A STATEFUL FIREWALL PACKET ANALYSIS FRAMEWORK FOR MITIGATING SESSION FIXATION ATTACKS

KAILANYA EUNICE

A Thesis Submitted in Partial Fulfilment of the Requirement for Conferment of the Master of Science in Computer Science of Meru University of Science and Technology

2025

DECLARATION

This thesis is my original work and has not been presented for a degree in any other institution.

CT401/200901/19

Signed…….Date…….

Kailanya Eunice

DECLARATION BY SUPERVISORS

This thesis has been submitted with our approval as University supervisors

Signed…….Date……

Prof. Amos Odhiambo Omamo, Ph.D.

Meru University of Science and Technology, Kenya

Signed…….Date……

Dr. Mary Walowe Mwadulo, Ph.D.

Meru University of Science and Technology, Kenya

DEDICATION

I dedicate this work to all my loved ones.

ACKNOWLEDGMENT

I cannot express enough thanks to my supervisors Prof. Amos Odhiambo Omamo and Dr. Mary Walowe Mwadulo for their continued support and encouragement: First and foremost, I also thank the entire Members of the School of Computing and Informatics for their Moral support. My classmates and my children have supported in completion of the research. Finally, I thank my loving and supportive husband, your encouragements are much appreciated and dully noted.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FORMULA

LIST OF APPENDICES

LIST OF ABBREVIATIONS AND ACRONYMS

ARP	Address Resolution Protocol
DOS	Denial-Of-Service
DDOS	Distributed Denial of Service
FOL	First Order Logic
FTP	File Transfer Protocol
ICMP	Internet Control Message Protocol
ICT	Information Communication Technology
LAN	Local Area Network
NACL	Network Access Control List
OSI	Open Systems Interconnection
PAN	Personal Area Network
PCAP	Packet Capture
SFPAM	Stateful Firewall Packet Analysis Mode
SMTP	Simple Mail Transfer Protocol
SPIM	Stateful Packet Inspection Model
SQL	Structured Query Language
TCP/IP	Transmission Control Protocol/ Internet Protocol
WAN	Wide Area Network

OPERATIONAL DEFINITION OF TERMS

Stateful Firewall: The ability to understand and remember the state of network Connections over time.

Session Fixation: attack that permits an invader to hijack a valid user session

Protocol: Set of rules specifying how connected devices communicate across a network

Dataset: Collection of data used to train machine learning

Cross site scripting: Web vulnerability that allows attackers to users interactions with Application

Events: user actions/input

ABSTRACT

Protecting the networks against web attacks has become increasingly critical. As network attacks continue to evolve in complexity and sophistication, stateful firewall solutions have proven to be insufficient in defending against session fixation attacks. Session fixation attacks pose a significant threat to web security by exploiting vulnerabilities in session management to hijack authenticated user sessions. Existing stateful firewall models can filter attacks such as denial of service, distributed denial of service, man-in-the-middle, malware, ransomware and spamming. However, they are unable to filter session fixation attacks due to their filtering mechanisms. The aim of this study was to develop a stateful firewall packet analysis model that operates in network layer to detect and filter session fixation attack. By maintaining state information across network sessions, the model analyzed packet sequences and patterns to identify anomalies indicative of session fixation attempts. Gradient booster classifier algorithm was incorporated into the model to enhance accuracy in analyzing the packet. Virtual machine simulation experiment was performed to evaluate the accuracy of the model using Cross-Site Scripting (XSS) datasets vulnerable to session fixation attacks alongside normal user traffic. The model detection rate, false positive and false negative metrics was measured to assess the accuracy of the model. The experimental results demonstrated that the model effectively detected and mitigated session fixation attacks by analyzing session parameters and maintaining session state consistency. Experimental evaluation validated the high model detection accuracy level of 98.5 % with minimal false positives. By tracking the state of each session and analyzing packet-level data the model is capable of detecting suspicious patterns associated with session fixation attempts. The adoption and integration of the model into the network security framework not only strengthens protection at the application layer but also reduces the risk of session hijacking.

CHAPTER ONE: INTRODUCTION

1.1 Overview

This chapter presents the introduction to the work presented herein. It provides the crucial background of the study, problem statement, research objectives, and research questions. Further, the chapter outlines the justification and significance of the study, its scope and limitations and finally the chapter presents contribution and assumption of the study.

1.2 Background of the Study

Firewall is a network security software tool that monitors incoming and outgoing network traffic and permits or blocks data packets based on a set of security rules (Teng et al., 2022). A firewall is considered a first line defense for securing the network and it is the barrier that sits between a private internal network and the public internet. Firewalls operates at layer three, the transport layers of TCP/IP model (Lee et al., 2024).Firewalls operate by inspecting incoming and outgoing traffic flow using a rule-based engine. This engine matches the packets sequentially with a predefined set of rules, namely access rules, and decide whether to block the packet or not (Hadji & Kholladi, 2023). Firewalls are classified as stateless or stateful (Matovic et al., 2023) . Stateless firewall allows a packet to pass through the network if only the source and destination is known and block it if the source and destination is not known (Alicea & Alsmadi, 2021).

Stateful firewalls are a type of network firewall that monitors and tracks the state of active connections passing through the firewall. Unlike stateless firewalls, which examine each packet in isolation, stateful firewalls maintain context about the communication, enabling more intelligent filtering decisions. (Squarcina et al., n.d.). Therefore, stateful firewall can allow a new packet to pass through the network if data in the new packet resembles the data

stored in the state table, if the data does not resemble, the packet is blocked (Gbur & Tschorsch, 2021).

