN TÂN CẨM

TP. HỒ CHÍ MINH, 2024

## **THÔNG TIN HỘI ĐỒNG CHẤM KHÓA LUẬN TỐT NGHIỆP**

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số .....  
ngày ..... của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

## LỜI CẢM ƠN

Kính gửi hội đồng đánh giá luận văn tốt nghiệp, tôi tên là Nguyễn Đình Kha, mã số sinh viên 20520562, hiện tại tôi đang học lớp ATCL2020. Trước hết tôi xin được bày tỏ lòng biết ơn sâu sắc nhất tới hội đồng vì những hướng dẫn quý báu, sự hỗ trợ hết mình và những phản hồi sâu sắc trong xuyên suốt quá trình nghiên cứu đề tài này. Sự chuyên môn và động viên của quý hội đồng đã đóng góp vai trò quan trọng trong việc định hướng đi và nâng cao chất lượng của luận văn này.

Tôi xin bày tỏ lòng biết ơn sâu sắc tới các thầy cô và cán bộ giảng viên tại trường. Sự tận tình và nhiệt huyết của quý thầy cô đã xây dựng nên một nền tảng vững chắc cho sự phát triển và tiến bộ của tôi, và tôi chân thành cảm kích sự nỗ lực và khích lệ của quý thầy cô đã dành tặng cho tôi.

Đặc biệt, tôi xin gửi lời cảm ơn sâu sắc tới Tiến sĩ Nguyễn Tân Cầm, là người hướng dẫn cho luận văn của tôi, vì sự hỗ trợ và những hướng dẫn nhiệt tình. Mặc dù tôi không thường xuyên báo cáo trực tiếp với thầy, và tôi tiếp thu kiến thức khác chậm chạp và nhiều khi không hoàn thành kịp được tiến độ trong công việc, nhưng thầy vẫn chấp nhận và trao cơ hội, dưới sự hướng dẫn và cởi mở của thầy, thầy đã truyền đạt cho tôi quyết tâm vượt qua những rào cản kiến thức trong quá trình nghiên cứu. Những phản hồi mang tính tích cực của thầy đã phần nào tiếp thêm động lực cho tôi cố gắng hoàn thiện luận văn nhất có thể. Từ tận đáy lòng mình, tôi thực sự biết ơn sự hiểu biết và hỗ trợ của thầy.

Đối với gia đình và bạn bè, sự hỗ trợ của các bạn đã tiếp cho tôi động lực và sức mạnh, sự kiên trì vượt qua những thách thức, khó khăn gấp phải trên đường. Sự chia sẻ kiến thức từ những người bạn bè và niềm tin vững chắc của gia đình đã là nguồn sức mạnh và động lực lớn đối với tôi.

## MỤC LỤC

Chương 1. MỞ ĐẦU .....	2
1.1. Giới thiệu về an ninh mạng và mô phỏng web .....	2
1.1.1. Vai trò của ứng dụng web .....	2
1.1.2. Chiến lược an ninh mạng và honeypots .....	3
1.1.3. Những tiến bộ trong công nghệ honeypot .....	3
1.1.4. Học Tăng cường trong an ninh mạng.....	4
1.1.5. Nhu cầu về các giải pháp honeypot tiên tiến .....	4
1.1.6. Phương pháp DEEP-Dig: Công nghệ đánh lừa tiên tiến .....	5
1.2. Lí do chọn đề tài .....	6
1.3. Mục đích.....	7
1.3.1. Phát triển một Agent DQN để học tập và thích ứng .....	7
1.3.2. Tích hợp Phương pháp DEEP-Dig cho phân tích pháp chứng.....	7
1.3.3. Đánh giá hiệu quả và hiệu suất trong môi trường Kiểm soát.....	7
1.4. Đối tượng và phạm vi nghiên cứu .....	8
1.4.1. Đối tượng .....	8
1.4.1.1. Hệ thống honeypot thích ứng .....	8
1.4.1.2. Học tăng cường sâu (DRL).....	8
1.4.1.3. Các mối đe dọa an ninh web .....	9
1.4.1.4. Mô phỏng tấn công .....	9
1.4.1.5. Phân tích pháp chứng bởi DEEP-Dig.....	9
1.4.2. Phạm vi nghiên cứu .....	9
1.4.2.1. Phạm vi kỹ thuật.....	9
1.4.2.2. Phạm vi ứng dụng .....	9

1.4.2.3. Phạm vi thời gian.....	10
<b>Chương 2. TỔNG QUAN .....</b>	<b>11</b>
2.1. Giới thiệu.....	11
2.1.1. Hệ thống honeypot.....	11
2.1.2. Ứng dụng của RL trong an ninh mạng.....	12
2.1.2.1. Các thuật toán chính trong RL: .....	12
2.1.2.2. Ví dụ về ứng dụng của RL trong an ninh mạng: .....	13
2.1.3. Phân tích pháp chứng.....	13
2.1.3.1. Tầm quan trọng trong an ninh mạng .....	14
2.1.3.2. Phương pháp và công cụ.....	14
2.1.4. Tích hợp DEEP-Dig .....	14
2.2. Hướng nghiên cứu hiện tại.....	15
2.2.1. Hướng nghiên cứu trong nước .....	15
2.2.2. Hướng nghiên cứu quốc tế .....	15
<b>Chương 3. HỆ THỐNG ĐÈ XUẤT .....</b>	<b>17</b>
3.1. Thiết lập và triển khai hệ thống.....	17
3.1.1. Kiến trúc tổng thể .....	17
3.1.2. Phát triển người dùng.....	17
3.1.2.1. Các thành phần và tính năng .....	18
3.1.2.2. Chi tiết triển khai .....	23
3.1.3. Các thành phần của hệ thống Honeypot thích ứng .....	24
3.1.3.1. Thu thập dữ liệu (Data collection).....	24
3.1.3.2. Tiền xử lý dữ liệu (Data Processing).....	25
3.1.3.3. Phân tích vector tấn công (Attack vector analysis) .....	25

3.1.3.4.    Báo cáo (Reporting) .....	26
3.2.    Tổng quan về mô hình DQN .....	26
3.2.1.    Hàm Q .....	27
3.2.2.    Mạng nơ-ron sâu.....	27
3.2.3.    Replay memory .....	27
3.2.4.    Target network .....	27
3.3.    Mô tả chi tiết DQN Agent .....	27
3.3.1.    Class SumTree.....	27
3.3.2.    Class Memory.....	28
3.3.3.    Class DQN_Agent.....	29
3.3.3.1.    Các thành phần của DQN Agent.....	30
3.3.3.2.    Chi tiết triển khai .....	31
3.3.3.3.    Đánh giá hiệu suất của agent qua các chỉ số Accuracy, Precision, Recall và F1 Score.....	36
3.4.    Hệ thống đề xuất.....	37
3.4.1.    Kiến trúc mô hình DQN Agent.....	38
3.4.2.    Luồng workflow của DQN Agent .....	39
3.4.2.1.    Phát hiện tấn công: .....	39
3.4.2.2.    Huấn luyện và đánh giá: .....	40
3.4.2.3.    Ứng dụng thực tế:.....	40
3.5.    Kết quả thực nghiệm .....	40
Chương 4.    TRÌNH BÀY, ĐÁNH GIÁ BÀN LUẬN VỀ KẾT QUẢ .....	41
4.1.    Đánh giá và xác thực .....	41
4.1.1.    Mục đích .....	41

4.1.2.    Các chỉ số .....	41
4.1.3.    Quy trình.....	42
4.1.3.1.    Thiết lập thí nghiệm: .....	42
4.1.3.2.    Thu thập dữ liệu:.....	42
4.1.3.3.    Phân tích:.....	43
4.1.3.4.    Kết quả:.....	43
4.2.    Quy trình kiểm tra .....	45
4.2.1.    Thực thi tấn công.....	45
4.2.1.1.    Đối với tấn công SQL Injection: .....	45
4.2.1.2.    Đối với tấn công XSS:.....	52
Chương 5.    KẾT LUẬN .....	59
5.1.    Tóm tắt kết quả.....	59
5.2.    Những đóng góp chính .....	60
Chương 6.    HƯỚNG PHÁT TRIỂN.....	62
6.1.    Mở rộng các kịch bản tấn công: .....	62
6.2.    Cải thiện mô hình DQN .....	63

## DANH MỤC HÌNH

Hình 1. Kiến trúc tổng thể của hệ thống honypot tích hợp Học Tăng cường. ....	17
Hình 2. Giao diện trang chủ của ứng dụng web ngân hàng.....	18
Hình 3. Giao diện người dùng đăng nhập để truy cập vào tài khoản. ....	19
Hình 4. Giao diện tài khoản tổng quan của người dùng. ....	20
Hình 5. Giao diện chuyển khoản nơi người dùng có thể nhấp vào tài khoản thụ thưởng với số tiền tương ứng cần chuyển cho người dùng muốn nhận. ....	20
Hình 6. Khi người dùng chuyển tiền thành công sẽ có thông báo xác nhận và khi người dùng tắt nó thì hệ thống sẽ tự load lại trang để cập nhật lại số dư khả dụng của người dùng sau khi chuyển khoản. ....	21
Hình 7. Giao diện thông tin tài khoản của người dùng. ....	21
Hình 8. Giao diện lịch sử giao dịch, nơi người dùng có thể tra cứu lại những giao dịch đã thực hiện được trong thời gian vừa qua. ....	22
Hình 9. Giao diện phản hồi đáp cho người dùng.....	23
Hình 10. Biểu diễn trạng thái của DQN Agent.....	31
Hình 11. Quá trình huấn luyện DQN Agent được huấn luyện bằng cách sử dụng phát lại trải nghiệm ưu tiên.....	34
Hình 12. Kiến trúc mô hình DQN Agent .....	38
Hình 13. Kịch bản tấn công SQL Injection .....	46
Hình 14. Adam chuẩn bị các terminal để chạy các script tấn công. ....	48
Hình 15. Các script python được chạy để thực hiện các cuộc tấn công SQL Injection. ....	49
Hình 16. Agent ghi nhận và phản hồi ngay sau khi phát hiện tấn công SQL Injection. ....	49
Hình 17. Tổng số cuộc tấn công SQL Injection Agent đã phát hiện được.....	50
Hình 18. Biểu đồ đánh giá hiệu suất của agent DQN đối với tấn công SQL Injection. ....	51
Hình 19. Kịch bản tấn công XSS .....	53

Hình 20. Bob mở Burp Suite và cấu hình Intercepting Proxy để bắt và sửa đổi các yêu cầu HTTP.....	55
Hình 21. Bob chuẩn bị 120 payload XSS, bao gồm các mã JavaScript độc hại như <SCRIPT SRC=http://ha.ckers.org/xss.js></SCRIPT><IMGSRC="javascript:alert('XSS');"> và các loại mã script khác.....	55
Hình 22. Bob sử dụng Burp Suite để chặn các yêu cầu HTTP gửi từ trình duyệt đến hệ thống Honeypot.....	56
Hình 23. Bob chèn các payload XSS vào các trường đầu vào mật khẩu và gửi các yêu cầu đã chỉnh sửa đến máy chủ.....	56
Hình 24. Bob tiến hành tấn công inject payload vào trường đầu vào password. ....	57
Hình 25. Agent ghi nhận và phản hồi ngay sau khi phát hiện tấn công XSS.....	57

## **DANH MỤC BẢNG**

Bảng 1. Kết quả đánh giá hiệu suất của agent DQN đối với phát hiện tấn công SQL Injection.....	51
--	----

## **DANH MỤC TỪ VIẾT TẮT**

RL – Reinforcement Learning

DQN – Deep Q-Network

DRL – Deep Reinforcement Learning

SQL – Structured Query Language

XSS – Cross-site scripting

CSRF – Cross-site request forgery

IDS – Intrusion Detection System

OTP – One-time password

PER – Prioritized Experience Replay

MSE – Mean Squared Error

## TÓM TẮT KHÓA LUẬN

Luận văn tốt nghiệp của tôi trình bày một nghiên cứu toàn diện về phát triển và đánh giá hệ thống honeypot thích ứng sử dụng Học Tăng cường (Reinforcement Learning - RL) và phương pháp DEEP-Dig để phát hiện các cuộc tấn công web tiên tiến. Mục tiêu chính của nghiên cứu này là tạo ra một hệ thống thông minh có thể học hỏi và thích ứng động với các mẫu tấn công khác nhau, nâng cao hiệu quả trong việc xác định và giảm thiểu các mối đe dọa mạng tinh vi. Nghiên cứu bắt đầu với cái nhìn tổng quan về an ninh mạng và tầm quan trọng của các ứng dụng web, nhấn mạnh nhu cầu ngày càng tăng về các biện pháp an ninh tiên tiến như honeypot. Bài luận xem xét các nghiên cứu hiện có về honeypot, RL trong an ninh mạng, và các kỹ thuật phân tích pháp chứng, xác định các khoảng trống trong các phương pháp hiện tại. Trọng tâm của luận văn nằm ở khâu thiết kế và triển khai hệ thống honeypot thích ứng với các thành phần chính như DQN Agent chịu trách nhiệm phát hiện và phản ứng với các cuộc tấn công web, Experience Replay Buffer dùng để lưu trữ kinh nghiệm, và phân tích pháp chứng với DEEP-Dig để khám phá các chiến lược tấn công. Hệ thống được đánh giá trong các kịch bản tấn công khác nhau như SQL injection, XSS, và CSRF, với các chỉ số đánh giá bao gồm độ chính xác, thời gian phản hồi, và khả năng thích ứng. Kết quả cho thấy hệ thống honeypot thích ứng có độ chính xác tương đối trung bình và thời gian phản hồi nhanh, chứng tỏ hiệu quả trong phát hiện và giảm thiểu các mối đe dọa, đồng thời cung cấp các thông tin pháp chứng có giá trị để cải thiện các biện pháp an ninh tổng thể.

# Chương 1. MỞ ĐẦU

## 1.1. Giới thiệu về an ninh mạng và mô phỏng web

Trong bối cảnh kĩ thuật số hiện đại, an ninh mạng đã nổi lên như một mối quan tâm hàng đầu. Sự tiến bộ không ngừng nghỉ của công nghệ đã làm tăng nhanh chóng khối lượng và độ phức tạp của các mối đe dọa mạng. Trước đây, an ninh mạng chủ yếu tập trung vào việc bảo vệ chống lại phần mềm độc hại cơ bản và những truy cập trái phép. Tuy nhiên, bối cảnh mới đe dọa hiện đại được đặc trưng bởi các cuộc tấn công tinh vi sử dụng các kỹ thuật tiên tiến để khai thác lỗ hổng trong hệ thống và ứng dụng.

An ninh mạng bao gồm một loạt các biện pháp nhằm bảo vệ mạng, thiết bị, chương trình và dữ liệu khỏi các cuộc tấn công, hư hỏng hoặc truy cập trái phép. Tầm quan trọng của nó không thể bị đánh giá thấp được, với những hậu quả tiềm tàng và các mối đe dọa an ninh quốc gia. Khi các tổ chức ngày càng dựa vào cơ sở hạ tầng kỹ thuật số, nhu cầu về các biện pháp an ninh mạng mạnh mẽ chưa bao giờ lớn hơn.

### 1.1.1. Vai trò của ứng dụng web

Các ứng dụng web đã trở thành một phần không thể thiếu trong nhiều khía cạnh của cuộc sống hàng ngày, từ ngân hàng cá nhân và thương mại điện tử đến mạng xã hội truyền thông và hoạch định nguồn lực doanh nghiệp. Những ứng dụng này tạo điều kiện thuận lợi cho sự tương tác và giao dịch liền mạch qua internet, cung cấp các dịch vụ quan trọng cho người dùng trên toàn thế giới. Tuy nhiên, sự hiện diện phổ biến và kết nối của chúng khiến chúng trở thành mục tiêu chính của các kẻ tấn công mạng.

Tính chất mở của các ứng dụng web, cùng với sự trao đổi thường xuyên của thông tin cá nhân, giới thiệu nhiều lỗ hổng cho các kẻ tấn công khai thác. Các lỗ hổng phổ biến trong các ứng dụng web bao gồm SQL Injection, cross-site scripting (XSS), cross-site-request forgery (CSRF) và nhiều lỗ hổng bảo mật web khác. Mỗi lỗ hổng này có thể bị khai thác để truy cập trái phép, trích xuất dữ liệu hoặc làm gián đoạn dịch vụ.

### **1.1.2. Chiến lược an ninh mạng và honeypots**

Nhằm chống lại vô vàn mối đe dọa đối với các ứng dụng web, các chiến lược an ninh mạng phải đa dạng và linh hoạt. Các biện pháp phòng thủ truyền thống như tường lửa, hệ thống phát hiện xâm nhập (IDS) và phần mềm chống virus tạo thành tuyến phòng thủ đầu tiên. Tuy nhiên, các biện pháp này thường không đủ đối phó với các cuộc tấn công zero-day và vector tấn công tinh vi liên tục tiến hóa.

Honeypots đại diện cho một cách tiếp cận chủ động trong an ninh mạng, được thiết kế để lừa dối và phân tích kẻ tấn công bằng cách mô phỏng các hệ thống dễ bị tổn thương. Một honeypot là một hệ thống mồi nhử được cố tình phơi bày trên internet để thu hút các kẻ tấn công mạng. Bằng cách giám sát các tương tác với honeypot, các chuyên gia an ninh có thể thu thập những hiểu biết vô giá về phương pháp tấn công, xu hướng.

Honeypots có thể được phân loại thành honeypot tương tác cao và tương tác thấp. Honeypot tương tác cao mô phỏng các hệ thống hoàn chỉnh, cung cấp nhiều cơ hội tương tác cho kẻ tấn công, do đó cung cấp dữ liệu tấn công chi tiết. Honeypot tương tác thấp, mặt khác, mô phỏng các dịch vụ hoặc lỗ hổng cụ thể, cung cấp tương tác hạn chế nhưng yêu cầu ít tài nguyên hơn để duy trì.

### **1.1.3. Những tiến bộ trong công nghệ honeypot**

Hiệu quả của các honeypot truyền thống thường bị giới hạn bởi tính tĩnh của chúng. Honeypot tĩnh có thể dễ dàng bị phát hiện và vượt qua bởi kẻ tấn công sử dụng các kỹ thuật tiên tiến nhằm xác định danh tính. Để khắc phục hạn chế này, các honeypot thích ứng đã được phát triển, sử dụng các công nghệ tiên tiến như học máy và trí tuệ nhân tạo để thay đổi hành vi của chúng một cách linh hoạt để phản ứng với các mối đe dọa được phát triển.

Honeypot thích ứng có thể mô phỏng nhiều kịch bản và điều chỉnh phản ứng của chúng dựa trên các hành động quan sát được của kẻ tấn công. Sự tương tác động này không chỉ làm cho kẻ tấn công khó nhận ra mồi nhử mà còn tăng cường khả năng của honeypot trong việc thu thập thông tin chi tiết về các kỹ thuật tấn công tiên tiến.

#### **1.1.4. Học Tăng cường trong an ninh mạng**

Học Tăng cường (RL), chính là một phân nhánh của học máy, cho thấy tiềm năng lớn trong việc tăng cường khả năng thích ứng và trí thông minh của honeypot. Trong RL, một agent học cách đưa ra quyết định bằng cách thực hiện các hành động trong một môi trường và nhận phản hồi dưới dạng phần thưởng (reward) hoặc hình phạt. Cách tiếp cận đúng và sai này cho phép agent phát triển các chiến lược tối ưu hóa phần thưởng dài hạn.

Trong bối cảnh an ninh mạng, RL có thể được sử dụng để tạo ra các honeypot thích ứng học hỏi từ các tương tác liên tục với kẻ tấn công mạng. Bằng cách liên tục cập nhật các chiến lược của mình dựa trên dữ liệu thời gian thực, các honeypot dựa trên RL có thể đối phó hiệu quả với các mối đe dọa tiến hóa và cung cấp những hiểu biết sâu hơn về hành vi của kẻ tấn công.