The reason why stateful firewall is commonly used today is that, it offers high security as compared to stateless firewall (Lei et al., 2021). In a stateful firewalls the state table stores session information such as IP address of the sender, IP address of the recipient, port number, connection status and protocol (Z. Wang et al., 2020a). A port number in a state table is a way to identify a specific process to which a message is to be forwarded (Matovic et al., 2023). While a connection status in a state table indicates an actual state of the network, and a protocol in a state table provides a medium set of rules to establish communication between different devices for the exchange of data and services (Aliea & Alsmadi, 2021). Firewall packet analysis is a technology that monitors the state of active connections and uses the information to determine which network packet to allow through the firewall (Lavrishchev et al., 2021).

Firewall technology has evolved significantly over the decades to keep pace with emerging cyber threats and the growing complexity of networks. Initially, firewalls operated as simple packet filters, inspecting basic information like IP addresses and ports to allow or block traffic (Roopesh, 2024). As attacks became more sophisticated, stateful inspection firewalls emerged, tracking the state of active connections and offering improved traffic analysis. With the rise of web-based threats and application-level attacks, next-generation firewalls (NGFWs) were developed to incorporate deep packet inspection, intrusion prevention systems (IPS), and application awareness. Today, modern firewalls also integrate with threat intelligence feeds, support cloud environments, and use machine learning to detect anomalies and zero-day attacks (Agrawal et al., 2024). This evolution reflects a shift from

static, perimeter-based defense to dynamic, intelligent, and distributed protection capable of countering modern cyber threats.

Stateful firewalls were introduced in the early 1990s as a major improvement over first-generation packet-filtering firewalls, which could only inspect individual packets based on static rules involving IP addresses, ports, and protocols (Ramesh et al., 2024). Unlike their predecessors, stateful firewalls maintained awareness of active network sessions by tracking the state and context of connections using a session or state table. This allowed them to evaluate traffic flows rather than isolated packets, enabling more accurate detection of unauthorized or abnormal behavior such as spoofing or session hijacking. Pioneered by companies like Check Point with the release of FireWall-1 in 1994, stateful inspection quickly became a standard in enterprise network security (Katragadda & Faulkner, 2022). Over time, stateful firewalls laid the foundation for more advanced technologies, including Next-Generation Firewalls (NGFWs), by introducing context-aware and behavior-based filtering capabilities that were essential in adapting to evolving cybersecurity threats.

Session fixation attacks are a type of web-based security threat where an attacker forces or tricks a victim into using a known or predetermined session ID, allowing the attacker to hijack the user's session after they log in (Lamdakkar et al., 2024). Unlike session hijacking, which typically involves stealing an active session ID, session fixation exploits poor session management practices by setting the session identifier before the user authenticates. This occurs through various vectors such as URL parameters, hidden form fields, or insecure cookies (Sarman Zürich, 2025). If the application does not regenerate the session ID upon successful login, the attacker can access the authenticated session using the fixed ID. These attacks highlight the importance of secure session handling, including regenerating session

tokens after login and rejecting externally supplied session IDs, to prevent unauthorized access to user accounts.

Machine learning has the potential to significantly enhance stateful firewall capabilities by enabling more intelligent and adaptive traffic analysis beyond traditional rule-based methods. Unlike conventional firewalls that rely on predefined patterns and static rules, machine learning algorithms can analyze vast amounts of network data in real time to detect subtle anomalies and evolving attack behaviors, including complex threats like session fixation that often evade standard inspection (Prazeres et al., 2023). By learning normal traffic patterns and user behaviors, these systems can identify deviations indicative of malicious activity, such as unusual session ID usage or suspicious authentication flows. Integrating machine learning with stateful firewalls can thus improve threat detection accuracy, reduce false positives, and enable proactive defense by predicting and blocking attacks before they cause harm (Md Rashed Buiya et al., 2024). This approach not only strengthens network security but also helps organizations keep pace with rapidly evolving cyber threats through continuous learning and adaptation.

Firewall packet analysis is a technology commonly deployed in modern network security infrastructure to mitigate threats (Wang et al., 2020). Firewall packet analysis detects communications over a period of time and examines both incoming and outgoing packets (Yuan et al., 2020). The firewall follows outgoing packets that request specific sorts of incoming packets and authorize incoming packets to pass though the network (Khan, 2023).

Session fixation is a web-based attack where an attacker sets a known session ID for a legitimate user before they log in. Once the user authenticates using that session ID, the attacker, who already knows the ID, can hijack the session and gain unauthorized access to

the user's account. This exploit takes advantage of web applications that do not generate a new session ID upon successful login, allowing the attacker to maintain access (Morshedi, 2023). Session fixation can be executed by tricking users into clicking on a crafted link with a preset session ID or by injecting the ID through other methods like form fields or cookies. The attack highlights the importance of secure session management practices in web development (Chandrappa et al., 2025).

To prevent session fixation attacks, network and application-level security measures should be implemented. One of the most critical defenses is regenerating the session ID immediately after a user logs in, ensuring that any previously fixed session ID becomes invalid (Kiran Gandikota et al., 2023). Web applications should also use secure cookie attributes such as Http Only and Secure to prevent access to session IDs via client-side scripts and to ensure cookies are only sent over HTTPS connections. Additionally, session IDs should never be passed through URLs, as they can be easily intercepted or logged. Enforcing strict session timeouts and automatic logouts after periods of inactivity can further reduce the risk of session hijacking. Finally, using HTTPS for all communications helps protect session tokens from being intercepted in transit, ensuring overall session integrity and confidentiality (Nasiketha & Athapaththu, 2025).

In the current threat landscape, session fixation attacks continue to pose a significant risk, especially as web applications increasingly rely on session-based authentication. Attackers exploit vulnerabilities in session management by forcing users to authenticate with a session ID controlled by the attacker, enabling unauthorized access to sensitive accounts and data. Despite being a well-known attack vector, many applications still fail to implement essential defenses such as regenerating session IDs after login or securing session cookies properly.

The rise of complex web applications and widespread use of third-party services increase the attack surface, making session fixation a persistent threat. Organizations that do not prioritize robust session management expose themselves to risks like data theft, account takeover, and privilege escalation, emphasizing the need for continual security improvements and awareness around session fixation mitigation (Nasiketha & Athapaththu, 2025).

Session fixation attacks are executed by an attacker who sets a known session ID and then tricks a user into authenticating with it often by sending a malicious link, embedding the ID in a URL, or manipulating cookies. Once the user logs in, the session remains valid with the fixed ID, allowing the attacker to hijack the session and gain unauthorized access to the user's account (Ghorpade & Pantridge, n.d.). The impact can be severe, attackers may steal personal data, perform fraudulent transactions, or gain administrative access, leading to data breaches, financial losses, and reputational harm for organizations. Despite advancements in security, session fixation remains a concern due to inconsistent implementation of secure session management practices, such as regenerating session IDs after login, enforcing HTTPS, and properly configuring cookies. As web applications grow in complexity and users increasingly rely on remote access, the risk persists, especially in poorly maintained applications (Jiwani et al., 2024)

While session fixation attacks are less common than high-volume exploits, they are often devastating: attackers assume legitimate user identities to steal data, perform unauthorized transactions, statistics from 2023 security research reveal that while over 98% of organizations deploy vulnerability scanning, only around 34% consider those scans highly effective, often due to root-cause issues like mismanaged sessions . This highlights that

even well-defended environments may lack proper session handling controls (Ray, 2025). In this context, stateful firewalls through dynamic packet tracking and connection-state awareness offer a foundational layer of defense: by ensuring that packets belong to valid, established sessions and rejecting any out-of-context or spoofed traffic, these firewalls can disrupt the very mechanisms that enable session fixation before the attacker even has a chance to leverage a preset session ID (Punitha & Preetha, 2025) .

In networks, Transmission Control Protocol/ Internet protocol (TCP/IP) plays a great role by administering how information moves from a computer/device to the other in a network (Mishra, 2021). The Transmission control protocol/Internet Protocol is made up of; Transmission Control Protocol (TCP) and Internet Protocol (IP) (Alicea & Alsmadi, 2021). The functions of Transmission Control Protocol are to break data into packets and to verify that all packets arrive at their destination (Liang & Kim, 2022). In addition, the function of Internet Protocol is to envelope data and assign devices a unique number or symbol known as address (Jake Frankenfield, 2020). Address distinguishes the device in a network, the IP also defines how much data can fit in a single envelop packet and ensures that network is able to read the envelop each data to be forwarded into its destination (Kumar & Deepa, 2020). There are four layers in TCP/ IP namely network, internet, transport and application layer. Network threats can affect different layers in TCP/ IP (Farooq, 2021).

Network threats include; man in the middle attacks, distributed denial of service (DDoS) (Ivanova et al., 2022) attacks, privilege escalation and SQL injection attacks (Tsiknas et al., 2021) among others. Session fixation is a web-based attack technique where an attacker tricks the user into opening a uniform resource locater (URL) (Filali, 2023) with a predefined session identifier (Prabakaran et al., 2022). Session fixation attack can allow the

attacker to take over legitimate user session and steal confidential data, transfer funds, or completely take over a user account (Rasheed & Bazeer Ahamed, 2020).

The existing Stateful Firewall models are; customized model (Yuan et al., 2020), packet inspection model (Tseng et al., 2021) and Phishcatcher model (Ahmed et al., 2023). On the other hand, the existing web application firewall models are; Cloudflare (Nadeem et al., 2023), adaptive security appliance (Abbas, n.d.), checkpoint firewall (Filali, 2023) and fortinet fortigate (Leena & Software, 2023).

Stateful firewall models can analyze a packet to detect Denial of Service attack (DOS) (Ivanova et al., 2022), Distributed Denial of Service attack and phishing attack that affect network, internet, and transport and Application layers of TCP/IP model (Galeano-Brajones et al., 2020). Customized model use string matching algorithm to filter incoming and outgoing packets. The model can analyze information in header part of a packet (Sikos, 2020). This information includes, IP address of the sender, IP address of the recipient, port number, connection status and protocol (Yuan et al., 2020). The limitation of this model is its capacity to solely scrutinize and screen inbound and outbound packets at the transport layer in TCP/IP model (Soepeno, 2023). Also, its configuration makes it unable to identify and filter Distributed Denial of Service Attacks (DDOS), a threat in the transport layer (Naqvi et al., 2023). Packet inspection model analysis the source and destination IP address, port number, protocol type and network connection status in header part of a packet (Xing, 2024). Packet inspection model also uses pattern marching algorithm to analysis the trailer part of a packet by locating, recognizing and categorizing a packet to check for errors and signature (Rasheed & Bazeer Ahamed, 2020). Therefore, packet inspection model can analyze both the header and trailer part of packer which are managed

in the internet layer and transport layer of TCP/IP model (Ponnusamy et al., 2022), though the model cannot analyze application layer of TCP/IP to filter session fixation attack in payloads part of a packet (Measures et al., 2021).