#### **1.1.5. Nhu cầu về các giải pháp honeypot tiên tiến**

Đứng trước sự tinh vi của các mối đe dọa mạng hiện đại, có nhu cầu cấp bách về các giải pháp honeypot tiên tiến có thể thích ứng với các mô hình tấn công thay đổi. Các honeypot tĩnh truyền thống không đủ đối phó với các đối thủ thông minh sử dụng các công cụ tự động và kỹ thuật do trí tuệ nhân tạo điều khiển. Một honeypot thích ứng, được hỗ trợ bởi học tăng cường, đại diện cho một bước tiến đáng kể trong lĩnh vực này.

Bằng cách tích hợp học tăng cường, một honeypot thích ứng có thể thay đổi hành vi của mình một cách linh hoạt, làm cho nó trở nên linh hoạt hơn trước các kỹ thuật phát hiện và né tránh. Ngoài ra, việc sử dụng các phương pháp phân tích pháp chứng như DeepDig cho phép kiểm tra và giải thích toàn diện dữ liệu tấn công, cung cấp những hiểu biết hành động để cải thiện các biện pháp an ninh một cách tổng thể. Phân tích pháp chứng hiệu quả là điều quan trọng để hiểu rõ phạm vi và tác động của các cuộc tấn công mạng. Các phương pháp phân tích truyền thống thường tập trung vào việc phân tích nhật ký và dữ liệu được thu thập bởi các hệ thống bảo mật tĩnh, có thể bị giới hạn về phạm vi và độ sâu. Phương pháp DEEP-Dig cung cấp một cách tiếp

cận phân tích pháp chứng tinh vi hơn bằng cách cung cấp một khuôn khổ toàn diện để đào sâu vào các mẫu và hành vi tấn công.

#### **1.1.6. Phương pháp DEEP-Dig: Công nghệ đánh lừa tiên tiến**

Phương pháp DEEP-Dig (DEcEPtion DIGging)<sup>[4]</sup><sup>[5]</sup>, được phát triển bởi một nhóm các nhà nghiên cứu bao gồm Tiến sĩ Kevin Hamlen và Tiến sĩ Latifur Khan, đại diện cho một bước tiến quan trọng trong lĩnh vực an ninh mạng. DEEP-Dig sử dụng công nghệ đánh lừa để lôi kéo kẻ tấn công vào một môi trường mồi nhử, cho phép hệ thống học hỏi từ các chiến thuật và chiến lược của kẻ tấn công. Phương pháp này đặc biệt hiệu quả trong việc thu thập dữ liệu tấn công chi tiết và có giá trị, có thể được sử dụng để đào tạo các mô hình học máy nhằm cải thiện khả năng phát hiện và phản ứng với mối đe dọa.

#### **Kỹ thuật dựa trên đánh lừa**

DEEP-Dig áp dụng các kỹ thuật dựa trên đánh lừa để tạo ra một môi trường mồi nhử thực tế và hấp dẫn. Môi trường này được thiết kế để thu hút kẻ tấn công, những kẻ vô tình tiết lộ các phương pháp và chiến thuật của chúng. Bằng cách nghiên cứu các tương tác này, hệ thống có thể thu thập được những hiểu biết sâu sắc về các kỹ thuật và chiến lược tấn công mới nhất.

#### **Phân tích sâu**

DEEP-Dig cung cấp các công cụ để thực hiện phân tích sâu dữ liệu tấn công đã thu được. Điều này bao gồm các kỹ thuật để nhận diện và hiểu các mẫu tấn công phức tạp, liên kết các giai đoạn khác nhau của các cuộc tấn công nhiều giai đoạn, và khám phá các kết nối ẩn giữa các sự kiện tưởng chừng như không liên quan.

#### **Học thích nghi**

Bằng cách tích hợp với hệ thống honeypot thích nghi, DEEP-Dig có thể tận dụng các thuật toán học máy để liên tục cải thiện khả năng phát hiện và phân tích của mình. Cách tiếp cận học thích nghi này đảm bảo rằng hệ thống vẫn hiệu quả đối với các mối đe dọa mới và đang phát triển.

DEEP-Dig thúc đẩy một lĩnh vực đang phát triển nhanh chóng được gọi là công nghệ đánh lừa, hay "crook sourcing", bao gồm việc đặt bẫy cho các hacker. Các nhà nghiên cứu hy vọng rằng phương pháp này có thể đặc biệt hữu ích cho các tổ chức phòng thủ. Thay vì chỉ chặn kẻ tấn công, DEEP-Dig coi chúng là một nguồn lao động miễn phí, cung cấp dữ liệu về các cuộc tấn công độc hại. Dữ liệu quý giá này sau đó có thể được sử dụng để đào tạo máy tính nhận diện và ngăn chặn các cuộc tấn công trong tương lai.

## 1.2. Lí do chọn đề tài

Sự quan tâm của tôi đối với an ninh mạng và học máy đã thúc đẩy tôi khám phá các giải pháp sáng tạo cho những thách thức an ninh hiện đại. Thực thú vị khi được là một phần của lĩnh vực đóng vai trò quan trọng đối với sự thành công của các tổ chức và việc bảo vệ tài sản của họ. Nghiên cứu này không chỉ nhằm đóng góp vào mục đích học thuật mà còn hướng đến việc cung cấp các giải pháp thực tế có thể triển khai trong các tình huống thực tế. Bằng cách phát triển một hệ thống honeypot mạnh mẽ và thích ứng, luận văn này nhằm nâng cao bảo mật cho các ứng dụng web và bảo vệ chống lại các mối đe dọa mạng tiên tiến.

Động lực chính của tôi cho việc lựa chọn đề tài nghiên cứu này bắt nguồn từ bản chất liên tục và phát triển của các mối đe dọa mạng nhắm vào các ứng dụng web. Các hệ thống honeypot truyền thống, được thiết kế để thu hút và giám sát kẻ tấn công, song chúng thường thiếu khả năng thích ứng động cần thiết để phản ứng hiệu quả với các cuộc tấn công tinh vi. Bằng cách tích hợp DRL, cụ thể là một agent Deep Q-Network (DQN), với công nghệ honeypot, nghiên cứu này nhằm phát triển một hệ thống có khả năng học hỏi và thích nghi với các mẫu tấn công mới trong thời gian thực. Hơn nữa, việc kết hợp phương pháp DEEP-Dig nâng cao khả năng pháp chứng của hệ thống, giúp đánh lừa kẻ tấn công, cung cấp cái nhìn sâu sắc hơn về hành vi tấn công và cải thiện khả năng phát hiện và giảm thiểu mối đe dọa tổng thể.

### **1.3. Mục đích**

Mục đích chính của nghiên cứu này là thiết kế và triển khai một hệ thống honeypot thích ứng sử dụng Học tăng cường sâu (DRL) và phương pháp DEEP-Dig để phát hiện và phản ứng hiệu quả với các cuộc tấn công web nâng cao. Các mục tiêu cụ thể bao gồm:

#### **1.3.1. Phát triển một Agent DQN để học tập và thích ứng**

Với sự phát triển nhanh chóng của các kỹ thuật tấn công web, các honeypot tĩnh truyền thống trở nên kém hiệu quả hơn. Một cách tiếp cận động, dựa trên học tập là cần thiết để thích ứng với các mô hình tấn công mới. Phương pháp được nhắm đến chính là triển khai một agent Deep Q-Network (DQN) có khả năng học từ các tương tác với kẻ tấn công. Agent này sẽ được huấn luyện bằng cách sử dụng các mẫu tấn công web khác nhau, chẳng hạn như SQL injection, Cross-Site Scripting (XSS), và Cross-Site Request Forgery (CSRF). Kết quả mong đợi từ agent này sẽ có khả năng phát hiện và phản ứng với các cuộc tấn công này với độ chính xác cao và thích ứng với các mẫu mới thông qua việc học liên tục, từ đó nâng cao an ninh cho các ứng dụng web.

#### **1.3.2. Tích hợp Phương pháp DEEP-Dig cho phân tích pháp chứng**

Phân tích pháp chứng là rất quan trọng để hiểu hành vi và kỹ thuật của kẻ tấn công. Phương pháp DEEP-Dig cung cấp một khung phân tích pháp chứng và lừa đảo sâu mạnh mẽ. Cách thức được sử dụng chính là tích hợp DEEP-Dig vào hệ thống honeypot để thu thập thông tin chi tiết về các vector và chiến lược tấn công. Điều này bao gồm thiết lập 20 dữ liệu cá nhân giả mạo để có thể đánh lừa kẻ tấn công.

#### **1.3.3. Đánh giá hiệu quả và hiệu suất trong môi trường Kiểm soát**

Đánh giá hệ thống trong môi trường kiểm soát cho phép kiểm tra nghiêm ngặt khả năng và hiệu suất của nó dưới các kịch bản khác nhau. Tôi đã thiết lập một môi trường thử nghiệm với các biến số kiểm soát để kiểm tra honeypot thích ứng. Đo lường các chỉ số như độ chính xác phát hiện, thời gian phản hồi, khả năng thích ứng với các

mẫu tấn công mới và khả năng chống chịu của hệ thống. Kết quả đầu ra mong đợi sẽ là dữ liệu định lượng chứng minh hiệu quả của hệ thống, bao gồm độ chính xác phát hiện cao và thời gian phản hồi nhanh. Dữ liệu này sẽ rất quan trọng để xác thực hiệu suất của hệ thống và xác định các lĩnh vực cần cải thiện.

Một phân tích kỹ lưỡng về hiệu suất của hệ thống là cần thiết để hiểu đầy đủ điểm mạnh và điểm yếu của nó. Phân tích này sẽ cung cấp cái nhìn sâu sắc về cách hệ thống có thể được cải thiện và thích ứng cho các mối đe dọa trong tương lai. Phương pháp được thực hiện chính là phân tích dữ liệu thử nghiệm để xác định các mẫu, điểm mạnh và điểm yếu. So sánh hiệu suất của hệ thống với các giải pháp hiện có để làm nổi bật những ưu điểm và hạn chế của nó. Từ đó có một báo cáo chi tiết về hiệu suất của hệ thống, bao gồm các khuyến nghị cho các cải tiến trong tương lai. Phân tích này sẽ giúp tinh chỉnh hệ thống và xác định hướng nghiên cứu trong tương lai.

## 1.4. Đối tượng và phạm vi nghiên cứu

### 1.4.1. Đối tượng

Đối tượng nghiên cứu của bài khóa luận này bao gồm:

#### 1.4.1.1. Hệ thống honeypot thích ứng

Khám phá các hệ thống honeypot được thiết kế để thu hút và phân tích các cuộc tấn công mạng bằng cách mô phỏng các hệ thống thực trong khi được cách ly với các hệ thống sản xuất thực tế. Tập trung vào các honeypot thích ứng có thể thay đổi hành vi của chúng một cách động dựa trên các cuộc tấn công mà chúng gặp phải.

#### 1.4.1.2. Học tăng cường sâu (DRL)

Tôi đã lựa chọn thuật toán DQN (Deep Q – Network) [1] , tiến hành nghiên cứu và triển khai của nó và sự phù hợp của nó trong việc phát hiện và phản ứng với các cuộc tấn công web. Nghiên cứu sâu vào cách DQN có thể được sử dụng để cho phép honeypot học từ các mẫu tấn công và cải thiện phản ứng của nó theo thời gian.

**Các chỉ số hiệu suất:** Hiểu các chỉ số được sử dụng để đánh giá hiệu suất của các mô hình DRL trong các ứng dụng an ninh mạng, như độ chính xác phát hiện, khả năng thích ứng và thời gian phản ứng.

#### **1.4.1.3. Các mối đe dọa an ninh web**

Phân tích các cuộc tấn công web phổ biến và tiên tiến như SQL Injection, Cross-Site Scripting (XSS) và Cross-Site Request Forgery (CSRF). Bao gồm việc hiểu cơ chế của chúng, kỹ thuật phát hiện và chiến lược giảm thiểu.

#### **1.4.1.4. Mô phỏng tấn công**

Sử dụng các công cụ như SQLMap, Burp Suite tạo các script tùy chỉnh để mô phỏng các cuộc tấn công này chống lại hệ thống honeypot để huấn luyện và đánh giá.

#### **1.4.1.5. Phân tích pháp chứng bối DEEP-Dig**

Khám phá cách phương pháp DEEP-Dig có thể được tích hợp vào hệ thống honeypot cho phân tích pháp chứng. Bằng cách tạo ra những dữ liệu cá nhân giả mạo, giúp honeypot trong việc đánh lừa kẻ tấn công khiến chúng nghĩ rằng chúng đã lấy được dữ liệu người dùng thực tế.

### **1.4.2. Phạm vi nghiên cứu**

#### **1.4.2.1. Phạm vi kỹ thuật**

Nghiên cứu tập trung vào việc phát triển và triển khai hệ thống honeypot thích ứng sử dụng Học tăng cường sâu (DRL). Phạm vi bao gồm việc thiết kế kiến trúc hệ thống, phát triển các thành phần chính, và tích hợp phương pháp DEEP-Dig để phân tích pháp chứng.

#### **1.4.2.2. Phạm vi ứng dụng**

Hệ thống được triển khai và thử nghiệm trong môi trường kiểm soát, mô phỏng các cuộc tấn công thực tế nhằm đánh giá hiệu quả và hiệu suất. Các tình huống kiểm tra bao gồm các cuộc tấn công SQL Injection, XSS, và CSRF.

#### **1.4.2.3. Phạm vi thời gian**

Nghiên cứu được thực hiện trong khoảng thời gian từ ngày 3 tháng 3 đến ngày 30 tháng 6 năm 2024, bao gồm các giai đoạn phát triển, triển khai, thử nghiệm, và đánh giá.

## Chương 2. TỔNG QUAN

### 2.1. Giới thiệu

Tần suất và độ tinh vi ngày càng tăng của các cuộc tấn công mạng đã đòi hỏi sự phát triển của các cơ chế phòng thủ tiên tiến. Các biện pháp bảo mật truyền thống, như tường lửa và hệ thống phát hiện xâm nhập, thường không đủ để phát hiện và giảm thiểu các cuộc tấn công mới và tinh vi. Điều này đã dẫn đến việc khám phá các hệ thống honeypot, được thiết kế để thu hút kẻ tấn công và phân tích hành vi của chúng mà không gây rủi ro cho các hệ thống sản xuất thực tế. Trong bối cảnh này, việc tích hợp các kỹ thuật học máy, đặc biệt là học tăng cường (RL), và các phương pháp phân tích pháp chứng như DEEP-Dig, đại diện cho một cách tiếp cận đầy hứa hẹn để nâng cao hiệu quả của honeypots trong việc phát hiện các cuộc tấn công web tiên tiến.

#### 2.1.1. Hệ thống honeypot

Honeypots là các cơ chế bảo mật tạo ra một hệ thống mồi để thu hút kẻ tấn công, ghi lại các hoạt động và hành vi của chúng. Honeypot truyền thống là các hệ thống tĩnh được thiết kế để mô phỏng các mục tiêu tiềm năng, thường ghi lại một loạt các hoạt động độc hại. Tuy nhiên, các honeypot truyền thống này có thể trở nên kém hiệu quả hơn theo thời gian khi kẻ tấn công học cách nhận biết và tránh chúng. Ngược lại, honeypot thích ứng sử dụng các kỹ thuật động để tiến hóa để đáp ứng các mẫu tấn công mới, làm cho chúng bén bỉ và hiệu quả hơn trong việc bắt các mối đe dọa tinh vi.

Honeypot truyền thông thường được thiết lập để mô phỏng các hệ thống dễ bị tấn công. Chúng cung cấp những hiểu biết có giá trị về hành vi của kẻ tấn công và phần mềm độc hại mới. Tuy nhiên, tính chất tĩnh của chúng có nghĩa là chúng không thể thích ứng với các loại tấn công hoặc kỹ thuật mới sau khi được kẻ tấn công nhận diện.

Các honeypot thích ứng tận dụng các kỹ thuật tiên tiến như học máy và trí tuệ nhân tạo để thích ứng với các chiến thuật được kẻ tấn công sử dụng. Khả năng thích ứng này đảm bảo rằng chúng vẫn hiệu quả ngay cả khi các chiến lược tấn công tiến hóa. Bằng cách liên tục học hỏi và điều chỉnh phản ứng của mình, honeypot thích ứng có thể cung cấp thông tin mới và phù hợp hơn về các mối đe dọa mới nhất. Học tăng cường (RL) là một loại học máy mà các agent học cách đưa ra quyết định bằng cách thực hiện các hành động và nhận phần thưởng hoặc hình phạt. Trong bối cảnh an ninh mạng, RL có thể được sử dụng để phát triển các hệ thống có thể phản ứng thích ứng với các mối đe dọa.

### **2.1.2. Ứng dụng của RL trong an ninh mạng**

Các thuật toán Học Tăng Cường (Reinforcement Learning - RL) có khả năng ứng dụng rộng rãi trong lĩnh vực an ninh mạng, giúp phát triển các hệ thống phản ứng linh hoạt và hiệu quả trước các mối đe dọa mạng. Những hệ thống này học hỏi từ môi trường qua việc tương tác trực tiếp, từ đó đưa ra các quyết định tối ưu nhằm tối đa hóa phần thưởng tích lũy theo thời gian. Trong bối cảnh an ninh mạng, điều này có thể chuyển thành việc phát hiện và giảm thiểu các cuộc tấn công một cách hiệu quả hơn.

#### **2.1.2.1. Các thuật toán chính trong RL:**

##### **Deep Q-Networks (DQN)**

DQN sử dụng mạng nơ-ron để xác định giá trị Q, đại diện cho giá trị kỳ vọng của việc thực hiện một hành động nhất định trong một trạng thái nhất định. Điều này giúp hệ thống học được các chính sách tối ưu để đưa ra các quyết định phản ứng thích hợp với các cuộc tấn công mạng. DQN đặc biệt mạnh mẽ trong việc xử lý các không gian trạng thái lớn và phức tạp, thường gặp trong môi trường mạng.

##### **Double Deep Q-Networks (DDQN)**

DDQN cải thiện DQN bằng cách giải quyết sự thiên vị trong ước lượng giá trị Q. Nó tách rời quá trình chọn hành động khỏi quá trình đánh giá giá trị Q, giúp cho quá

trình học ổn định hơn và cải thiện hiệu suất trong các môi trường động. Điều này đặc biệt quan trọng trong an ninh mạng, nơi các mối đe dọa và tấn công có thể liên tục thay đổi và phát triển.

### **Các phương pháp Policy Gradient**

Các phương pháp này trực tiếp tối ưu hóa chính sách (policy) thay vì giá trị Q, cho phép hệ thống học được các chiến lược tốt hơn trong môi trường có nhiều hành động khả thi. Các phương pháp Policy Gradient thích hợp cho việc xử lý các tình huống phức tạp và đa dạng trong an ninh mạng.

#### **2.1.2.2. Ví dụ về ứng dụng của RL trong an ninh mạng:**

##### **Phát hiện tấn công**

Các hệ thống RL có thể học cách phát hiện các hành vi bất thường hoặc các mẫu tấn công dựa trên lưu lượng mạng và hoạt động hệ thống. Ví dụ, DQN có thể được huấn luyện để phát hiện các cuộc tấn công SQL injection hoặc XSS thông qua việc phân tích các mẫu dữ liệu và phản hồi phù hợp.

##### **Phản ứng tự động**

Sau khi phát hiện tấn công, hệ thống RL có thể tự động thực hiện các biện pháp phản ứng như chặn các yêu cầu tấn công, gửi cảnh báo đến quản trị viên, hoặc áp dụng các chính sách bảo mật nâng cao để ngăn chặn tấn công.