Phishcatcher (Ahmed et al., 2023) model uses random forest machine learning algorithm to analyze the header, trailer and payload of a packet to detect network threats (Akinsanya et al., 2024). Phishcatcher model can analyze information in Payload part of a packet in application layer of TCP/IP model to filter phishing attack in network (Josso et al., 2023). In addition, the model uses random forest machine learning algorithm which is a powerful and versatile machine learning algorithm (Ahmed et al., 2023). Though its performance and scalability on large datasets can be impacted by factors such as sample size, whereby random forest may not scale to large datasets (Maslennikova et al., 2023). As dataset increases, more time and computational resources are required for training and testing a model hence random forest algorithm becomes expensive (Lavrishchev et al., 2021).

Different existing web application firewall models include; adaptive security appliance (Abbas, n.d.), checkpoint firewall (Filali, 2023),fortinet fortigate (Leena & Software, 2023) and Cloudflare (Nadeem et al., 2023),

Adaptive security appliance model (Abbas, n.d.) is a web firewall model developed to address challenges associated with deploying intelligent threat detection and response mechanisms within hybrid mesh firewalls. By combining the robust capabilities of hybrid mesh firewalls with intelligent threat detection, adaptive security appliance model elevates their ability to detect, analyze, and respond to security incidents in real-time (Soepeno, 2023). Hybrid mesh firewalls used in adaptive security appliance model offers a versatile security architecture that combines the benefits of both mesh and traditional firewalls

(Aghoutane et al., 2020). Through empirical analysis and case studies, the research demonstrates the efficacy in filtering Malware and ransomware attacks (Josso et al., 2023). Adaptive security appliance model use case study and empirical analysis methodology. This model does not consider web parameters and algorithm.

Checkpoint firewall model (Filali, 2023) is another web application firewall model studied. The model states that the analysis of security access network-based firewall log in LAPAN.(González-granadillo et al., 2021), Center in the form of six messages that occur when a network passes through the firewall. By doing some analysis phases of network security access-based firewall log in LAPAN Center the model produces a record of every incident inside the firewall logs (J. Li et al., 2020). This model filters denial of service attack (Trabelsi & Zeidan, 2019a). In checkpoint firewall model observation methodology was used when collecting data. Checkpoint firewall model does not factor web Parameters, algorithm and level of accuracy when developing the model.

Fortinet fortigate model (Leena & Software, 2023) is the third web application firewall model developed. Fortinet model employs mitigation strategies against phishing attacks and session Hijacking (De Leoni & Dündar, 2020). The underlying technologies considered in the development of these strategies (Al-Heety et al., 2020). The most considered phishing vectors in fortinet model are the mitigation strategies, anti-phishing guidelines and recommendations for organizations and end-users respectively (Schwind & Asbach, 2022). The Fortinet fortigate model is considered the abilities of human users during the design and development of the mitigation strategies as only technology centric solutions, suffice to cater to the challenges posed by phishing attacks (Sikos, 2020). Fortinet fertigate employs

searching string algorithm and experimental research methodology. In fortinet fortigate web parameters and level of accuracy are not considered.

Cloudflare model (Nadeem et al., 2023) is the last web application firewall model developed, to prevent the cloud server from internal spamming attacks. When an attacker attempts to use different spamming techniques on a cloud server, the attacker is intercepted through two effective techniques, Cloudflare and K-nearest neighbors (KNN) classification (Aryeh et al., 2020). Cloudflare blocks those IP addresses that the attacker uses and prevent spamming attacks. However, the KNN classifiers determine which area the spammer belongs to (Togay et al., 2022). This web application firewall model describes various prevention algorithm such as brute force and pattern matching attacks (Alexander, 2020), for securing cloud servers are discussed. Experimental methodology is used when developing the model. In addition, deep learning algorithm is used (Carta et al., 2020). The accuracy level of Cloudflare model to filter spamming attacks is 96.3 percent. Web parameters are not used in Cloudflare model.

While the current existing stateful firewall models offer enhanced security over basic packet filtering models by tracking the state of active connections and ensuring that packets are part of valid sessions, they have significant limitations in mitigating session fixation attacks due to their operational focus on the internet and transport layers of TCP/IP model (Said et al., 2024). These firewall models are designed to monitor and manage traffic based on IP addresses, ports, and the state of TCP connections, which helps prevent unauthorized access and some types of attacks such as denial of service attacks, distributed denial of service, man in the middle and phishing attacks (Jiang et al., 2025) .

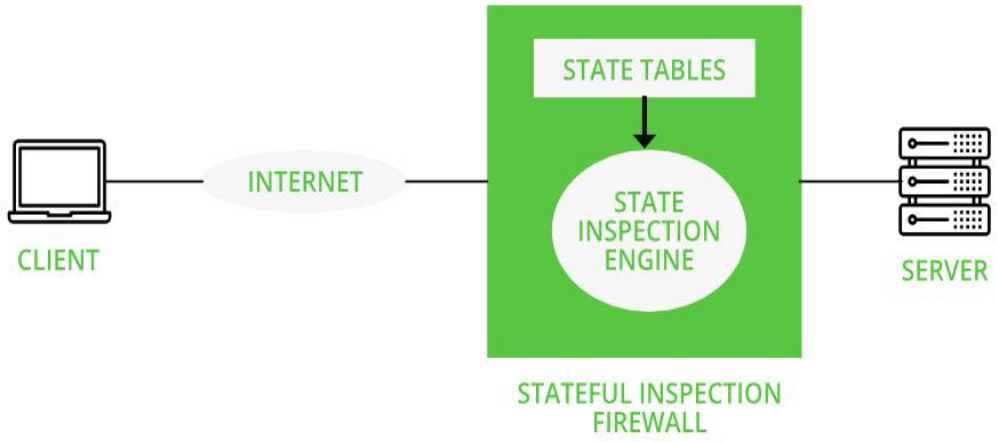
The existing stateful firewall models lack visibility into application-layer data, such as HTTP headers, cookies, and session tokens, where session fixation vulnerabilities reside. Since these attacks manipulate session identifiers within the application layer, a stateful firewall cannot detect whether a session ID has been fixed or reused maliciously (., 2024) . As a result, they cannot detect or block session fixation attacks. Therefore, there was a need , to develop stateful firewall packet analysis model, capable of inspecting network traffic by analyzing web application parameters such as session identifier, visitors identifier, event timestamp, event type and percentage count, referrer header, user agent and IP address identifying potential session fixation attempts(H. Zhang, Member, et al., 2021). By use of advanced packet inspection techniques such as signature-based and anomaly-based detection methods and maintaining session state information. The model offered high accuracy level in filtering session fixation attack compared to the existing models (Thankappan, 2024).

General data protection regulation (GDPR) and the payment card industry data security standard (PCI DSS) are regulatory frameworks and security standards that play a crucial role in shaping network security practices, especially in organizations that handle sensitive data. General data protection regulation (GDPR) enforced across the European Union and impacting any organization that processes the personal data of European Union citizens, places a strong emphasis on data protection and privacy (Miller et al., 2025). While it is often associated with data handling policies. General data protection regulation has direct implications for network security to ensure the confidentiality, integrity, and availability of personal data. This includes using firewalls, encryption, secure access controls, and real-time monitoring to prevent unauthorized access or data breaches (Kadhim, 2024).

Payment card industry data security standard (PCI DSS) is a globally accepted set of policies and procedures designed to optimize the security of credit, debit, and cash card transactions. It applies to any organization that stores, processes, or transmits cardholder data. PCI DSS has specific and detailed requirements related to network security, including the installation and maintenance of firewalls to protect cardholder data, secure configuration of network devices, and use of encryption over public networks, and regular monitoring and testing of security systems (Vikstr, 2025).

Figure 1. 1

Stateful firewall model



Source: *Pasham, 2024*

Figure 1.1 shows how Stateful firewall model works, while a client is accessing resource from the server over the internet (Sethi et al., 2022). Firewall model capture client information such as source address, destination address, port number, connection status and protocol and store them in a state table (Subakti, 2022). When a new packet arrives at the firewall model, the filtering mechanism first checks to determine if the information of a new

packet resembles the information stored in state table, and if not, the packet is blocked (Alarood & Ibrahim, 2023)

1.3 Problem Statement

Currently, customized model (Yuan et al., 2020), packet Inspection model (Tseng et al., 2021) and Phishcatcher model (Ahmed et al., 2023) are stateful firewall models that were studied. Customized model can analyze information in header part of a packet (Sikos, 2020). This information includes, IP address of the sender, IP address of the recipient, port number, connection status and protocol, to detect Denial of Service attack (DOS). Packet inspection model analysis the source and destination IP address, port number, protocol type and network connection status in header part of a packet (Xing, 2024). Packet inspection model can filter distributed denial of service attack (DDOS).

Phishcatcher model can analyze information in Payload part of a packet in application layer of TCP/IP model to filter phishing attack in network (Josso et al., 2023). Different existing web application firewall models include; adaptive security appliance (Abbas, n.d.), checkpoint firewall (Filali, 2023),fortinet fortigate (Leena & Software, 2023) and Cloudflare (Nadeem et al., 2023)

Through the use of empirical analysis and case studies, adaptive security appliance model demonstrates the efficiency in filtering Malware and ransomware attacks (Josso et al., 2023). By doing analysis phases of network security access-based firewall log in LAPAN Center to produce a record of every incident inside the firewall logs checkpoint firewall model can filter denial of service attacks. Fortinet model use phishing vectors as a mitigation strategy, to filter phishing attacks.

Cloudflare model blocks IP addresses that the attacker uses, Cloudflare and K-nearest neighbors (KNN) classification techniques. Cloudflare model filter spamming attacks.

Therefore, after conducting analysis on the current existing stateful firewall models it was noted that the existing models do not filter session fixation attack.

The objective of a stateful firewall packet analysis model was to monitor and evaluate the state of network connections to detect and filter session fixation attacks. This involved tracking active sessions, inspecting packets for abnormal patterns such as fixed or reused session IDs, and ensuring session tokens were properly regenerated after authentication. The model also linked session activities with legitimate user authentication events and enforced strict session lifecycle policies. If session fixation attacks were not addressed, attackers could hijack user sessions, leading to unauthorized access, data breaches, identity theft, financial loss, and non-compliance with security standards. Failure to mitigate such threats undermined user trust, and organizational security position.