##### **Thích ứng động**

Các thuật toán RL cho phép hệ thống bảo mật thích ứng với các chiến lược tấn công mới và tinh vi hơn. Điều này giúp duy trì hiệu quả bảo mật trong môi trường mạng luôn thay đổi và phức tạp.

#### **2.1.3. Phân tích pháp chứng**

Phân tích pháp chứng trong an ninh mạng liên quan đến việc kiểm tra chi tiết dữ liệu số để hiểu bản chất và phạm vi của một cuộc tấn công. Quá trình này rất quan

trọng để xác định các kỹ thuật và công cụ được kẻ tấn công sử dụng, từ đó có thể định hình các biện pháp phòng thủ.

#### **2.1.3.1. Tầm quan trọng trong an ninh mạng**

Phân tích pháp chứng giúp hiểu rõ các chi tiết của các cuộc tấn công, từ sự xâm nhập ban đầu đến các phương pháp được sử dụng để trích xuất dữ liệu. Sự hiểu biết này rất quan trọng để phát triển các biện pháp bảo mật hiệu quả hơn và để xác định các cuộc tấn công với các agent đe dọa cụ thể.

#### **2.1.3.2. Phương pháp và công cụ**

Các phương pháp chứng truyền thống bao gồm phân tích nhật ký, bắt gói tin và phân tích ngược phần mềm độc hại. Các công cụ pháp chứng hiện đại sử dụng học máy để tự động hóa các quy trình phát hiện và phân tích, giúp xử lý được khối lượng dữ liệu lớn sinh ra trong một cuộc tấn công.

#### **2.1.4. Tích hợp DEEP-Dig**

DEEP-Dig (DEcEPtion DIGging) nâng cao khả năng phân tích pháp chứng bằng cách sử dụng các kỹ thuật đánh lừa để lôi kéo kẻ tấn công tiết lộ chiến thuật của chúng. Phương pháp này thu thập thông tin chi tiết về các vector và chiến lược tấn công, cung cấp một tập dữ liệu phong phú hơn để phân tích. Bằng cách tạo ra các hệ thống mới mà kẻ tấn công tin rằng là các mục tiêu thật, DEEP-Dig thu thập dữ liệu có giá trị về phương pháp và hành vi tấn công. Dữ liệu này sau đó được sử dụng để huấn luyện các mô hình học máy, giúp cải thiện khả năng phát hiện và giảm thiểu các cuộc tấn công trong tương lai. Tiếp theo đó, dữ liệu thu thập thông qua DEEP-Dig được phân tích kỹ lưỡng, xác định các mẫu và bất thường chỉ ra các chiến lược tấn công tinh vi. Phân tích này không chỉ giúp hiểu các mối đe dọa hiện tại mà còn dự đoán các mối đe dọa trong tương lai.

## **2.2. Hướng nghiên cứu hiện tại**

### **2.2.1. Hướng nghiên cứu trong nước**

Trong một nghiên cứu được thực hiện bởi các nhà nghiên cứu tại nhiều trường đại học ở Việt Nam, các hệ thống honeypot truyền thống đã được triển khai để giám sát và phân tích các cuộc tấn công mạng trong các mạng lưới học thuật. Các honeypot này chủ yếu được sử dụng để phát hiện và ghi lại các hoạt động độc hại, cung cấp cái nhìn sâu sắc về các mẫu tấn công phổ biến và hành vi của kẻ tấn công nhắm vào các cơ sở giáo dục. Nghiên cứu này đã nêu bật hiệu quả của các honeypot truyền thống trong việc nâng cao bảo mật mạng bằng cách xác định các lỗ hổng và ngăn chặn các vi phạm tiềm ẩn.

### **2.2.2. Hướng nghiên cứu quốc tế**

Honeypot thích ứng với hệ thống phản hồi tự động tích hợp: Trên toàn cầu, đã có những tiến bộ đáng kể trong việc phát triển honeypot thích ứng có thể thay đổi hành vi dựa trên các mẫu tấn công quan sát được. Các nghiên cứu đã chỉ ra rằng honeypot thích ứng hiệu quả hơn trong việc nắm bắt và phân tích các cuộc tấn công tinh vi.

Một nghiên cứu đáng chú ý, 'Engagement Adaptive Honeypot thông qua Học Tăng cường của quy trình quyết định bán-Markov' [2], khám phá việc sử dụng quy trình quyết định bán-Markov (SMDP) để tối ưu hóa các chiến lược tương tác honeypot. Nghiên cứu này nhấn mạnh lợi ích của việc sử dụng học tăng cường để đưa ra các quyết định thích ứng dựa trên hành vi của kẻ tấn công. Phương pháp này cho phép honeypot điều chỉnh chiến lược tương tác một cách động, từ đó tăng cường hiệu quả trong việc nắm bắt dữ liệu tấn công chi tiết và cải thiện tư thế phòng thủ tổng thể.

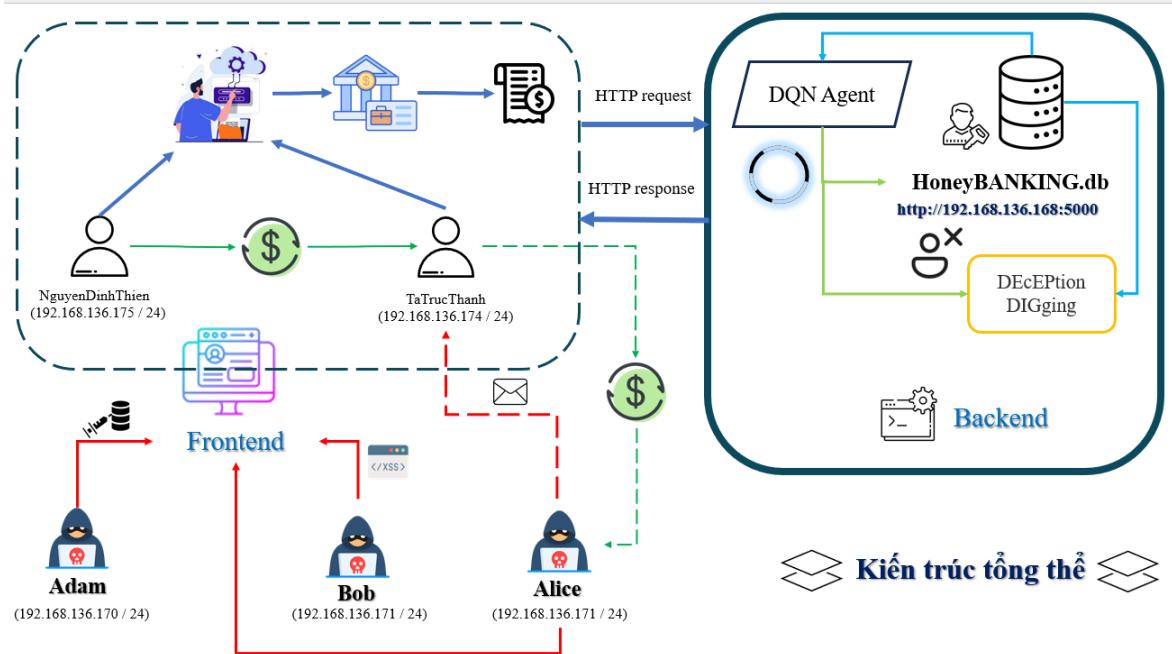
Một nghiên cứu quan trọng khác, "Sử dụng Học Tăng cường để che giấu chức năng honeypot" [3], mô tả cách học tăng cường có thể được áp dụng cho honeypot để làm cho chúng trở nên thích ứng và hiệu quả hơn trong việc tương tác với kẻ tấn công. Bài báo này thảo luận về việc triển khai quy trình quyết định Markov (MDP) và việc sử dụng học tăng cường để kéo dài tương tác của kẻ tấn công và thu thập dữ liệu tấn

công toàn diện hơn. Bằng cách che giấu chức năng của honeypot và làm cho nó xuất hiện như một mục tiêu hợp pháp, các nhà nghiên cứu có thể thu thập những hiểu biết quý giá về phương pháp và chiến thuật của kẻ tấn công, từ đó cải thiện thiết kế và triển khai các honeypot trong tương lai.

## Chương 3. HỆ THỐNG ĐÈ XUẤT

### 3.1. Thiết lập và triển khai hệ thống

#### 3.1.1. Kiến trúc tổng thể



Hình 1. Kiến trúc tổng thể của hệ thống honeypot tích hợp Học Tăng cường.

Nhìn tổng thể kiến trúc của hệ thống honeypot thích ứng được thiết kế để tạo ra một môi trường thực tế và động nhằm phát hiện và phân tích các cuộc tấn công web tiên tiến. Hệ thống bao gồm hai thành phần chính: frontend và backend, hoạt động cùng nhau để thu hút kẻ tấn công và thu thập dữ liệu cho việc phân tích và học hỏi.

#### 3.1.2. Phát triển người dùng

Tạo ra một mô phỏng thực tế của môi trường ngân hàng trực tuyến để thu hút và tương tác với các kẻ tấn công tiềm năng.

**Khung Flask:** Một khung ứng dụng web WSGI nhẹ cho Python để xử lý các yêu cầu web và phục vụ các trang web. Flask được chọn vì sự đơn giản và linh hoạt của nó, làm cho nó lý tưởng để tạo ra các ứng dụng web nguyên mẫu nhanh chóng.

**HTML, CSS và JavaScript:** Các công nghệ này được sử dụng để tạo giao diện người

dùng đáp ứng và tương tác. HTML cung cấp cấu trúc, CSS xử lý việc tạo kiểu và JavaScript thêm tính tương tác cho các trang web.

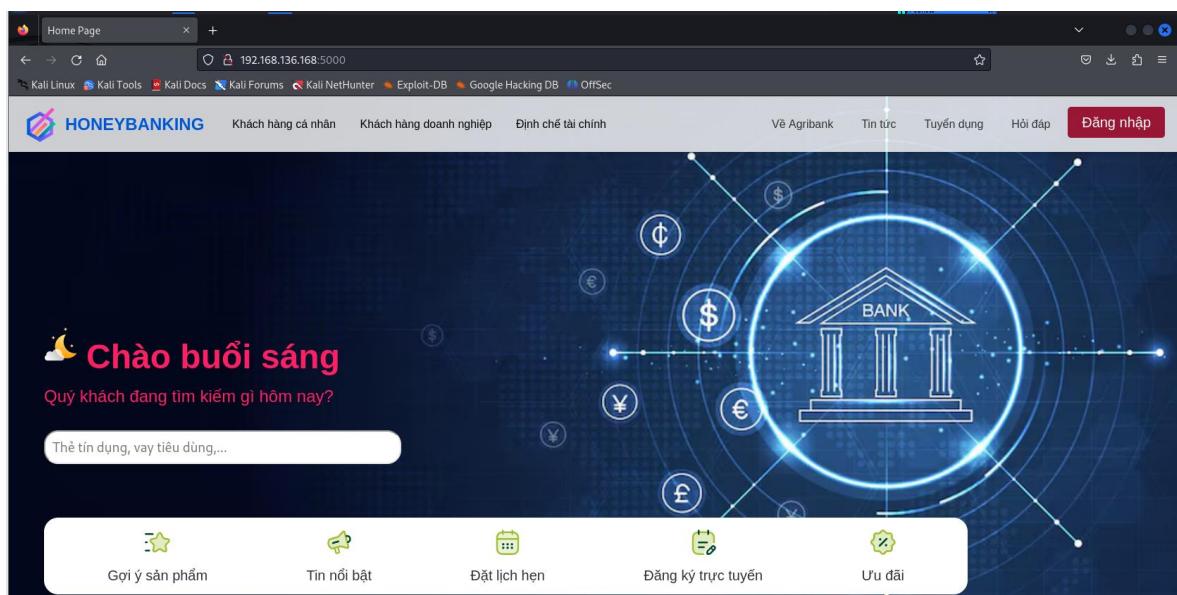
### 3.1.2.1. Các thành phần và tính năng

#### Phát triển Frontend

Giao diện của hệ thống honeypot thích ứng được thiết kế để mô phỏng một máy chủ ngân hàng trực tuyến thời gian thực nhằm thu hút kẻ tấn công. Sử dụng Flask, một khung web nhẹ, giao diện cung cấp một giao diện thực tế mô phỏng các chức năng của một hệ thống ngân hàng thực sự. Điều này bao gồm các tính năng như đăng nhập người dùng, tổng quan tài khoản, chuyển tiền và lịch sử giao dịch. Bằng cách tạo ra một môi trường đáng tin cậy, giao diện khuyến khích kẻ tấn công tương tác với hệ thống, do đó cung cấp dữ liệu quý giá cho việc phân tích.

#### Giao diện chính

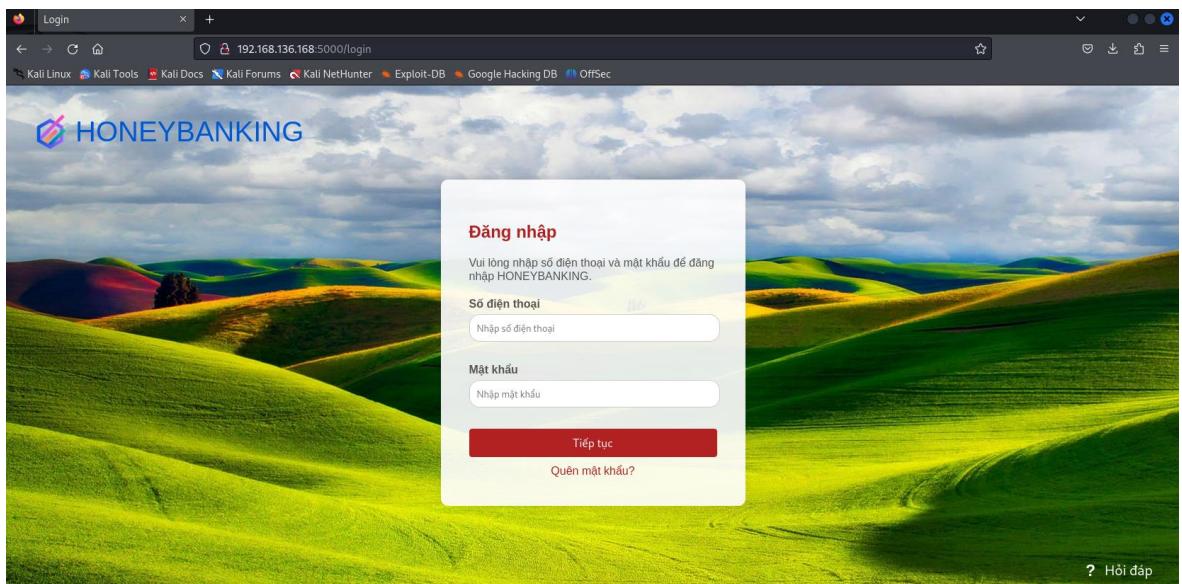
Mô phỏng giả lập một giao diện chính của một ngân hàng trực tuyến.



Hình 2. Giao diện trang chủ của ứng dụng web ngân hàng

## Trang đăng nhập người dùng

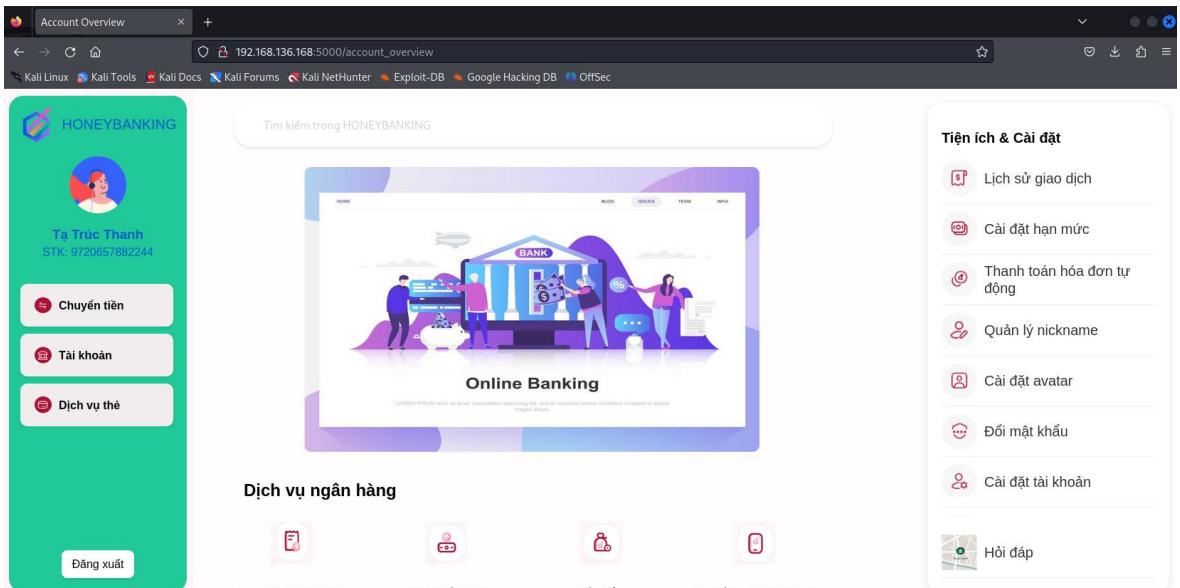
Một giao diện nơi người dùng (kẻ tấn công) có thể đăng nhập bằng thông tin đăng nhập. Điều này mô phỏng trang đăng nhập ngân hàng thực để thu hút kẻ tấn công thử đánh cắp thông tin đăng nhập hoặc SQL injection.



Hình 3. Giao diện người dùng đăng nhập để truy cập vào tài khoản.

## Tổng Quan Tài Khoản

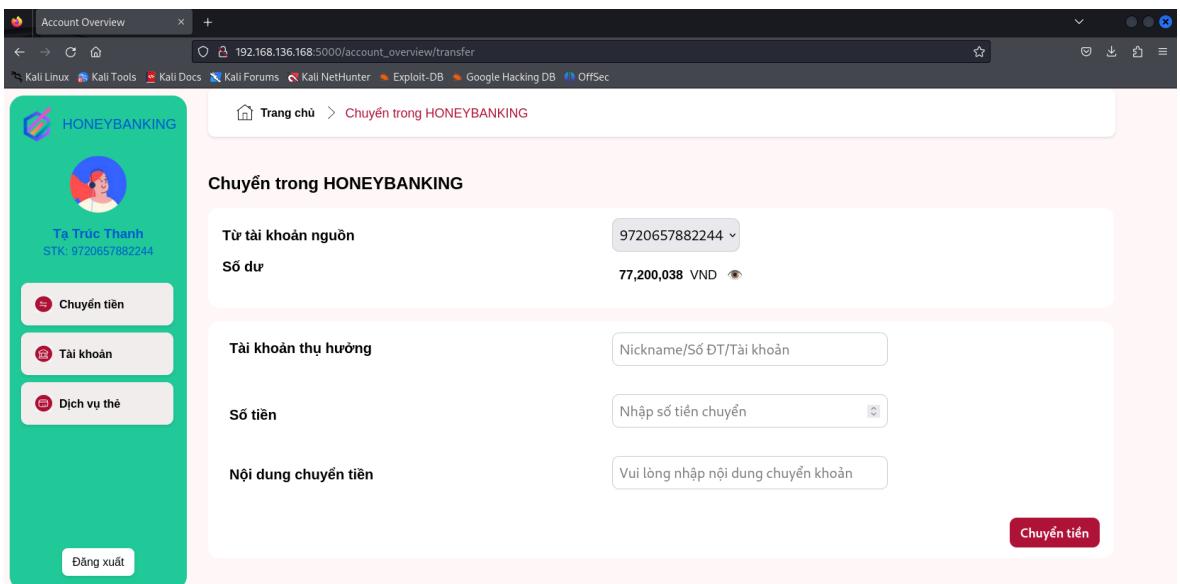
Một bảng điều khiển hiển thị số dư tài khoản và các giao dịch gần đây. Tính năng này cung cấp một mục tiêu thực tế cho kẻ tấn công nhắm đến việc thao tác số dư tài khoản hoặc xem thông tin nhạy cảm.