The developed stateful firewall packet analysis model, is capable of filtering session fixation attack by inspecting network traffic and analyzing web application parameters such as session identifier, visitor's identifier, event timestamp, event type, percentage count, referrer header, user agent and IP address (Alsaqour et al., 2021). Further, the study use advanced packet inspection techniques such as signature-based and anomaly-based detection methods to maintain session state information (Moradi et al., 2021), Finally, the model offers high accuracy level in filtering session fixation attack compared to the existing stateful firewall and web application firewall models (Thankappan, 2024).

Security metrics such as detection rate, false positive rate, and false negative rate were used to measure how accurately the model identified true attacks while minimizing incorrect classifications.

1.4 Research Gap

Unlike the existing stateful firewall models that only inspect headers or payloads, stateful firewall packet analysis model offered Full-Packet Inspection capability by analyzing header, payload, and trailer, ensuring deeper threat detection across all packet components. Session Fixation-Specific Filtering Logic Introduces a targeted mechanism to detect reused or injected Session IDs, which are often overlooked in by existing firewall models. Stateful firewall packet analysis model adjusted session timeout durations based on traffic behavior, reducing the risk of hijacked idle sessions remaining. The model also offered session tracking enhancing session table architecture to monitor session ID value**,** initiation timestamp**,** and timeout duration key variables for detecting session fixation attempts.

1.5 Objectives

1.5.1 Main objective

The study aims to develop a stateful firewall packet analysis model that can effectively filter and mitigate session fixation attacks

1.5.2 Specific objectives

i. To assess existing research on firewall models currently in use
ii. To design a stateful firewall packet analysis model for filtering and mitigating session fixation attacks
iii. To validate the accuracy of the designed model for filtering and mitigating session fixation attacks

1.6 Research Questions

i. How to assess existing research on firewall models currently in use

ii. How to design a stateful firewall packet analysis model for filtering and mitigating session fixation attacks?

iii. How to validate the accuracy of the designed model for filtering and mitigating session fixation attacks?

1.7 Significance of the Study

Due to the ever-changing sophistication on network security (Fadziso, 2023), organizations are overwhelmed in dealing with network threats, it is necessary to take measures through implementation of a firewall to safeguard network from illegal entry, alteration of information and improper disclosure of information which emerge due to network threats (L. Chen, 2021).

Network administrators are responsible for maintaining the security and functionality of organizational networks (Demertzi et al., 2023). Therefore, the result of this study helps the system administrator to prevent session fixation attacks, thereby safeguarding network resources and data (Lyu et al., 2024). In addition, Implementing advanced security measures like the stateful firewall packet analysis model helps in reducing the risk of data breaches and maintaining trust with customers and stakeholders in businesses and organizations (Naga & Keerthi, 2023) . Finally, this study helps organizations comply with regulations by providing effective defenses against session fixation attack hence, the implemented security measures protect sensitive information (De Leoni & Dündar, 2020).

The results of the stateful firewall packet analysis model for filtering session fixation attacks significantly influenced the design of application-level firewalls by introducing enhanced

session awareness and dynamic packet inspection capabilities (Leppänen, 2024). This awareness influenced the design of application level firewalls by driving the integration of more advanced features such as deep packet inspection, session token validation, and application specific logic analysis (Foreman et al., 2024). Such enhancement was crucial because session fixation attacks exploit vulnerabilities within application protocols often hidden from network level monitoring, making it essential for firewalls to understand and interpret application data to effectively secure user sessions.

1.8 Scope of the Study

The general purpose of this research was to develop a stateful firewall packet analysis model that will filter session fixation attack (Subakti, 2022), the study was restricted to monitoring and analyzing session related data in web applications. These include parameters such as; session identifier, visitors identifier, event timestamp, event type and percentage count (Rasheed & Bazeer Ahamed, 2020). Analyzing session parameters is crucial in filtering session fixation attack, by providing insights into session activities, user behavior, and potential anomalies (Muzammil et al., 2024). Analyzing these parameters ensures robust and effective approach in identifying and filtering session fixation attack (Song et al., 2022).

1.9 Limitations of the Study

The cybersecurity landscape is constantly evolving, with new attack techniques, vulnerabilities, and defense mechanisms emerging regularly (Ozkan-okay et al., 2023). Research findings on stateful firewall packet analysis model, may become outdated quickly as attackers adapt their tactics and security measure evolve (Akinsanya et al., 2024). A static model may struggle to keep up with emerging threats without regular updates (Filali, 2023). Therefore, to mitigate this limitation, there is a need to conduct regular penetration testing

and simulations of session fixation attack (Asituha, 2024). to validate the capabilities of the model to detect session fixation attack and identify areas of improvement (Kamrul et al., 2023a).

1.10 Contribution of the Study

Contribution of this study includes; dynamic rule generation, Session tracking enforcement and Integration with Security Ecosystem

1.10.1 Dynamic rule generation

The stateful firewall packet analysis model can dynamically generate and update filtering rules based on real-time analysis of packet payloads and session attributes (Alsaqour et al., 2021). This adaptability allows the firewall to respond to emerging threats and variations in attack patterns, including those associated with session fixation (Son & Lee, 2022).

1.10.2 Session tracking enforcement

Through stateful inspection, the stateful firewall packet analysis model tracked the lifecycle of application- layer session and enforce security policies to prevent session fixation attack (Technology, 2022). This includes validating session identifiers, monitoring session attributes for consistency and enforcing session management best practices (Peinado Gomez et al., 2021).

1.10.3 Integration with security ecosystem

The Stateful Firewall packet analysis model can integrate with existing security ecosystem components (Assadpour et al., 2022), such as intrusion detection/prevention systems (IDS), Web application Firewalls (WAF), and Security information and event management (SIEM) systems (Mauthner, 2021). This integration enhances overall security posture by correlating

session fixation alerts with other security events and facilitating coordinated incident response (Sequeiros et al., 2020).

1.11 Assumption of the Study

The assumptions made in this study is that the scenarios and methodologies used to simulate and study session fixation attack accurately represent real-world scenarios (H. Zhang, Member, et al., 2021) . This assumption was crucial for drawing meaningful conclusions about the effectiveness of countermeasures or detection techniques (Nife & Kotulski, 2020).

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

This chapter present a review of the literature related to the study. The key areas include the concept relationship between existing models and developed model (Assadpour et al., 2022) , model accuracy in relation to TCP/IP model, various stateful firewall models developed and the type of threats that they analyze. The chapter describes algorithms used in different stateful firewall models, strength and weaknesses of the current models and metrics for measuring effectiveness of packet analysis models and conceptual framework (Al-Heety et al., 2020).

2.2 Current Stateful Firewall Models

The existing stateful firewall models, includes customized model (Yuan et al., 2020), packet inspection model (Tseng et al., 2021), and Phishcatcher model (Ahmed et al., 2023). This area describes which layer of TCP/IP model addressed by each stateful model and threats each model can analyze.

The existing stateful firewall models encompass a variety of approaches designed to enhance network security through contextual packet analysis, including customized packet inspection models and specialized frameworks like the Phishcatcher model. Customized packet inspection models extend traditional stateful firewalls by tailoring inspection rules mechanisms to the specific needs of an organization or application, allowing for more precise filtering and detection of complex threats such as denial of service attacks. Packet inspection model incorporates deep packet inspection and behavior-based analytics to monitor Source IP Address and Destination IP Address and Port (Kannan, 2024). The Phishcatcher model, on the other hand, is a targeted approach initially developed to identify

phishing attempts by analyzing the destination IP addresses and domains for known malicious or spoofed URLs that phishing emails or websites often use. By integrating heuristic and signature-based techniques, Phishcatcher adds an additional layer of security focused on protocol behavior and content consistency to detect attempts to deliver malicious links or scripts that aim to steal credentials or deploy malware (Taylor et al., 2024). Together, these models illustrated the evolving capabilities of stateful firewalls, combining flexible customization, advanced packet inspection, and specialized threat detection to address a broad spectrum of source and destination IP based vulnerabilities in modern network environments (Polonio et al., 2024).

Session fixation is a type of web attack in which an attacker assigns a predetermined session ID to a legitimate user before they log in. After the user authenticates using that session ID, the attacker—already aware of the ID—can take over the session and access the user's account without permission. This vulnerability occurs in web applications that fail to issue a new session ID after login, enabling the attacker to stay connected to the authenticated session (Lotto et al., 2024).

The Current methodologies for packet analysis combine a range of techniques to effectively monitor, inspect, and secure network traffic. Traditional approaches like deep packet inspection (DPI) analyze both packet headers and payloads to detect application-level threats, while flow-based analysis focuses on summarizing traffic patterns for performance monitoring and anomaly detection. Signature-based detection uses known threat patterns to identify malicious traffic, whereas heuristic and behavior-based analysis detects deviations from normal activity, enabling the discovery of unknown or zero-day threats. Protocol decoding examines the structure and behavior of specific protocols to identify misuse or

errors, and encrypted traffic analysis (ETA) inspects metadata and TLS handshakes without decrypting content to preserve privacy while flagging anomalies (Wiles et al., 2024). Emerging machine learning-based techniques further enhance threat detection by learning from network behavior over time, while stateful inspection remains foundational by tracking the state of active connections to apply context-aware filtering. Together, these methodologies offer a layered, adaptive approach to securing modern network environments.

2.2.1 Customized model

Customized model (Yuan et al., 2020) analyzes the internet layer which is the second layer of the TCP/IP. The model can filter incoming and outgoing packets on internet layer of the TCP/IP model using string matching algorithms (De Leoni & Dündar, 2020). Customized model analyzes information in header part of a packet such as; IP address of the sender, IP address of the recipient, port number, connection status and protocol (Al-Heety et al., 2020)**.** The accuracy level of customized model in filtering Denial of Service attacks that affect internet layer of TCP/IP model is 90.8 percent. Parameters used in this model are as follows; packet rate, traffic volume, memory usage and band width consumption (Muthukumar et al., 2021). The weakness of customized model is that it cannot filter distributed denial of service attack (DDOS) (Vladimirov et al., 2023). The model is developed by use of string-matching algorithm designed to detect specific pattern within a data stream. String matching algorithms are only effective in identifying known signature(Afzal et al., 2021) , they are not able to detect novel attackers which cannot march predefined patterns, and hence attackers can easily evade detection by slightly modifying their attack payloads, rendering string matching approach ineffectiveness (Song et al., 2022).