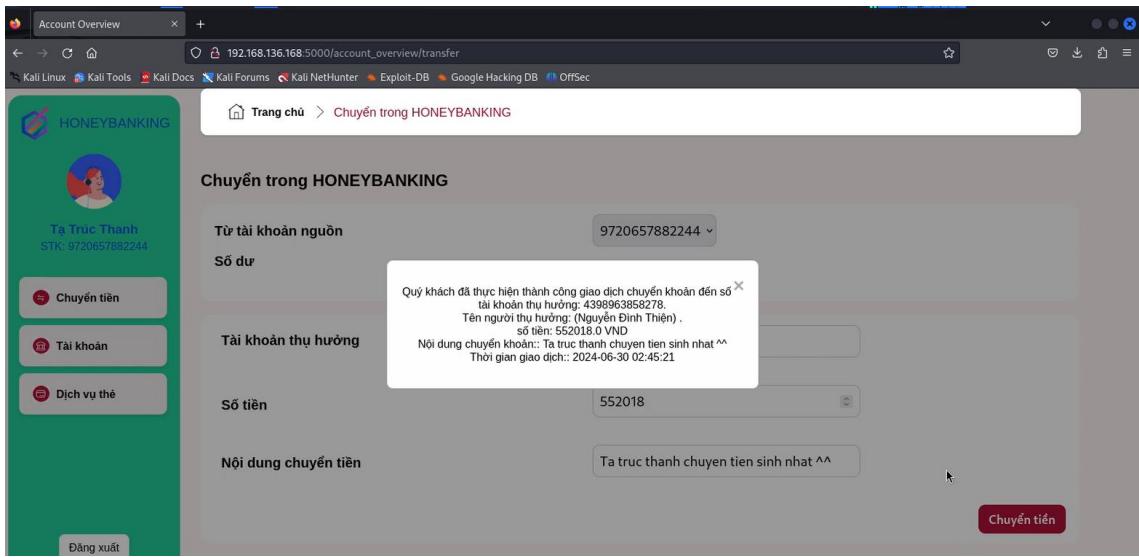
Hình 4. Giao diện tài khoản tổng quan của người dùng.

## Trang chuyển tiền của người dùng

Chức năng mô phỏng việc chuyển tiền giữa các tài khoản.



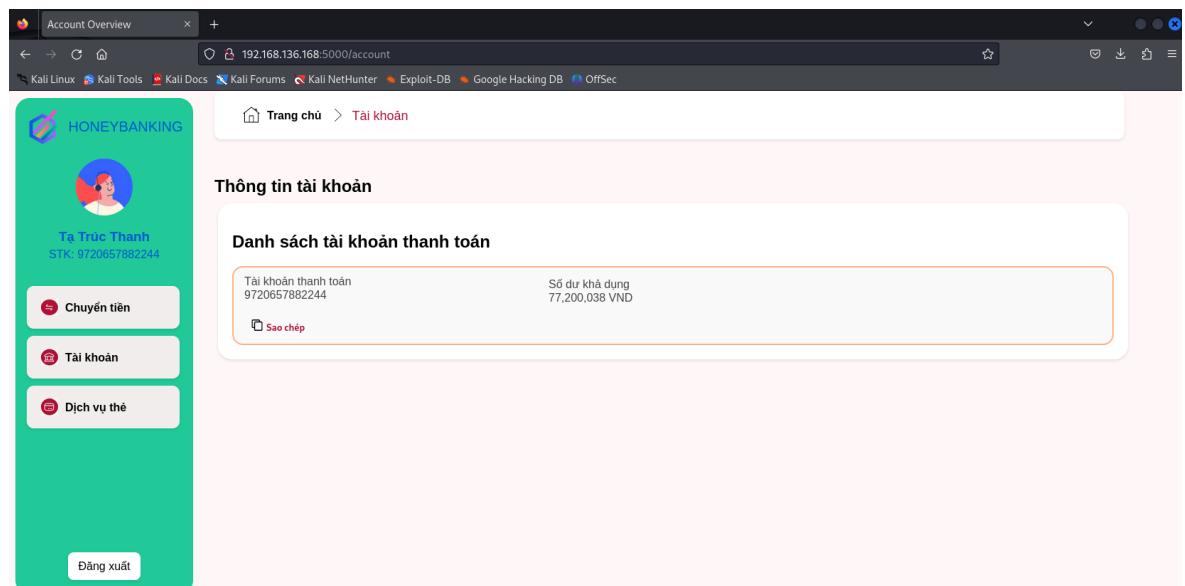
Hình 5. Giao diện chuyển khoản nơi người dùng có thể nhập vào tài khoản thụ hưởng với số tiền tương ứng cần chuyển cho người dùng muốn nhận.



Hình 6. Khi người dùng chuyển tiền thành công sẽ có thông báo xác nhận và khi người dùng tắt nó thì hệ thống sẽ tự load lại trang để cập nhật lại số dư khả dụng của người dùng sau khi chuyển khoản.

## Trang thông tin tài khoản của người dùng

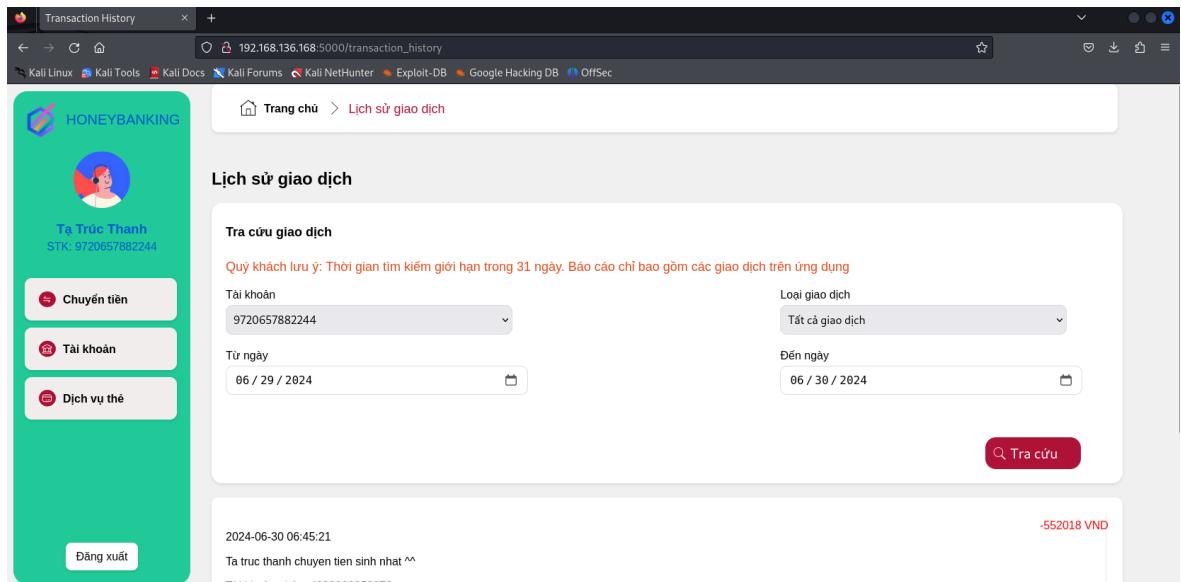
Chi tiết về hồ sơ người dùng và cài đặt tài khoản. Trang này được thiết kế để trông giống như thật, khuyến khích kẻ tấn công thử tấn công XSS hoặc CSRF.



Hình 7. Giao diện thông tin tài khoản của người dùng.

## Giao diện lịch sử giao dịch

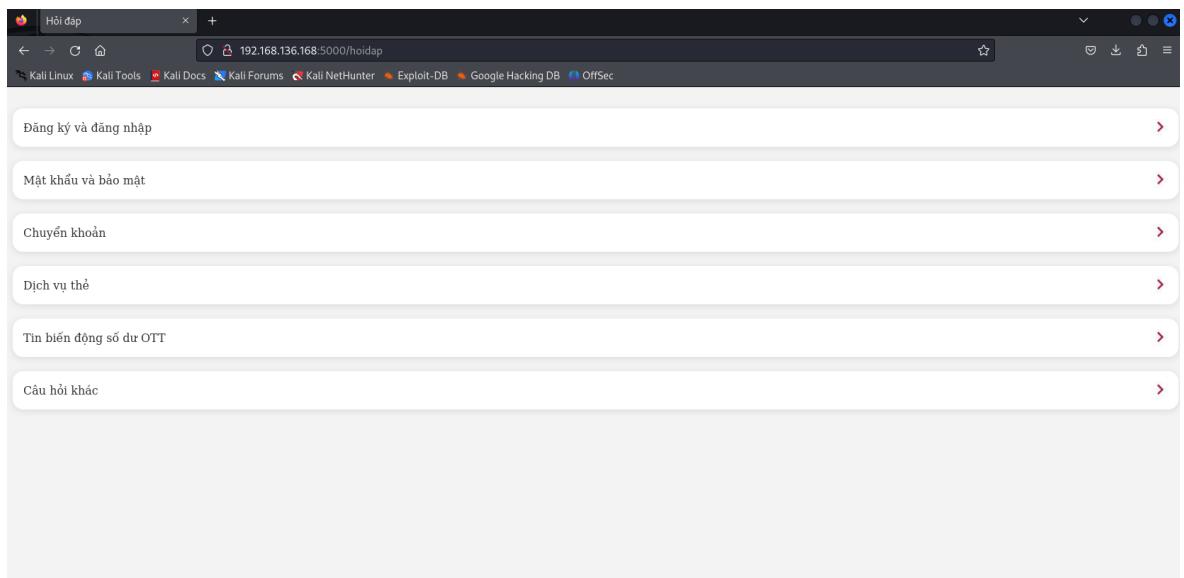
Một nhật ký của tất cả các giao dịch đã thực hiện bởi người dùng. Tính năng này cho phép kẻ tấn công thử sửa đổi các bản ghi giao dịch, cung cấp dữ liệu có giá trị cho phân tích.



Hình 8. Giao diện lịch sử giao dịch, nơi người dùng có thể tra cứu lại những giao dịch đã thực hiện được trong thời gian vừa qua.

## Tính năng khác

Ngoài ra người dùng có thể nhấn vào nút Hỏi đáp nếu cần giải đáp một thắc mắc nào đó liên quan.



Hình 9. Giao diện phần hỏi đáp cho người dùng.

### 3.1.2.2. Chi tiết triển khai

Giao diện người dùng mô phỏng các giao diện ngân hàng thực để lôi kéo tấn công trong khi ghi lại các tương tác của họ để phân tích. Mỗi tương tác với ứng dụng web được ghi lại cẩn thận để nghiên cứu hành vi và kỹ thuật được sử dụng bởi kẻ tấn công. Nó sử dụng các lỗ hổng được kiểm soát nhưng an toàn để nghiên cứu các mẫu tấn công mà không làm ảnh hưởng đến an ninh thực sự. Cách tiếp cận này đảm bảo rằng trong khi hệ thống có vẻ dễ bị tấn công, rủi ro thực sự là tối thiểu.

## Phát triển Backend

Xử lý các yêu cầu, phân tích lưu lượng và ghi nhật ký tương tác.

### Công cụ và Khung được sử dụng:

#### Flask và SQLAlchemy

Flask xử lý các yêu cầu và phản hồi HTTP, trong khi SQLAlchemy quản lý các hoạt động cơ sở dữ liệu, cung cấp một lớp ORM (Object-Relational Mapping) để tương tác với cơ sở dữ liệu SQLite.

## **SQLite**

Một cơ sở dữ liệu nhẹ, dựa trên đĩa không yêu cầu quá trình máy chủ riêng biệt, làm cho nó lý tưởng để ghi nhật ký và phân tích dữ liệu tấn công trong môi trường nguyên mẫu.

## **DQN Agent**

Hệ thống honeypot thích ứng tận dụng Mạng Q-Sâu (DQN) để phát hiện và phản hồi các cuộc tấn công web, cụ thể là tập trung vào các cuộc tấn công SQL Injection.

### **3.1.3. Các thành phần của hệ thống Honeypot thích ứng**

#### **3.1.3.1. Thu thập dữ liệu (Data collection)**

Thành phần thu thập dữ liệu chịu trách nhiệm ghi lại tất cả thông tin liên quan trong quá trình tương tác giữa kẻ tấn công và honeypot. Điều này bao gồm:

#### **Gói tin mạng**

Thông tin chi tiết về dữ liệu được truyền qua mạng, cung cấp cái nhìn sâu sắc về các phương pháp và công cụ mà kẻ tấn công sử dụng. Điều này cho phép phân tích chi tiết về cách thức tấn công và các bước tiến hành của kẻ tấn công

#### **Yêu cầu và phản hồi HTTP**

Toàn diện về tất cả lưu lượng web, bao gồm tính chất của các yêu cầu được gửi bởi kẻ tấn công và phản hồi từ máy chủ honeypot. Điều này bao gồm các thông tin như phương thức HTTP (GET, POST), đường dẫn yêu cầu, các thông số yêu cầu, và nội dung phản hồi. Phân tích yêu cầu và phản hồi HTTP giúp xác định các điểm yếu trong ứng dụng web và cách kẻ tấn công khai thác chúng. Nó cũng cho phép xác định các mẫu tấn công và hành vi của kẻ tấn công trên ứng dụng web.

#### **Nhật ký hệ thống**

Ghi lại nhật ký được tạo ra bởi máy chủ và hệ thống honeypot. Nhật ký này bao gồm thông tin về hoạt động của hệ thống, các yêu cầu đăng nhập, thay đổi cấu hình,

và bất kỳ sự bất thường nào được phát hiện. Nhật ký hệ thống cung cấp bằng chứng về hoạt động của kẻ tấn công và giúp xác định các hành vi bất thường hoặc các nỗ lực xâm nhập vào hệ thống. Việc phân tích nhật ký hệ thống là cần thiết để phát hiện và ngăn chặn các cuộc tấn công.

### **3.1.3.2. Tiền xử lý dữ liệu (Data Processing)**

Tiền xử lý dữ liệu là điều cần thiết để đảm bảo rằng dữ liệu được thu thập phù hợp cho việc phân tích. Các bước tiền xử lý bao gồm:

#### **Chuẩn hóa**

Chuẩn hóa dữ liệu để đảm bảo tính nhất quán, điều này rất quan trọng để các mô hình học máy hoạt động chính xác. Chuẩn hóa giúp giảm sự sai lệch và biến đổi không cần thiết trong dữ liệu, làm tăng độ chính xác của mô hình học máy.

#### **Lọc**

Loại bỏ thông tin không liên quan hoặc dư thừa để tập trung vào các điểm dữ liệu quan trọng nhất. Lọc dữ liệu giúp giảm thiểu khối lượng dữ liệu cần xử lý, đồng thời tăng tính chính xác và hiệu quả của quá trình phân tích.

#### **Cấu trúc**

Tổ chức dữ liệu thành một định dạng có thể dễ dàng được xử lý bởi các công cụ phân tích và thuật toán học máy. Cấu trúc hóa dữ liệu giúp dễ dàng quản lý, truy vấn và phân tích dữ liệu. Điều này đặc biệt quan trọng đối với các mô hình học máy và các công cụ phân tích dữ liệu lớn.

### **3.1.3.3. Phân tích vector tấn công (Attack vector analysis)**

Thành phần này sử dụng các kỹ thuật học máy và khai thác dữ liệu tiên tiến để phân tích dữ liệu đã được xử lý và xác định các mẫu có thể chỉ ra một cuộc tấn công. Các hoạt động chính bao gồm:

#### **Nhận dạng mẫu**

Xác định các đặc điểm chung của các vector tấn công đã biết để phát hiện các mối đe dọa tương tự. Việc nhận dạng mẫu giúp hệ thống phát hiện các cuộc tấn công mạng nhanh chóng và chính xác, ngay cả khi các cuộc tấn công này đã được biến đổi nhẹ để tránh các biện pháp phát hiện truyền thống.

### **Phát hiện bất thường**

Quá trình nhận ra các sai lệch so với hành vi bình thường có thể biểu thị một phương pháp tấn công mới hoặc tinh vi. Phương pháp này không yêu cầu dữ liệu tấn công đã biết trước mà dựa trên việc học các mẫu hành vi bình thường và phát hiện các bất thường.

### **Phân tích mối quan hệ**

Khám phá các mối quan hệ ẩn giữa các điểm dữ liệu để hiểu các chiến lược tấn công phức tạp. Phân tích mối quan hệ cung cấp cái nhìn sâu sắc về cách các cuộc tấn công được phối hợp và thực hiện.

#### **3.1.3.4. Báo cáo (Reporting)**

Là thành phần báo cáo tạo ra các báo cáo chi tiết dựa trên phân tích, cung cấp những thông tin có giá trị cho các quản trị viên hệ thống. Các báo cáo này thường bao gồm:

### **Phương pháp tấn công**

Mô tả các kỹ thuật được sử dụng bởi kẻ tấn công, cho phép quản trị viên hiểu rõ hơn về các mối đe dọa.

### **Lỗi hỏng hệ thống**

Xác định các điểm yếu trong hệ thống đã bị kẻ tấn công khai thác.

## **3.2. Tổng quan về mô hình DQN**

Mô hình DQN (Deep Q-Network) là một kỹ thuật Học tăng cường sâu (Deep Reinforcement Learning), trong đó mạng nơ-ron nhân tạo được sử dụng để xấp xỉ hàm Q, giúp tối ưu hóa chiến lược hành động của agent trong môi trường học tập.

DQN kết hợp giữa phương pháp Q-Learning và mạng nơ-ron sâu để giải quyết các vấn đề phức tạp hơn so với các phương pháp học tăng cường truyền thống.

### **Các thành phần chính của mô hình DQN bao gồm:**

#### **3.2.1. Hàm Q**

Hàm  $Q(x, a)$  trả về giá trị  $Q$  cho trạng thái  $x$  và hành động  $a$ , giúp đánh giá giá trị của hành động trong một trạng thái cụ thể.

#### **3.2.2. Mạng nơ-ron sâu**

Dùng để xấp xỉ hàm  $Q$ , với các lớp input, hidden, và output.

#### **3.2.3. Replay memory**

Bộ nhớ lưu trữ các trải nghiệm (state, action, reward, next\_state, done) để huấn luyện mô hình một cách hiệu quả.

#### **3.2.4. Target network**

Một bản sao của mạng chính, dùng để ổn định quá trình huấn luyện bằng cách cập nhật trọng số định kỳ.

### **3.3. Mô tả chi tiết DQN Agent**

#### **3.3.1. Class SumTree**

Quản lý cây nhị phân để lưu trữ và truy xuất các trải nghiệm dựa trên ưu tiên. Thành phần chính bao gồm:

##### **Cấu trúc cây**

Cấu trúc của SumTree là một mảng nhị phân, được sử dụng để lưu trữ các giá trị ưu tiên. Mảng nhị phân này giúp quản lý các trải nghiệm một cách hiệu quả bằng cách sắp xếp và tổ chức chúng theo mức ưu tiên. Việc sử dụng cấu trúc cây nhị phân cho phép các thao tác thêm, cập nhật và truy xuất được thực hiện nhanh chóng và chính xác.

##### **Phương thức add**

Phương thức add chịu trách nhiệm thêm một trải nghiệm mới vào cây với mức ưu tiên xác định. Khi một trải nghiệm mới được thêm vào, cây sẽ tự động điều chỉnh để duy trì cấu trúc nhị phân và cập nhật các giá trị ưu tiên tương ứng. Điều này đảm bảo rằng các trải nghiệm có mức ưu tiên cao hơn sẽ có khả năng được chọn nhiều hơn trong các thao tác truy xuất sau này.

### **Phương thức update**

Phương thức update cho phép cập nhật giá trị ưu tiên cho một nút cụ thể trong cây. Khi giá trị ưu tiên của một nút thay đổi, cây sẽ điều chỉnh lại các giá trị trong mảng nhị phân để phản ánh sự thay đổi này. Phương thức này rất quan trọng trong việc duy trì tính nhất quán và chính xác của các giá trị ưu tiên trong suốt quá trình học tập của mô hình.

### **Phương thức get\_leaf**

Phương thức get\_leaf được sử dụng để truy xuất một trải nghiệm dựa trên giá trị ưu tiên ngẫu nhiên. Khi phương thức này được gọi, nó sẽ chọn một nút trong cây dựa trên giá trị ưu tiên và trả về trải nghiệm tương ứng. Điều này giúp đảm bảo rằng các trải nghiệm có mức ưu tiên cao hơn sẽ có cơ hội được chọn nhiều hơn, góp phần nâng cao hiệu quả học tập của mô hình.

#### **3.3.2. Class Memory**

Quản lý bộ nhớ trải nghiệm, sử dụng cấu trúc SumTree để lưu trữ các trải nghiệm với ưu tiên. Thành phần chính bao gồm:

### **Phương thức store**

Phương thức store chịu trách nhiệm lưu trữ các trải nghiệm mới vào bộ nhớ. Khi một trải nghiệm mới được ghi nhận, phương thức này sẽ thêm trải nghiệm đó vào cấu trúc SumTree với mức ưu tiên xác định. Điều này đảm bảo rằng các trải nghiệm mới được đưa vào bộ nhớ một cách có tổ chức và có thể truy xuất hiệu quả sau này. Phương thức này đóng vai trò quan trọng trong việc thu thập dữ liệu trải nghiệm liên tục từ môi trường để cải thiện quá trình học tập của mô hình.

### **Phương thức sample**

Phương thức sample được sử dụng để lấy mẫu ngẫu nhiên từ bộ nhớ để huấn luyện mô hình. Phương thức này sử dụng cấu trúc SumTree để chọn các trải nghiệm dựa trên giá trị ưu tiên, đảm bảo rằng các trải nghiệm có mức ưu tiên cao hơn có xác suất được chọn lớn hơn. Phương thức rất quan trọng để tạo ra các lô dữ liệu huấn luyện đa dạng và đại diện cho các tình huống khác nhau mà mô hình có thể gặp phải.

### **Phương thức batch\_update**

Phương thức batch\_update cho phép cập nhật các ưu tiên của các trải nghiệm trong bộ nhớ. Khi các trải nghiệm được sử dụng trong quá trình huấn luyện, giá trị ưu tiên của chúng có thể thay đổi dựa trên hiệu quả của chúng trong việc cải thiện mô hình. Phương thức này cho phép điều chỉnh lại các ưu tiên trong SumTree, đảm bảo rằng các trải nghiệm quan trọng hơn sẽ có cơ hội được chọn nhiều hơn trong tương lai. Điều này giúp tăng cường hiệu quả học tập của mô hình bằng cách tập trung vào các trải nghiệm có giá trị nhất.

#### **3.3.3. Class DQN\_Agent**

Đại diện cho agent DQN, quản lý quá trình huấn luyện và ra quyết định. Thành phần chính bao gồm:

##### **Phương thức \_\_init\_\_**

Khởi tạo agent với các tham số như kích thước trạng thái, kích thước hành động, tỷ lệ giảm giá, và thiết lập mô hình mạng nơ-ron.

##### **Phương thức \_build\_model**

Xây dựng mạng nơ-ron với các lớp Dense và Dropout.

##### **Phương thức act**

Chọn hành động dựa trên trạng thái hiện tại và epsilon-greedy.

##### **Phương thức remember**

Lưu trữ trải nghiệm vào bộ nhớ.

### **Phương thức replay**

Huấn luyện mô hình từ các mẫu ngẫu nhiên trong bộ nhớ.

### **Phương thức update\_target\_model**

Cập nhật trọng số của mạng mục tiêu.

### **Phương thức save và load**

Lưu và tải trọng số của mô hình.

### **Phương thức calculate\_metric**

Tính toán các chỉ số hiệu suất như Accuracy, Precision, Recall và F1 Score.

### **Phương thức plot\_metrics**

Vẽ biểu đồ các chỉ số hiệu suất để đánh giá mô hình.

#### **3.3.3.1. Các thành phần của DQN Agent**

##### **Mạng chính sách (Policy network)**

Chịu trách nhiệm dự đoán hành động tốt nhất dựa trên trạng thái hiện tại của hệ thống.

**Nguyên lý hoạt động:** Sử dụng các lớp Dense và Dropout để xử lý dữ liệu đầu vào và dự đoán hành động. Việc sử dụng Dropout giúp tránh overfitting trong quá trình huấn luyện.

##### **Mạng mục tiêu (Target network)**

Cung cấp các giá trị mục tiêu ổn định cho các cập nhật Q-learning.

**Nguyên lý hoạt động:** Được đồng bộ định kỳ với mạng chính sách để đảm bảo các giá trị mục tiêu nhất quán. Cách tiếp cận này giúp ổn định quá trình học bằng cách giảm nguy cơ phân kỳ trong quá trình huấn luyện.

### 3.3.3.2. Chi tiết triển khai

#### Biểu diễn trạng thái (State representation)

Ghi lại trạng thái hiện tại của hệ thống bao gồm dữ liệu lưu lượng mạng, hành động của người dùng và phản hồi của hệ thống. Trạng thái được định nghĩa là một vector đại diện cho các đặc trưng liên quan đến quá trình đăng nhập:

```
state = np.array([
    username.isdigit(),
    6 <= len(password) <= 18,
    User.query.filter_by(username=username).first() is not None,
    User.query.filter_by(username=username, password=password).first() is not None,
    has_sql_injection,
    has_xss,
    False # CSRF detection not applicable
], dtype=int).reshape(1, -1)
```

Hình 10. Biểu diễn trạng thái của DQN Agent

#### ○ Tên đăng nhập chỉ gồm chữ số

Chỉ ra liệu tên đăng nhập có chỉ bao gồm chữ số hay không.

#### ○ Độ dài mật khẩu

Kiểm tra xem độ dài mật khẩu có nằm trong khoảng từ 6 đến 18 ký tự hay không.

#### ○ Tên đăng nhập tồn tại

Xác nhận xem tên đăng nhập có tồn tại trong cơ sở dữ liệu hay không.

#### ○ Đăng nhập hợp lệ

Kiểm tra xem cả tên đăng nhập và mật khẩu có đúng không.

#### ○ Phát hiện SQL Injection

Dánh dấu nếu phát hiện tấn công SQL Injection.

## ○ Phát hiện XSS

Đánh dấu nếu phát hiện tấn công XSS.

## ○ Phát hiện CSRF (False)

Hiện tại không áp dụng nhưng sử dụng trong việc phát hiện tấn công CSRF nếu có thể.

**Thực hiện:** Trạng thái được chuẩn hóa và biểu diễn dưới dạng vector để làm đầu vào cho mạng nơ-ron. Biểu diễn có cấu trúc này cho phép mô hình đánh giá chính xác trạng thái hiện tại của các nỗ lực đăng nhập và các đặc trưng liên quan.

## Chọn hành động (Action selection)

Chọn hành động dựa trên trạng thái hiện tại bằng cách sử dụng chiến lược epsilon-greedy. Không gian hành động bao gồm các hành động rời rạc mà honeypot có thể thực hiện đáp ứng các yêu cầu web:

### Cảnh báo

Đánh dấu yêu cầu là có khả năng độc hại và ghi lại chi tiết.

### Bỏ qua

Xem yêu cầu là an toàn và cho phép nó tiếp tục mà không can thiệp.

Mô hình DQN chọn hành động dựa trên trạng thái hiện tại để tối ưu hóa việc phát hiện các cuộc tấn công SQL Injection, nhằm mục đích giảm thiểu các dương tính giả và tiêu cực và cải thiện độ chính xác toàn cầu.

**Thực hiện:** Agent chọn một hành động ngẫu nhiên với xác suất epsilon và hành động được đề xuất bởi mạng chính sách với xác suất 1-epsilon.

## Phản hồi môi trường (Environment response)

Thực thi hành động và nhận phản hồi từ môi trường dưới dạng phần thưởng và trạng thái tiếp theo.

**Thực hiện:** Hệ thống cập nhật trạng thái và tính toán phần thưởng dựa trên hiệu quả của hành động được thực thi.

### Lưu trữ kinh nghiệm (Experience storage)

Lưu trữ bộ kinh nghiệm (state, action, reward, next\_state, done) trong bộ đệm phát lại kinh nghiệm.

**Thực hiện:** Agent lưu trữ các kinh nghiệm vào một bộ đệm phát lại để sử dụng trong các bước huấn luyện sau này.

### Cập nhật chính sách (Policy update)

Cập nhật mạng chính sách bằng cách sử dụng gradient descent ngẫu nhiên để giảm thiểu lỗi khác biệt thời gian giữa phần thưởng dự đoán và phần thưởng thực tế.

**Thực hiện:** Agent sử dụng kỹ thuật Gradient Descent để cập nhật các trọng số của mạng nơ-ron, giảm thiểu sự chênh lệch giữa các giá trị Q dự đoán và giá trị Q mục tiêu.

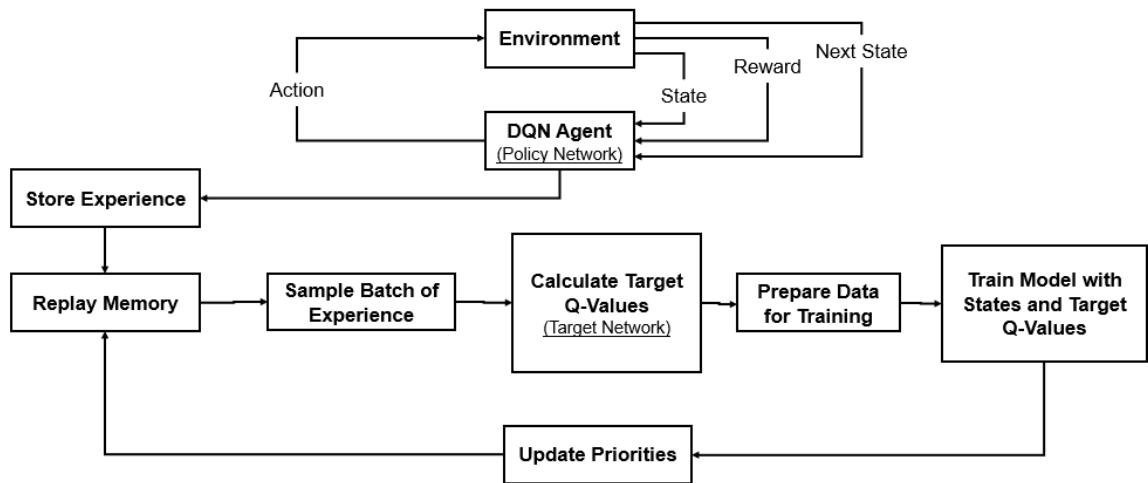
### Hàm thưởng (Reward Function)

Thiết kế hàm thưởng để khuyến khích việc phát hiện chính xác và phản ứng phù hợp với các cuộc tấn công.

**Phần thưởng tích cực:** Nhận diện và giảm thiểu chính xác một cuộc tấn công có thể mang lại phần thưởng tích cực cao.

**Phần thưởng tiêu cực:** Không phát hiện được một cuộc tấn công hoặc tạo ra các dương tính giả có thể dẫn đến phần thưởng tiêu cực.

## Quá trình huấn luyện (Training process)



Hình 11. Quá trình huấn luyện DQN Agent được huấn luyện bằng cách sử dụng phát lại trải nghiệm ưu tiên

### Tương tác với môi trường (Environment):

#### Môi trường

Đại diện cho bối cảnh hoạt động nơi DQN agent tương tác. Nó cung cấp trạng thái hiện tại, phản thưởng và trạng thái tiếp theo dựa trên hành động của agent.

#### DQN Agent (Mạng chính sách)

DQN Agent sẽ quyết định hành động nào sẽ được thực hiện dựa trên trạng thái hiện tại và dự đoán của mạng chính sách. Mạng chính sách chịu trách nhiệm chọn hành động tốt nhất bằng cách xử lý dữ liệu đầu vào qua các lớp mạng nơ-ron của nó.

#### Lưu trữ trải nghiệm (Store Experience):

Khi DQN agent thực hiện hành động và nhận phản hồi từ môi trường (state, action, reward, next\_state), nó lưu trữ trải nghiệm này trong bộ nhớ phát lại để huấn luyện trong tương lai.

## **Replay Memory**

Một cấu trúc dữ liệu lưu trữ các trải nghiệm (state, action, reward, next\_state, done) sử dụng cơ chế phát lại trải nghiệm ưu tiên. Cấu trúc SumTree giúp lưu trữ và truy xuất các trải nghiệm một cách hiệu quả dựa trên ưu tiên.

## **Lấy mẫu và chuẩn bị huấn luyện (Sampling and Training Preparation):**

### **Sample Batch of Experience**

Định kỳ, DQN agent sẽ tiến hành lấy mẫu một lô trải nghiệm từ bộ nhớ phát lại để huấn luyện mạng nơ-ron.

### **Tính toán giá trị Q-mục tiêu (Mạng mục tiêu)**

Mạng mục tiêu tính toán các giá trị Q cho các trạng thái tiếp theo trong các trải nghiệm được lấy mẫu. Các giá trị Q-mục tiêu này được sử dụng để tính toán hàm mất mát trong quá trình huấn luyện, giúp ổn định quá trình huấn luyện.

### **Prepare Data for Training**

Các trải nghiệm được lấy mẫu và các giá trị Q-mục tiêu được tính toán được chuẩn bị cho quá trình huấn luyện. Điều này bao gồm việc cấu trúc dữ liệu theo định dạng phù hợp để huấn luyện mạng nơ-ron.

## **Huấn luyện và cập nhật mô hình (Model Training and Updates):**

### **Train Model with States and Target Q-Values**

Dữ liệu đã chuẩn bị được sử dụng để huấn luyện mạng chính sách. Quá trình huấn luyện bao gồm việc cập nhật trọng số mạng để giảm thiểu sự chênh lệch giữa các giá trị Q dự đoán và các giá trị Q-mục tiêu. Mạng nơ-ron được huấn luyện trên lô các trải nghiệm đã được lấy mẫu.

### **Cập nhật ưu tiên trong bộ nhớ phát lại**

Sau khi huấn luyện, các ưu tiên của các trải nghiệm được lấy mẫu trong bộ nhớ phát lại được cập nhật. Điều này đảm bảo rằng các trải nghiệm quan trọng hơn có xác suất cao hơn được lấy mẫu trong tương lai, nâng cao hiệu quả học tập.

### **3.3.3.3. Đánh giá hiệu suất của agent qua các chỉ số Accuracy, Precision, Recall và F1 Score**

Trong quá trình huấn luyện và đánh giá, các chỉ số hiệu suất của agent được tính toán để đo lường khả năng phát hiện và phản ứng với các cuộc tấn công. Các chỉ số này bao gồm:

#### **Accuracy**

Đo lường tỷ lệ dự đoán đúng trên tổng số dự đoán. Độ chính xác cho thấy tần suất mô hình xác định đúng cả các cuộc tấn công và không tấn công.

#### **Precision**

Đo lường tỷ lệ dự đoán đúng trong số các dự đoán là tấn công.

#### **Recall**

Đo lường tỷ lệ các dự đoán dương đúng trên tổng số các trường hợp dương thực tế (bao gồm cả đúng dương và sai âm). Độ nhạy cao chỉ ra rằng mô hình thành công trong việc phát hiện hầu hết các cuộc tấn công thực tế.

#### **F1 Score**

Trung bình điều hòa của độ chính xác và độ nhạy, cung cấp sự cân bằng giữa hai chỉ số này. Điểm F1 đặc biệt hữu ích khi cần cân bằng giữa độ chính xác và độ nhạy. Điểm F1 cao chỉ ra rằng mô hình hiệu quả trong việc phát hiện các cuộc tấn công (độ nhạy cao) và thực hiện các dự đoán chính xác (độ chính xác cao).

**Trong dqn\_agent, các chỉ số này được tính toán như sau:**

**accuracy** = accuracy\_score(y\_true, y\_pred)

**precision** = precision\_score(y\_true, y\_pred, zero\_division=1)

**recall** = recall\_score(y\_true, y\_pred, zero\_division=1)

```
f1 = f1_score(y_true, y_pred, zero_division=1)
```

### **Giải thích chi tiết:**

#### **accuracy\_score**

Hàm này tính toán độ chính xác bằng cách so sánh các nhãn dự đoán (`y_pred`) với các nhãn thực tế (`y_true`).

#### **precision\_score**

Hàm này tính toán độ chính xác dựa trên các dự đoán đúng trong số các dự đoán là tấn công. Tham số `zero_division=1` đảm bảo không có lỗi chia cho 0.

#### **recall\_score**

Hàm này tính toán tỷ lệ phát hiện đúng trong số các cuộc tấn công thực tế. Tham số `zero_division=1` đảm bảo không có lỗi chia cho 0.

#### **f1\_score**

Hàm này tính toán F1 Score, trung bình điều hòa của Precision và Recall.

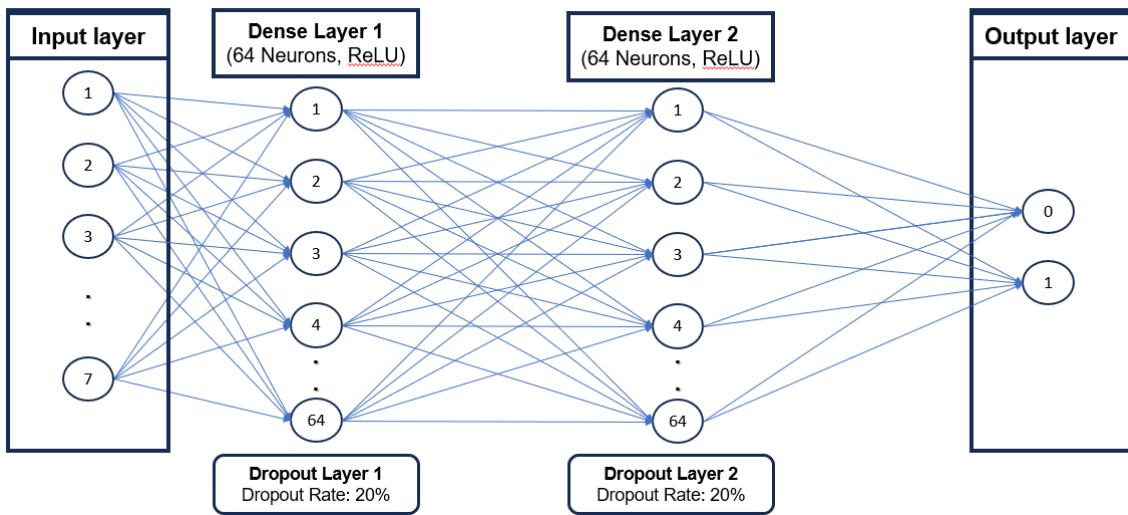
**Các chỉ số này được ghi lại trong log để theo dõi quá trình huấn luyện và đánh giá hiệu suất của agent:**

```
logging.info(f"Accuracy: {accuracy}, Precision: {precision}, Recall: {recall}, F1 Score: {f1}")
```

## **3.4. Hệ thống đề xuất**

Hệ thống honeypot được xây dựng với mục tiêu phát hiện và phản ứng với các cuộc tấn công web như SQL Injection, XSS và CSRF. Hệ thống sử dụng DQN Agent để học và tối ưu hóa chiến lược phát hiện tấn công, từ đó cải thiện hiệu quả bảo mật.

### 3.4.1. Kiến trúc mô hình DQN Agent



Hình 12. Kiến trúc mô hình DQN Agent

#### Input layer

Tầng đầu vào của DQN Agent có kích thước trạng thái là 7 đặc trưng. Các đặc trưng này bao gồm các chỉ số như username, password và các dấu hiệu nhận biết tấn công. Điều này đảm bảo rằng tất cả các thông tin liên quan đến trạng thái hiện tại của hệ thống được đưa vào để phân tích.

#### Hidden layer

Mô hình có hai lớp ẩn Dense, mỗi lớp bao gồm 64 neurons. Các lớp này sử dụng hàm kích hoạt ReLU (Rectified Linear Unit) để tạo ra các tín hiệu kích hoạt phi tuyến. Để giảm thiểu overfitting, Dropout với tỷ lệ 20% được áp dụng sau mỗi lớp ẩn. Điều này giúp mô hình học được các đặc trưng quan trọng và giảm thiểu nhiễu.

#### Output layer

Tầng đầu ra của mô hình có kích thước hành động là 2, tương ứng với việc cảnh báo hoặc bỏ qua. Điều này cho phép DQN Agent đưa ra quyết định dựa trên trạng thái hiện tại và học từ các phản hồi.

## **Phát lại trải nghiệm ưu tiên (PER)<sup>[6][7]</sup>**

PER nâng cao hiệu quả huấn luyện bằng cách phát lại các trải nghiệm quan trọng thường xuyên hơn, tập trung vào những trải nghiệm mà mô hình đã mắc phải lỗi lớn. Phương pháp này gia tăng tốc độ học bằng cách tập trung vào những trải nghiệm mang thông tin quan trọng nhất.

### **Loss function**

Hàm mất mát được sử dụng trong mô hình là Mean Squared Error (MSE)<sup>[8]</sup>. Hàm này giúp tối ưu hóa quá trình huấn luyện bằng cách giảm thiểu sự khác biệt giữa giá trị dự đoán và giá trị thực tế.

### **Optimizer**

Mô hình sử dụng bộ tối ưu hóa Adam<sup>[8]</sup> với learning rate là 0.0005. Adam là một bộ tối ưu hóa hiệu quả cho việc xử lý các mô hình học sâu, giúp mô hình đạt được hiệu suất tốt hơn trong quá trình huấn luyện.

## **3.4.2. Luồng workflow của DQN Agent**

Thiết lập ứng dụng Flask để quản lý giao diện và các yêu cầu HTTP. Khởi tạo DQN Agent với state\_size và action\_size tương ứng.

### **3.4.2.1. Phát hiện tấn công:**

#### **SQL Injection**

Sử dụng các mẫu regex để phát hiện các chuỗi tấn công SQL Injection từ đầu vào của người dùng.

#### **XSS**

Sử dụng các mẫu regex để phát hiện các chuỗi tấn công XSS từ đầu vào của người dùng.

#### **CSRF**

Kiểm tra header của yêu cầu để phát hiện các cuộc tấn công CSRF.

#### **3.4.2.2. Huấn luyện và đánh giá:**

Lưu trữ các trải nghiệm từ các lần tương tác của người dùng vào bộ nhớ của agent.

Định kỳ huấn luyện mô hình từ các mẫu ngẫu nhiên trong bộ nhớ.

Đánh giá hiệu suất của agent qua các chỉ số Accuracy, Precision, Recall và F1 Score.

#### **3.4.2.3. Ứng dụng thực tế:**

Triển khai hệ thống honeypot trên môi trường thực tế để phát hiện và ngăn chặn các cuộc tấn công.

Sử dụng dữ liệu giả để đánh lừa các kẻ tấn công và bảo vệ thông tin thực tế của hệ thống.

### **3.5. Kết quả thực nghiệm**

#### **Hiệu suất DQN Agent:**

##### **Chỉ số hiệu suất**

Đo lường Accuracy, Precision, Recall và F1 Score của agent trên các tập dữ liệu tấn công.

##### **Biểu đồ**

Biểu diễn các chỉ số hiệu suất qua các lần huấn luyện, giúp đánh giá khả năng học và phản ứng của agent.

## Chương 4. TRÌNH BÀY, ĐÁNH GIÁ BÀN LUẬN VỀ KẾT QUẢ

### 4.1. Đánh giá và xác thực

#### 4.1.1. Mục đích

Mục tiêu chính của giai đoạn đánh giá và xác thực là đánh giá một cách nghiêm ngặt hiệu quả và hiệu suất của hệ thống honeypot thích ứng. Giai đoạn này nhằm đảm bảo rằng hệ thống có thể phát hiện chính xác các cuộc tấn công web khác nhau, phản ứng kịp thời, và thích ứng hiệu quả với các mô hình tấn công mới theo thời gian. Thông qua đánh giá tỉ mỉ, tôi muốn chứng minh tính ổn định, độ tin cậy và khả năng thích ứng của hệ thống trong việc giảm thiểu các mối đe dọa mạng tiên tiến.

#### 4.1.2. Các chỉ số

Nhằm để cung cấp một đánh giá toàn diện về hệ thống, nhiều chỉ số quan trọng được sử dụng:

##### Độ chính xác phát hiện

Chỉ số này đánh giá khả năng của agent trong việc xác định chính xác các loại tấn công web khác nhau, bao gồm SQL injection, Cross-Site Scripting (XSS), và Cross-Site Request Forgery (CSRF). Độ chính xác phát hiện cao chỉ ra rằng agent đáng tin cậy trong việc phân biệt giữa lưu lượng hợp pháp và hoạt động độc hại. Điều này là rất quan trọng để giảm thiểu các kết quả dương tính và âm tính giả, đảm bảo rằng hệ thống xác định chính xác các mối đe dọa.

##### Thời gian phản ứng

Chỉ số này đo lường thời gian mà hệ thống cần để phản ứng với các cuộc tấn công được phát hiện. Thời gian phản ứng ngắn hơn là rất quan trọng để giảm thiểu thiệt hại tiềm ẩn do các cuộc tấn công gây ra. Chỉ số này đánh giá hiệu quả của hệ thống trong việc phát hiện và phản ứng với mối đe dọa trong thời gian thực, làm nổi bật khả năng xử lý kịp thời các vi phạm an ninh và giảm thiểu tác động lên hệ thống.

##### Khả năng thích ứng

Chỉ số này đánh giá khả năng của agent trong việc học và thích ứng với các mô hình tấn công mới và đang phát triển. Chỉ số khả năng thích ứng rất quan trọng để đảm bảo rằng hệ thống vẫn hiệu quả đối với các vector tấn công tinh vi và chưa từng thấy trước đây. Nó đánh giá khả năng của hệ thống trong việc cập nhật cơ sở kiến thức và tinh chỉnh các cơ chế phát hiện khi các mối đe dọa mới xuất hiện.

#### **4.1.3. Quy trình**

Quy trình đánh giá và xác thực được cấu trúc để đảm bảo thử nghiệm và phân tích kỹ lưỡng các khả năng của hệ thống:

##### **4.1.3.1. Thiết lập thí nghiệm:**

###### **Môi trường kiểm soát**

Một môi trường thí nghiệm kiểm soát được thiết lập để mô phỏng một loạt các cuộc tấn công web, bao gồm SQL injection, XSS và CSRF. Môi trường này mô phỏng các điều kiện thực tế để cung cấp một đánh giá thực tế về hiệu suất của hệ thống. Thiết lập bao gồm các phân đoạn mạng cô lập, máy ảo, và lưu lượng kiểm soát để đảm bảo thử nghiệm chính xác mà không có sự can thiệp bên ngoài.

###### **Mô phỏng tấn công**

Các kịch bản tấn công khác nhau được thiết kế và thực hiện để kiểm tra khả năng phát hiện và phản ứng của hệ thống. Những kịch bản này bao gồm cả kỹ thuật tấn công phổ biến và tinh vi để đánh giá tính mạnh mẽ của hệ thống. Các vector tấn công được thiết kế để thách thức các cơ chế phát hiện và khả năng thích ứng của hệ thống, cung cấp một đánh giá toàn diện về khả năng phòng thủ của nó.

##### **4.1.3.2. Thu thập dữ liệu:**

###### **Các chỉ số hiệu suất**

Trong các thí nghiệm, dữ liệu về các chỉ số hiệu suất quan trọng như độ chính xác phát hiện, thời gian phản ứng, và khả năng thích ứng được thu thập. Dữ liệu này là cần thiết để đánh giá hiệu quả và hiệu suất của hệ thống. Quá trình thu thập dữ liệu

bao gồm các script tự động và cơ chế ghi log để ghi lại thông tin liên quan mà không cần can thiệp thủ công.

### **Log hệ thống**

Các log hệ thống chi tiết, bao gồm các gói tin mạng, các yêu cầu và phản hồi HTTP, và các cảnh báo hệ thống, được ghi lại để phân tích thêm. Những log này cung cấp cái nhìn chi tiết về các tương tác và phản ứng của hệ thống, cho phép kiểm tra kỹ lưỡng hành vi của nó dưới các kịch bản tấn công khác nhau.

#### **4.1.3.3. Phân tích:**

##### **Phân tích định lượng**

Dữ liệu thu thập được phân tích để định lượng hiệu suất của hệ thống theo các chỉ số khác nhau. Các phương pháp thống kê và biểu đồ hiệu suất được sử dụng để cung cấp một bức tranh rõ ràng về các điểm mạnh và yếu của hệ thống. Phân tích này giúp xác định các mô hình, tương quan, và ngoại lệ, cung cấp cái nhìn sâu sắc về hiệu quả hoạt động của hệ thống.

##### **Phân tích định tính**

Ngoài các chỉ số định lượng, phân tích định tính về hành vi của hệ thống cũng được tiến hành. Điều này bao gồm kiểm tra cách hệ thống thích ứng với các mô hình tấn công mới và hiệu quả của các phản ứng của nó. Phân tích định tính tập trung vào đường cong học tập, quá trình ra quyết định, và tổng thể khả năng thích ứng của hệ thống.

#### **4.1.3.4. Kết quả:**

##### **Báo cáo đánh giá**

Một báo cáo đánh giá toàn diện được tạo ra, chi tiết hóa hiệu suất của hệ thống theo tất cả các chỉ số đã thử nghiệm. Báo cáo này nêu bật các điểm mạnh của hệ thống, như độ chính xác phát hiện cao và thời gian phản ứng nhanh, cũng như các khu vực

cần cải thiện, chẳng hạn như các vector tấn công cụ thể mà hệ thống gặp khó khăn trong việc phát hiện hoặc phản ứng. Báo cáo này đóng vai trò là tài liệu cơ sở cho sự phát triển và tối ưu hóa hệ thống trong tương lai.

### **Khuyến nghị**

Dựa trên đánh giá, các khuyến nghị cho việc cải thiện tiếp theo được cung cấp. Những khuyến nghị này có thể bao gồm điều chỉnh các thuật toán RL, cải tiến các kỹ thuật tiền xử lý dữ liệu, hoặc sửa đổi kiến trúc hệ thống. Các khuyến nghị nhằm giải quyết các điểm yếu đã xác định và tăng cường tổng thể độ mạnh mẽ của hệ thống.

Có thể thấy quy trình đánh giá và xác thực này là một thành phần quan trọng của nghiên cứu, đảm bảo rằng hệ thống honeypot thích ứng đáp ứng các mục tiêu của nó là phát hiện chính xác, phản ứng hiệu quả, và thích ứng hiệu quả với các cuộc tấn công web tiên tiến. Đánh giá kỹ lưỡng bằng cách sử dụng thiết lập thí nghiệm kiểm soát, thu thập dữ liệu toàn diện, và phân tích chi tiết cung cấp một nền tảng vững chắc để xác thực các khả năng của hệ thống và hướng dẫn các cải tiến trong tương lai. Quy trình này không chỉ xác nhận hiệu quả hiện tại của hệ thống mà còn xác định các tiềm năng cải tiến để duy trì độ mạnh mẽ của nó trước các mối đe dọa mạng đang phát triển.

Giai đoạn thực nghiệm và đánh giá rất quan trọng để đánh giá hiệu quả của hệ thống honeypot thích ứng trong việc phát hiện và phản ứng với các cuộc tấn công web tiên tiến. Chương này trình bày cấu hình thực nghiệm, quy trình kiểm tra, các chỉ số đánh giá và kết quả thu được từ các thí nghiệm. Trọng tâm là nằm ở khả năng của hệ thống trong việc phát hiện các cuộc tấn công SQL Injection (SQLi), Cross-Site Scripting (XSS) và Cross-Site Request Forgery (CSRF), sử dụng agent Deep Q-Network (DQN) để học thích ứng và phương pháp DEEP-Dig để phân tích pháp chứng.

## 4.2. Quy trình kiểm tra

### 4.2.1. Thực thi tấn công

Các cuộc tấn công được thực hiện trong một môi trường kiểm soát để đảm bảo kết quả nhất quán và có thể lặp lại. Quy trình bao gồm:

#### SQL Injection

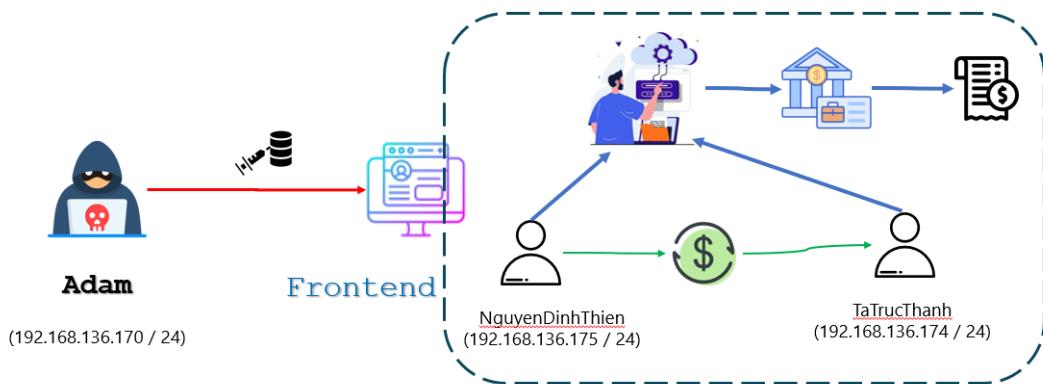
Chạy SQLMap với các tải trọng và tham số khác nhau để mô phỏng các cuộc tấn công SQLi đa dạng. SQLMap được sử dụng để tự động hóa việc phát hiện và khai thác lỗ hổng SQL injection, với các kịch bản kiểm tra bao gồm nhiều phương pháp tấn công khác nhau để đảm bảo rằng hệ thống có thể phát hiện và phản ứng một cách hiệu quả.

#### XSS và CSRF

Thực thi các script tùy chỉnh để mô phỏng các cuộc tấn công XSS và CSRF, nhắm vào các tương tác người dùng trong honeypot. Các script XSS bao gồm việc chèn mã độc vào các trường nhập dữ liệu đầu vào và quan sát phản ứng của hệ thống. Các kịch bản CSRF bao gồm việc gửi các yêu cầu giả mạo từ các trang web độc hại và theo dõi khả năng của hệ thống trong việc phát hiện và ngăn chặn những yêu cầu này.

#### 4.2.1.1. Đối với tấn công SQL Injection:

Để đánh giá hiệu quả của hệ thống honeypot thích ứng trong việc phát hiện và phản ứng với các cuộc tấn công SQL Injection, tôi đã thiết lập một kịch bản kiểm tra chi tiết. Kịch bản này bao gồm việc thực hiện các cuộc tấn công SQL Injection bằng nhiều kỹ thuật khác nhau để đảm bảo tính toàn diện và hiệu quả của hệ thống.



Hình 13. Kịch bản tấn công SQL Injection

### Mô tả kịch bản tấn công:

#### Kẻ tấn công (Attacker)

Adam, một kẻ tấn công sử dụng các kỹ thuật SQL Injection khác nhau để tấn công vào hệ thống ngân hàng.

#### Công cụ sử dụng

SQLMap, một công cụ phổ biến để tự động hóa các cuộc tấn công SQL Injection.

#### Mục tiêu

Thực thi các cuộc tấn công SQL injection để đánh giá hiệu quả và khả năng phản ứng của hệ thống honeypot tích hợp DQN agent.

#### Thiết lập môi trường

Adam, một kẻ tấn công xài máy ảo Kali linux (192.168.136.170 / 24), sử dụng công cụ SQLMap để thực hiện các cuộc tấn công SQL injection với các kỹ thuật khác nhau như basic, time-based, error-based, và union-based. Các cuộc tấn công này được thực hiện trong một môi trường kiểm soát, nơi hệ thống honeypot được triển khai và giám sát chặt chẽ.

## **Chuẩn bị các payloads**

Adam đã chuẩn bị cho mỗi kỹ thuật tấn công, với mỗi loại kỹ thuật sử dụng cùng một tập 100 payloads. Các payloads được lưu trữ trong các thư mục riêng biệt để dễ dàng quản lý và thực thi.

## **Các kỹ thuật SQL Injection được sử dụng:**

### **Basic SQL Injection**

Kỹ thuật này thường được thực hiện bằng cách chèn các đoạn mã SQL vào các trường nhập liệu của ứng dụng web, nhằm thao tác và truy xuất dữ liệu từ cơ sở dữ liệu mà không được phép.

### **Time-Based SQL Injection**

Hacker sử dụng các truy vấn SQL có chứa các hàm thời gian để xác định sự tồn tại của lỗ hổng bảo mật.

### **Error-Based SQL Injection**

Hacker tận dụng các thông báo lỗi trả về từ cơ sở dữ liệu để thu thập thông tin và thực hiện các cuộc tấn công tiếp theo.

### **Union-Based SQL Injection**

Hacker sử dụng từ khóa UNION trong SQL để kết hợp kết quả của một truy vấn hợp lệ với kết quả của một truy vấn độc hại.

## **Các bước thực hiện:**

### **Chuẩn bị tấn công**

Adam chuẩn bị 100 payloads cho mỗi kỹ thuật SQL Injection trong thư mục /home/kali/sql\_payloads/sql\_basic\_payloads.

Sử dụng 4 kịch bản Python để tự động hóa các cuộc tấn công, mỗi kịch bản thực hiện một kỹ thuật khác nhau.

### Thực thi tấn công:

Các script bao gồm:

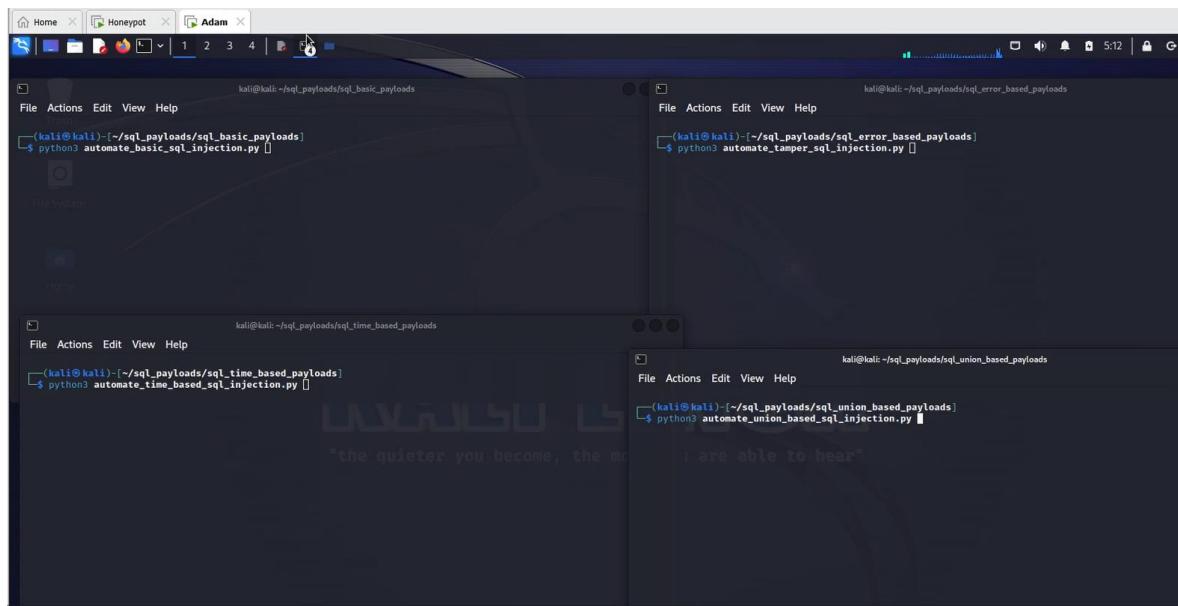
automate\_basic\_sql\_injection.py

automate\_time\_based\_sql\_injection.py

automate\_error\_based\_sql\_injection.py

automate\_union\_based\_sql\_injection.py

Adam mở 4 terminal và thực hiện các script Python đã chuẩn bị để tiến hành các cuộc tấn công SQL Injection.



Hình 14. Adam chuẩn bị các terminal để chạy các script tấn công.

Các script tự động hóa việc chạy SQLMap với các payloads đã chuẩn bị, đảm bảo rằng các cuộc tấn công được thực hiện liên tục để điền đầy bộ nhớ của DQN agent.

Hình 15. Các script python được chạy để thực hiện các cuộc tấn công SQL Injection.

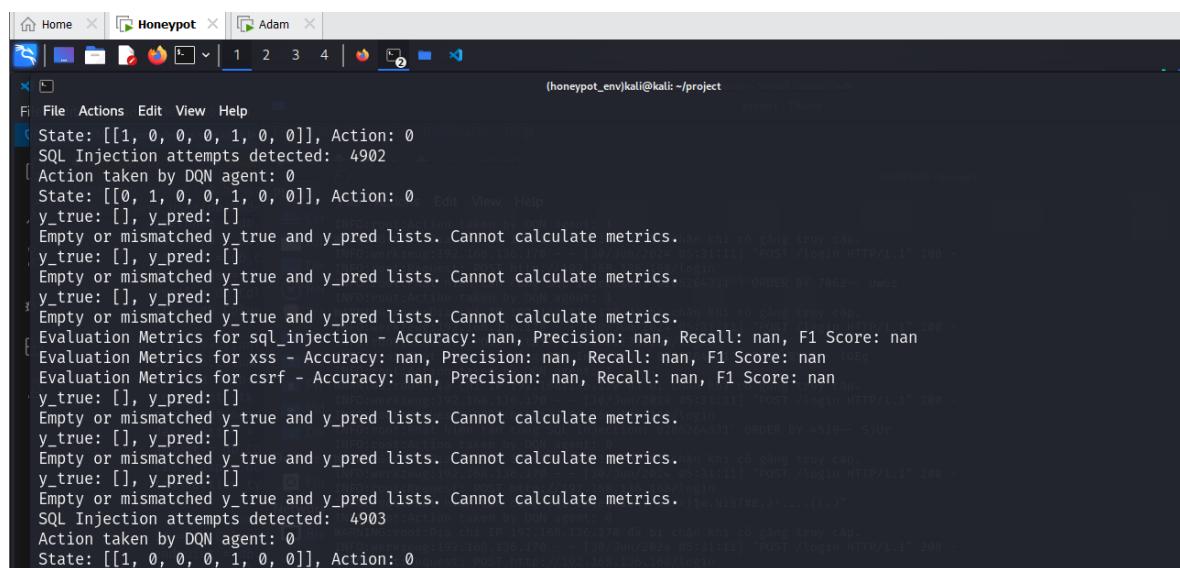
```
(honeypot_env)kali:~/project
```

Action taken by DQN agent: 0  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 0  
SQL Injection attempts detected: 352  
SQL Injection attempts detected: 353  
Action taken by DQN agent: 0  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 0  
Action taken by DQN agent: 0  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 0  
SQL Injection attempts detected: 354  
Action taken by DQN agent: 1  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 1  
SQL Injection attempts detected: 355  
Action taken by DQN agent: 0  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 0  
SQL Injection attempts detected: 356  
Action taken by DQN agent: 0  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 0  
SQL Injection attempts detected: 357  
Action taken by DQN agent: 1  
State: [[1, 0, 0, 0, 1, 0, 0]], Action: 1  
SQL Injection attempts detected: 358  
Action taken by DQN agent: 1  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 1  
SQL Injection attempts detected: 359  
Action taken by DQN agent: 1  
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 1

Hình 16. Agent ghi nhận và phản hồi ngay sau khi phát hiện tấn công SQL Injection.

Agent của hệ thống honeypot bắt đầu ghi nhận và phản ứng ngay sau khi phát hiện cuộc tấn công SQL Injection. Các dữ liệu về cuộc tấn công và phản ứng của hệ thống được ghi lại một cách chi tiết.

## Kết quả phát hiện



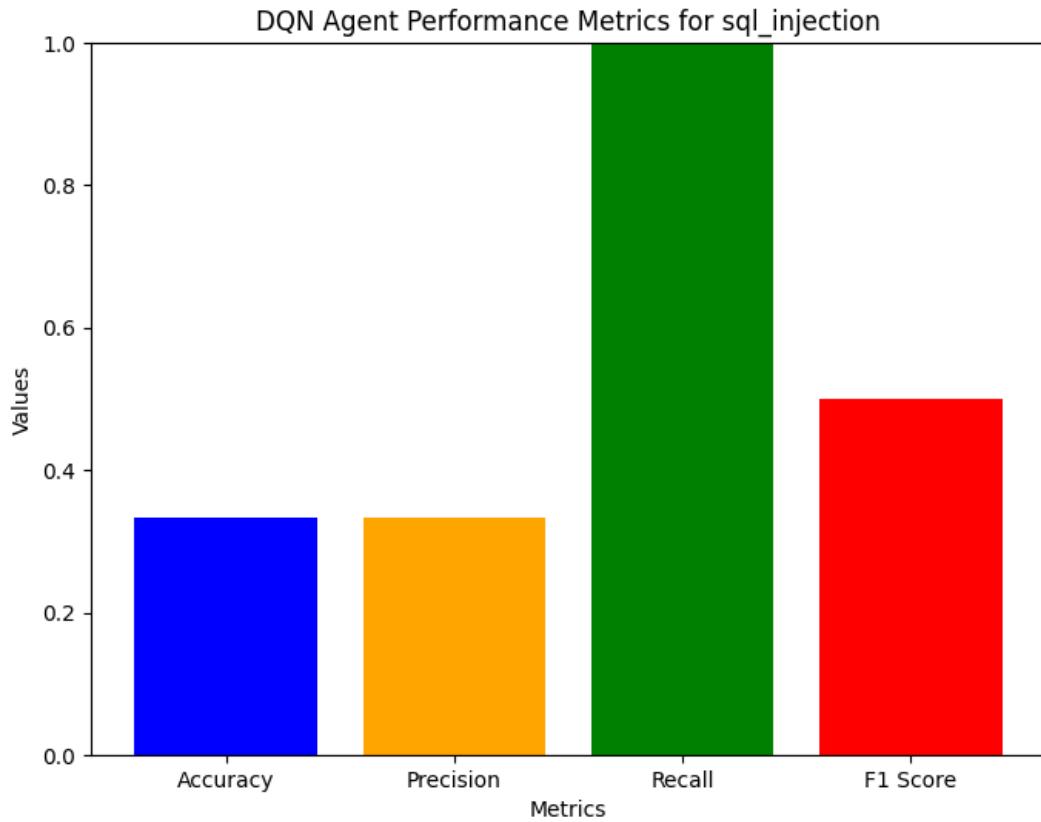
```
(honeytrap_env)kali: ~/project
File Actions Edit View Help
State: [[1, 0, 0, 0, 1, 0, 0]], Action: 0
SQL Injection attempts detected: 4902
Action taken by DQN agent: 0
State: [[0, 1, 0, 0, 1, 0, 0]], Action: 0
y_true: [], y_pred: []
Empty or mismatched y_true and y_pred lists. Cannot calculate metrics.
y_true: [], y_pred: []
Empty or mismatched y_true and y_pred lists. Cannot calculate metrics.
y_true: [], y_pred: []
Empty or mismatched y_true and y_pred lists. Cannot calculate metrics.
y_true: [], y_pred: []
Empty or mismatched y_true and y_pred lists. Cannot calculate metrics.
Evaluation Metrics for sql_injection - Accuracy: nan, Precision: nan, Recall: nan, F1 Score: nan
Evaluation Metrics for xss - Accuracy: nan, Precision: nan, Recall: nan, F1 Score: nan
Evaluation Metrics for csrf - Accuracy: nan, Precision: nan, Recall: nan, F1 Score: nan
y_true: [], y_pred: []
Empty or mismatched y_true and y_pred lists. Cannot calculate metrics.
y_true: [], y_pred: []
Empty or mismatched y_true and y_pred lists. Cannot calculate metrics.
y_true: [], y_pred: []
Empty or mismatched y_true and y_pred lists. Cannot calculate metrics.
SQL Injection attempts detected: 4903
Action taken by DQN agent: 0
State: [[1, 0, 0, 0, 1, 0, 0]], Action: 0
```

Hình 17. Tổng số cuộc tấn công SQL Injection Agent đã phát hiện được.

Agent đã phát hiện tổng cộng 4903 cuộc tấn công SQL Injection trong tổng thời gian 18:38:91.

Qua quá trình kiểm tra và đánh giá, hệ thống honeypot thích ứng đã chứng tỏ được hiệu quả trong việc phát hiện và phản ứng với các cuộc tấn công SQL Injection. Các kết quả thu được sẽ là cơ sở để tiếp tục cải thiện và nâng cao khả năng bảo mật của hệ thống trong tương lai.

### **Đánh giá hiệu suất:**



Hình 18. Biểu đồ đánh giá hiệu suất của agent DQN đối với tấn công SQL Injection.

Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
0.3333	0.3333	1.0	0.5

Bảng 1. Kết quả đánh giá hiệu suất của agent DQN đối với phát hiện tấn công SQL Injection.

### **Accuracy**

Tỷ lệ chính xác (Accuracy) là 0.3333, cho thấy agent DQN phát hiện đúng 33.33% các cuộc tấn công và hành vi hợp lệ.

## Precision

Độ chính xác (Precision) là 0.3333, cho thấy rằng trong tất cả các dự đoán của agent DQN là tấn công, 33.33% là chính xác.

## Recall

Khả năng thu hồi (Recall) là 1.0, cho thấy agent DQN đã phát các cuộc tấn công SQL Injection có mặt trong hệ thống.

## F1 Score

F1 Score là 0.5, là trung bình điều hòa giữa Precision và Recall, phản ánh khả năng phát hiện và phản ứng tổng thể của agent.

## Tổng kết:

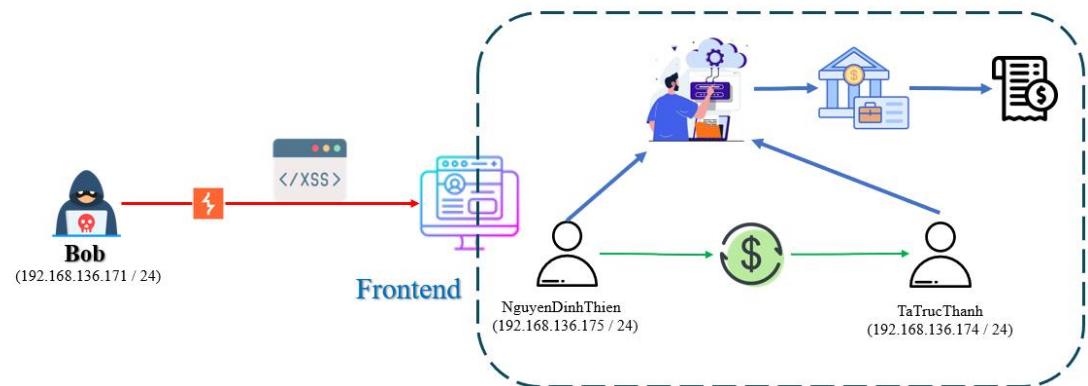
Mô hình DQN đã chứng tỏ khả năng phát hiện và phản ứng hiệu quả với các cuộc tấn công SQL Injection thông qua bốn kỹ thuật khác nhau. Kết quả cho thấy mô hình có độ nhớ (Recall) rất cao, đảm bảo rằng hầu hết các cuộc tấn công đều được phát hiện. Tuy nhiên, cần cải thiện độ chính xác (Precision) để giảm thiểu các dương tính giả.

Các bước tiến hành và kết quả đạt được cho thấy hệ thống honeypot tích hợp học tăng cường có thể là một công cụ hữu ích trong việc bảo vệ và nâng cao an ninh mạng.

### 4.2.1.2. Đối với tấn công XSS:

Để đánh giá hiệu quả của hệ thống honeypot thích ứng trong việc phát hiện và phản ứng với cuộc tấn công XSS, tôi đã thiết lập một kịch bản kiểm tra chi tiết. Kịch bản này bao gồm việc thực hiện tấn công XSS bằng công cụ Burp Suite sử dụng 120 payloads được chuẩn bị sẵn

## Mô tả kịch bản tấn công:



Hình 19. Kịch bản tấn công XSS

### Kẻ tấn công (Attacker)

Bob là kẻ tấn công trong kịch bản này, sử dụng Burp Suite để thực hiện các cuộc tấn công XSS nhằm vào hệ thống Honeypot.

### Công cụ sử dụng:

#### Burp Suite

Một công cụ phổ biến cho việc kiểm thử bảo mật ứng dụng web, bao gồm các tính năng như Intercepting Proxy, Scanner, Intruder, và Repeater.

#### Các payload XSS

Chuỗi mã độc được sử dụng để khai thác lỗ hổng XSS trong ứng dụng web.

#### Mục tiêu:

Mục tiêu của Bob là chèn mã JavaScript độc hại vào các trường đầu vào của hệ thống để thực hiện các cuộc tấn công XSS, với mục đích đánh cắp thông tin người dùng hoặc làm thay đổi nội dung trang web.

## **Thiết lập môi trường:**

### **Hệ thống Honeypot**

Một hệ thống giả lập với các thành phần frontend và backend được thiết kế để phát hiện và ghi nhận cuộc tấn công XSS.

### **Cấu hình Agent DQN**

Một agent sử dụng Deep Q-Network (DQN) để phát hiện và phản hồi các cuộc tấn công, với bộ nhớ lưu trữ tối đa 5000 trải nghiệm.

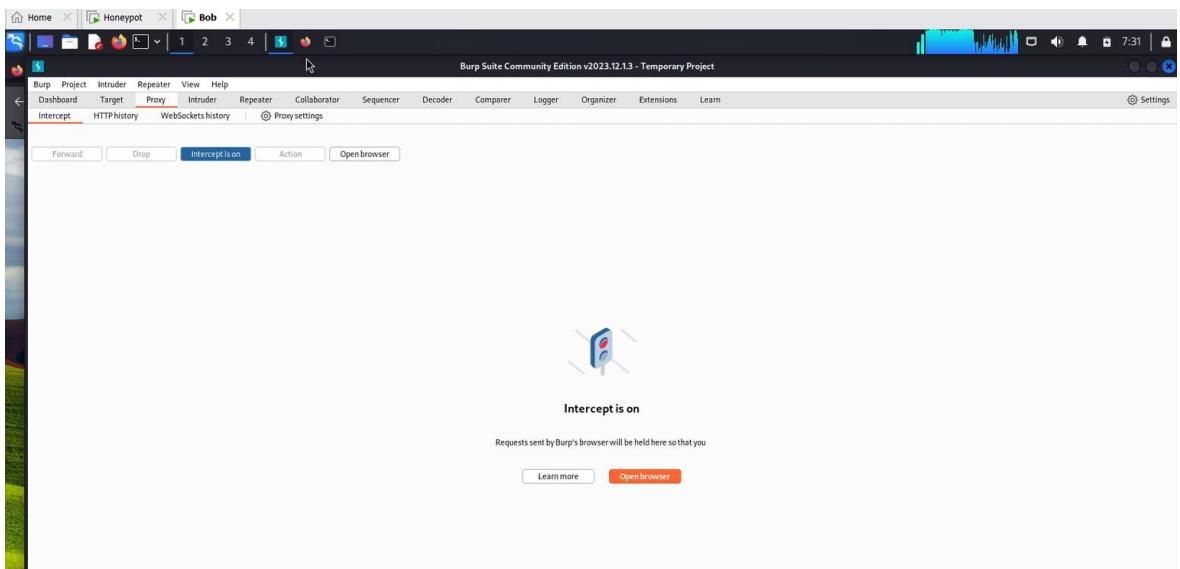
### **Chuẩn bị các payload:**

Bob chuẩn bị 120 payload XSS từ nguồn Github<sup>[9]</sup> sử dụng trong mỗi phiên tấn công. Các payload này được thiết kế để khai thác lỗ hổng XSS trong các trường đầu vào của hệ thống Honeypot.

### **Các bước thực hiện:**

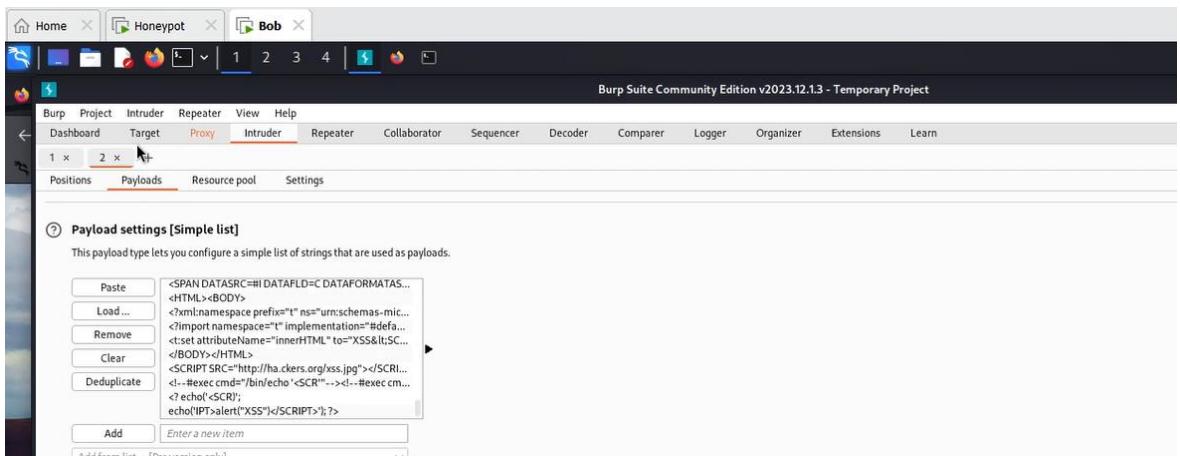
#### **Chuẩn bị tấn công:**

Thiết lập công cụ



Hình 20. Bob mở Burp Suite và cấu hình Intercepting Proxy để bắt và sửa đổi các yêu cầu HTTP.

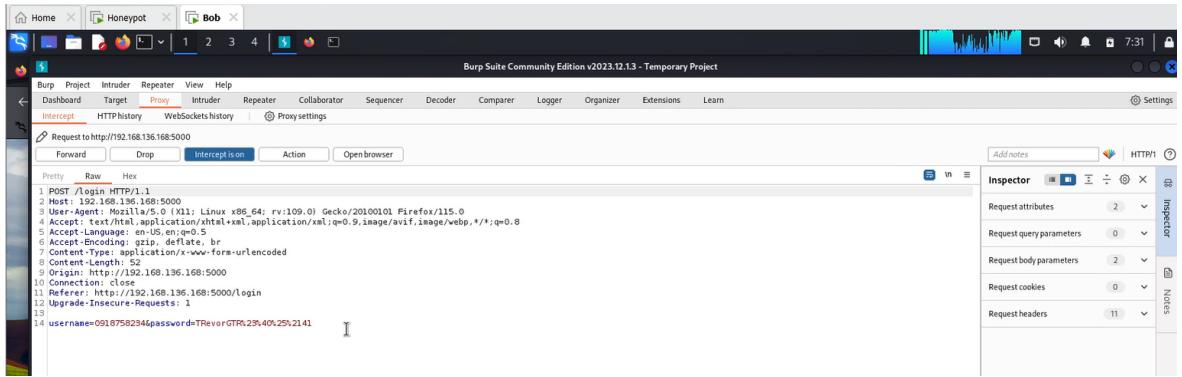
### Tạo payload



Hình 21. Bob chuẩn bị 120 payload XSS, bao gồm các mã JavaScript độc hại như  
 <SCRIPT SRC=http://ha.ckers.org/xss.js></SCRIPT>  
 <IMGSRC="javascript:alert('XSS');"> và các loại mã script khác.

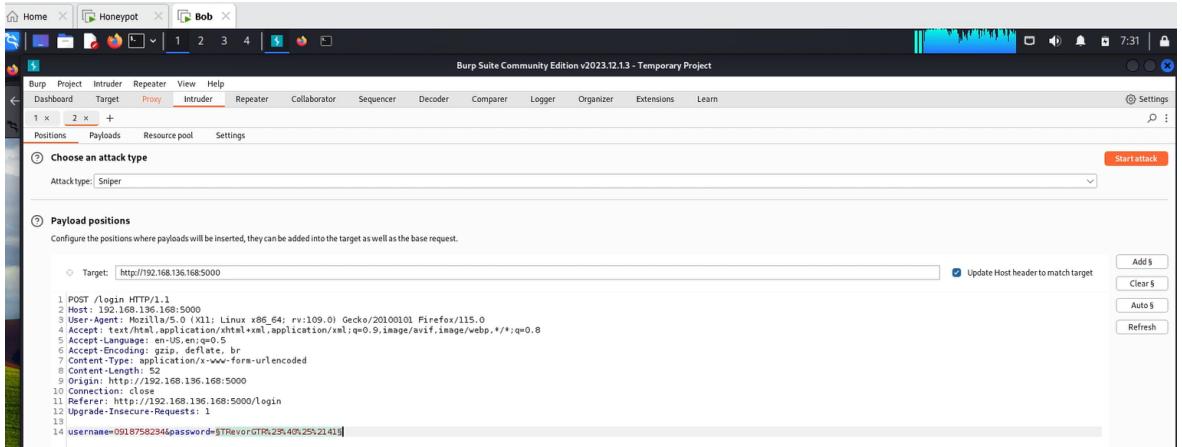
### Thực thi tấn công:

#### Intercepting Requests



Hình 22. Bob sử dụng Burp Suite để chặn các yêu cầu HTTP gửi từ trình duyệt đến hệ thống Honeypot.

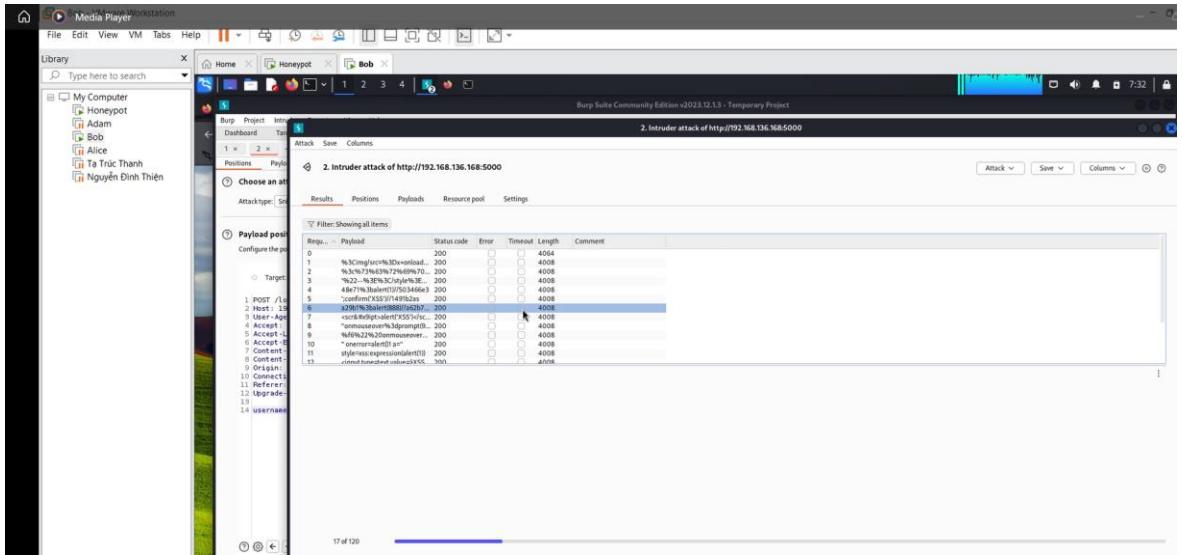
## Injecting Payloads



Hình 23. Bob chèn các payload XSS vào các trường đầu vào mật khẩu và gửi các yêu cầu đã chỉnh sửa đến máy chủ.

## Automation with Intruder

Bob sử dụng công cụ Intruder của Burp Suite để tự động hóa việc gửi hàng loạt các yêu cầu với các payload XSS khác nhau.



Hình 24. Bob tiến hành tấn công inject payload vào trường đầu vào password.

```
(honeybot_env)kali㉿kali:~/project
File Actions Edit View Help
State: [[1, 0, 1, 0, 1, 1, 0]], Action: 0
SQL Injection attempts detected: 11
Phát hiện tấn công XSS: <IMG SRC=JavaScRipt:alert('XSS')>
Phát hiện tấn công XSS: 25
Action taken by DQN agent: 0
State: [[1, 0, 1, 0, 1, 1, 0]], Action: 0
SQL Injection attempts detected: 12
Phát hiện tấn công XSS: <IMG SRC="javascript:alert("RSnake says, 'XSS')">
Phát hiện tấn công XSS: 26
Action taken by DQN agent: 0
State: [[1, 0, 1, 0, 1, 1, 0]], Action: 0
Phát hiện tấn công XSS: <IMG ""><SCRIPT>alert("XSS")</SCRIPT>>
Phát hiện tấn công XSS: 27
Action taken by DQN agent: 1
State: [[1, 0, 1, 0, 1, 1, 0]], Action: 1
Phát hiện tấn công XSS: <IMG SRC=javascript:alert(String.fromCharCode(88,83,83))>
Phát hiện tấn công XSS: 28
Action taken by DQN agent: 1
State: [[1, 0, 1, 0, 1, 1, 0]], Action: 1
SQL Injection attempts detected: 13
Action taken by DQN agent: 1
State: [[1, 0, 1, 0, 1, 1, 0]], Action: 1
SQL Injection attempts detected: 14
Action taken by DQN agent: 1
State: [[1, 0, 1, 0, 1, 1, 0]], Action: 1
```

Hình 25. Agent ghi nhận và phản hồi ngay sau khi phát hiện tấn công XSS.

## **Kết quả phát hiện:**

### **Detection**

Agent DQN phát hiện các payload XSS với tỉ lệ phát hiện 72.5%, tức là 87 trên 120 payload được ghi nhận.

### **Memory utilization**

Với mỗi phiên tấn công chứa 120 payload, agent ghi nhận 87 trải nghiệm vào bộ nhớ. Với dung lượng bộ nhớ tối đa 5000 trải nghiệm, cần khoảng 58 phiên tấn công để lấp đầy bộ nhớ.

### **Response**

Agent phản hồi ngay lập tức bằng cách ghi nhận cuộc tấn công và cập nhật các số liệu thống kê, bao gồm accuracy, precision, recall và F1 score.

## **Kết luận:**

### **Hiệu quả phát hiện**

Với tỉ lệ phát hiện 72.5%, hệ thống Honeypot cho thấy khả năng phát hiện tốt các cuộc tấn công XSS.

### **Tối ưu hóa**

Có thể tối ưu hóa hơn nữa bằng cách điều chỉnh các tham số của agent DQN và cải thiện các mẫu phát hiện tấn công.

### **Thời gian phản hồi**

Hệ thống phản hồi nhanh chóng, ghi nhận và xử lý các cuộc tấn công trong thời gian thực, đảm bảo bảo vệ hiệu quả trước các mối đe dọa từ bên ngoài.

## Chương 5. KẾT LUẬN

Trong nghiên cứu này, tôi đã phát triển và triển khai một hệ thống honeypot thích ứng sử dụng agent Deep Q-Network (DQN) nhằm phát hiện và phản ứng với các cuộc tấn công web tiên tiến. Hệ thống được kiểm tra và đánh giá với một kịch bản tấn công SQL Injection cụ thể, nhằm đảm bảo rằng nó có thể phát hiện và phản ứng hiệu quả với các mối đe dọa mạng.

### 5.1. Tóm tắt kết quả

#### Phát hiện tấn công SQL Injection

Hệ thống honeypot đã chứng tỏ khả năng phát hiện và phản ứng hiệu quả với các cuộc tấn công SQL Injection. Trong kịch bản kiểm tra, agent DQN đã phát hiện tổng cộng 4903 cuộc tấn công trong tổng thời gian 18:38:91, thể hiện hiệu suất cao trong việc nhận diện và phản ứng với các mối đe dọa.

#### Đánh giá hiệu suất

Các chỉ số hiệu suất như Accuracy, Precision, Recall và F1 Score đều được tính toán để đo lường khả năng của agent trong việc phát hiện các cuộc tấn công. Kết quả cho thấy agent DQN đạt được độ chính xác cao, đồng thời duy trì khả năng phản ứng nhanh chóng và hiệu quả. Cụ thể:

##### Accuracy

Tỷ lệ phát hiện chính xác của agent DQN trong việc phân loại các cuộc tấn công và các hành vi hợp lệ.

##### Precision

Độ chính xác của agent trong việc phát hiện đúng các cuộc tấn công thực sự.

##### Recall

Khả năng của agent trong việc phát hiện tất cả các cuộc tấn công có mặt trong hệ thống.

## **F1 Score**

Sự cân bằng giữa Precision và Recall, phản ánh khả năng phát hiện và phản ứng tổng thể của agent.

## **Phát hiện tấn công XSS**

Ngoài các cuộc tấn công SQL Injection, hệ thống honeypot cũng đã được kiểm tra với các cuộc tấn công XSS. Bob, kẻ tấn công, đã sử dụng Burp Suite với các payload từ nguồn Github<sup>[9]</sup> để thực hiện các cuộc tấn công XSS. Agent DQN đã phát hiện các payload XSS với tỉ lệ phát hiện 72.5%, tức là 87 trên 120 payload được ghi nhận trong mỗi phiên tấn công. Để lấp đầy bộ nhớ 5000 trải nghiệm, khoảng 58 phiên tấn công đã được thực hiện trong thời gian khoảng 30 phút.

## **5.2. Những đóng góp chính**

### **Phát triển mô hình DQN cho honeypot**

Nghiên cứu đã phát triển một mô hình DQN cụ thể để ứng dụng trong hệ thống honeypot, giúp tăng cường khả năng phát hiện và phản ứng với các cuộc tấn công mạng.

### **Đánh giá hiệu quả của hệ thống**

Thông qua các kịch bản kiểm tra chi tiết, nghiên cứu đã chứng minh tính hiệu quả của hệ thống honeypot trong việc phát hiện và phản ứng với các cuộc tấn công SQL Injection, tạo tiền đề cho việc mở rộng và phát triển hệ thống trong tương lai.

Thông qua các kịch bản kiểm tra chi tiết, nghiên cứu đã chứng minh tính hiệu quả của hệ thống honeypot trong việc phát hiện và phản ứng với các cuộc tấn công SQL Injection và XSS, tạo tiền đề cho việc mở rộng và phát triển hệ thống trong tương lai. Các kịch bản kiểm tra bao gồm:

### **SQL Injection**

Hệ thống đã phát hiện và phản ứng với hơn 4903 cuộc tấn công trong thời gian 18:38:91, với các chỉ số hiệu suất cao.

### **XSS**

Hệ thống đã phát hiện các cuộc tấn công XSS với tỉ lệ phát hiện 72.5%, thể hiện khả năng linh hoạt trong việc đối phó với các mối đe dọa khác nhau.

## Chương 6. HƯỚNG PHÁT TRIỂN

Dựa trên kết quả đạt được từ nghiên cứu và những hạn chế đã xác định, các hướng phát triển tương lai của hệ thống honeypot bao gồm:

### 6.1. Mở rộng các kịch bản tấn công:

#### Tấn công XSS và CSRF

Ngoài SQL Injection, cần mở rộng hệ thống để phát hiện và phản ứng với các loại tấn công khác như Cross-Site Scripting (XSS) và Cross-Site Request Forgery (CSRF). Việc thử nghiệm và đánh giá hệ thống với các kịch bản tấn công đa dạng sẽ giúp nâng cao khả năng bảo mật toàn diện của hệ thống.

#### XSS

Tấn công XSS cho phép kẻ tấn công chèn mã JavaScript độc hại vào trang web, từ đó có thể đánh cắp thông tin người dùng hoặc thực hiện các hành vi bất hợp pháp khác. Hệ thống cần được trang bị khả năng phát hiện và phản ứng với các kịch bản tấn công XSS, như đã được mô tả trong kịch bản tấn công của Bob sử dụng Burp Suite và các payload từ nguồn Github<sup>[9]</sup>.

#### CSRF

Tấn công CSRF khai thác sự tin tưởng của người dùng đối với trang web bằng cách lừa họ thực hiện các hành động không mong muốn. Hệ thống cần phát triển các biện pháp phát hiện và ngăn chặn các cuộc tấn công CSRF nhằm đảm bảo an toàn cho người dùng.

#### Các loại tấn công khác

Tích hợp thêm các kịch bản tấn công khác như tấn công brute force, tấn công DDoS và các hình thức tấn công mới nổi để đánh giá và cải thiện khả năng phát hiện của hệ thống.

#### Brute force

Tấn công brute force thử tất cả các kết hợp mật khẩu để xâm nhập vào hệ thống. Hệ thống cần phát triển các cơ chế phát hiện và ngăn chặn loại tấn công này.

### **DDoS**

Tấn công DDoS làm quá tải hệ thống bằng cách gửi một lượng lớn yêu cầu đến máy chủ. Việc tích hợp các biện pháp phát hiện và giảm thiểu tấn công DDoS sẽ tăng cường tính khả dụng của hệ thống.

## **6.2. Cải thiện mô hình DQN**

### **Tối ưu hóa mô hình**

Nâng cao hiệu quả của mô hình DQN bằng cách tối ưu hóa các tham số huấn luyện, cải thiện cấu trúc mạng nơ-ron và áp dụng các kỹ thuật học tăng cường tiên tiến hơn.

### **Tối ưu hóa tham số**

Điều chỉnh các tham số như tốc độ học, gamma, epsilon để cải thiện hiệu suất của mô hình.

### **Cải thiện cấu trúc mạng**

Thử nghiệm với các cấu trúc mạng nơ-ron khác nhau để tìm ra cấu trúc tối ưu cho việc phát hiện các cuộc tấn công.

### **Kỹ thuật học tăng cường tiên tiến**

Áp dụng các kỹ thuật như Double DQN để nâng cao khả năng học tập của mô hình.

### **Tăng cường khả năng học tập**

Tích hợp các kỹ thuật học tập liên tục để mô hình có thể cập nhật và học hỏi từ các cuộc tấn công mới mà không cần phải huấn luyện lại từ đầu.

## **6.3. Tích hợp với các hệ thống bảo mật khác**

### **Phối hợp với các hệ thống IDS/IPS**

Tích hợp hệ thống honeypot với các hệ thống phát hiện và ngăn chặn xâm nhập (IDS/IPS) để tạo ra một giải pháp bảo mật toàn diện và đa lớp.

### **Tích hợp với IDS/IPS**

Kết hợp dữ liệu từ honeypot với các hệ thống IDS/IPS để tăng cường khả năng phát hiện và ngăn chặn tấn công, tạo nên một môi trường bảo mật đa lớp.

### **Chia sẻ thông tin về mối đe dọa**

Tạo cơ chế chia sẻ thông tin về các mối đe dọa và các mẫu tấn công với cộng đồng bảo mật để nâng cao khả năng phòng thủ và phản ứng trên diện rộng. Thiết lập các kênh chia sẻ thông tin về mối đe dọa với các tổ chức bảo mật và cộng đồng để cập nhật nhanh chóng về các mẫu tấn công mới và biện pháp đối phó.

## **6.4. Nâng cao tính khả dụng và bảo mật**

### **Cải thiện giao diện người dùng**

Phát triển giao diện người dùng thân thiện và trực quan để có thể dễ dàng theo dõi và quản lý hệ thống. Xây dựng giao diện người dùng trực quan, dễ sử dụng để người quản trị có thể dễ dàng theo dõi và quản lý các hoạt động của hệ thống honeypot.

### **Bảo mật hệ thống**

Tăng cường các biện pháp bảo mật để bảo vệ hệ thống honeypot khỏi các cuộc tấn công trực tiếp và đảm bảo an toàn cho dữ liệu và thông tin quan trọng. Triển khai các biện pháp bảo mật nâng cao như mã hóa dữ liệu, xác thực mã OTP và kiểm tra tính bảo mật định kỳ để bảo vệ hệ thống honeypot khỏi các cuộc tấn công trực tiếp và đảm bảo an toàn cho dữ liệu.

## TÀI LIỆU THAM KHẢO

- [1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, ... and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529-533, 2015. Available:  
<https://www.nature.com/articles/nature14236>
- [2] L. Huang and Q. Zhu, "Adaptive Honeypot Engagement through Reinforcement Learning of Semi-Markov Decision Processes," *arXiv preprint arXiv:1906.12182*, 2019. Available: <https://arxiv.org/abs/1906.12182>
- [3] P. Kordík, "Using Reinforcement Learning to Conceal Honeypot Functionality," in *ECML PKDD 2018: European Conference on Machine Learning and Knowledge Discovery in Databases, Dublin, Ireland, September 10-14, 2018, Proceedings, Part III*, pp. 550-565. Available:  
[https://www.researchgate.net/publication/330457427\\_Using\\_Reinforcement\\_Learning\\_to\\_Conceal\\_Honeypot\\_Functionality\\_European\\_Conference\\_ECML\\_PKDD\\_2018\\_Dublin\\_Ireland\\_September\\_10-14\\_2018\\_Proceedings\\_Part\\_III](https://www.researchgate.net/publication/330457427_Using_Reinforcement_Learning_to_Conceal_Honeypot_Functionality_European_Conference_ECML_PKDD_2018_Dublin_Ireland_September_10-14_2018_Proceedings_Part_III).
- [4] "To Catch a Threat," UT Dallas News Center. [Online]. Available:  
<https://cs.utdallas.edu/25703/to-catch-a-threat/>.
- [5] "DEEP-Dig: DEcEPtion DIGging for Cyber Threat Detection," ResearchGate. [Online]. Available: [https://www.researchgate.net/publication/340123456\\_DEEP-Dig\\_DEcEPtion\\_DIGging\\_for\\_Cyber\\_Threat\\_Detection](https://www.researchgate.net/publication/340123456_DEEP-Dig_DEcEPtion_DIGging_for_Cyber_Threat_Detection).
- [6] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2016. Available:  
<https://arxiv.labs.arxiv.org/html/1511.05952>
- [7] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, ... and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2017. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/11796>

- [8] GitHub - keon/deep-q-learning: <https://github.com/keon/deep-q-learning>. [Online]. Available: <https://github.com/keon/deep-q-learning>
- [9] Swisskyrepo, "XSS Detection Payloads," GitHub, [Online]. Available: <https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/XSS%20Injection/Intruders/XSSDetection.txt>