2.2.2 Packet inspection model

Packet inspection model (Tseng et al., 2021) can analyze both the header and trailer part of packer which are managed in the internet and transport layers of TCP/IP model. A header of a packet contains the source and destination IP address, port number, protocol type and network connection status in header part of a packet, the model locate, recognize and categorize a packet to check for errors and signature (Kamrul et al., 2023b). The packet Inspection model developed employs a pattern matching algorithm method to evaluate the internet and transport layers within the TCP/IP models and is able to examine session information in the header section of a packet. The model can perform tasks such as error checking, and signature verification (Farmer, n.d.). The accuracy level of Packet inspection model in filtering distribution denial of serve attack is 92.4 percent to filter Distribution denial of serve attack (Tseng et al., 2021). Parameters used in this model include; Traffic variability, IP address and network congestion. Nevertheless, the capability of this model is constrained, as it lacks the capacity to inspect an entire packet due to its filtering mechanism which cannot identify threats within the payload in the application layer of the TCP/IP model (Tyagi, 2020).

2.2.3 Phishcatcher model

Phishcatcher model (Ahmed et al., 2023) can analyze the header, trailer and payloads part of packer which are managed in the internet, transport and application layers of TCP/IP model filter information in header, trailer and Payload part of a packet in application layer of TCP/IP model (Vladimirov et al., 2023). The accuracy level of Phishcatcher model in filtering phishing attack is 94.1 percent (Ahmed et al., 2023). Parameters used in this model are as follows; sender Information, response time and image analysis. However, the model

uses random forest machine learning algorithm, which combines several decision trees (L. Chen, 2021), thus the performance and scalability of the model on large datasets are affected. Random forest cannot scale to large datasets, as dataset increases, more time and computational resources are required for training and testing a model therefore, developing a model using a random forest algorithm becomes expensive (Brophy & Lowd, 2021) .

2.3 Web Application Firewall Models

Web Application Firewalls (WAFs) are designed to protect web applications by filtering and monitoring traffic between a web application and the internet (Machado et al., 2020) . They can help defend against various types of attacks, including SQL injection and session fixation (Sethi et al., 2022) . Here's an overview of different WAF models, their features, and how they work:

2.3.1 Adaptive security appliance model

Adaptive security appliance model (Abbas, n.d.) is a web firewall model developed to address challenges associated with deploying intelligent threat detection and response mechanisms within hybrid mesh firewalls. This model is able to detect, analyze, and respond to security incidents in real-time (Soepeno, 2023). Through empirical analysis and case studies (Aghoutane et al., 2020), demonstrated the efficacy in filtering Malware and ransomware attacks (Josso et al., 2023). Adaptive security appliance model use case study and empirical analysis methodology. Adaptive security appliance model does not consider factors such as web application parameters and algorithm (Khan, 2023).

2.3 2 Checkpoint firewall model

Checkpoint firewall model (Filali, 2023) is a web firewall model The study states that the analysis of security access network based firewall log (J. Li et al., 2020), in LAPAN Center

in the form of six messages that occur when a network passes through the firewall. By doing some analysis phases of network security access-based firewall log in LAPAN Center, the model produces a record of every incident inside the firewall logs. This model filters denial of service attack (González-granadillo et al., 2021), In checkpoint firewall model observation methodology was used when collecting data. The weakness of checkpoint firewall model is that accuracy level, algorithm and web parameters are not considered when developing the model for filtering denial of service (Trabelsi & Zeidan, 2019a).

2.3.3 Fortinet fortigate model

Fortinet fortigate model (Leena & Software, 2023) is a web firewall model. Fortinet model employs mitigation strategies against phishing attacks and session Hijacking (De Leoni & Dündar, 2020), and the underlying technologies considered in the development of these strategies. The most considered phishing vectors in fortinet model are the mitigation strategies (Al-Heety et al., 2020). Anti-phishing guidelines and recommendations for organizations and end-users respectively (Schwind & Asbach, 2022). The open issues that exist in the state-of-the-art in. The Fortinet fortigate model considered the abilities of human users during the design and development of the mitigation strategies as only technology centric solutions suffice to cater to the challenges posed by phishing attacks and session Hijacking (Sikos, 2020). Fortinet fortigate employs searching string algorithm and experimental research methodology. Though the model does not analyze web parameters when developing the model for filtering denial of service (Trabelsi & Zeidan, 2019b). In addition, the accuracy level when filtering attacks is not considered.

2.3.4 Cloudflare model

Cloudflare model (Nadeem et al., 2023) is a web firewall model developed tool to prevent the cloud server from internal spamming attacks. When an attacker attempts to use different spamming techniques on a cloud server, the attacker is intercepted through two effective techniques: Cloudflare and K-nearest neighbors (KNN) classification (Aryeh et al., 2020). Cloudflare blocks those IP addresses that the attacker uses and prevent spamming attacks. However, the KNN classifiers determine which area the spammer belongs to (Togay et al., 2022). This web firewall model describes various prevention techniques and algorithm such as brute force and pattern matching attacks (Alexander, 2020), for securing cloud servers are discussed. Experimental methodology is used when developing the model. In addition, deep learning algorithm is used (Carta et al., 2020). The accuracy level of Cloudflare model to filter spamming attacks is 96.3 percent. Web application parameters are not considered.

2.4 Relationship Between Existing Stateful Firewall Models and Developed Model

Previous studies which includes; customized, packet inspection and Phishcatcher that focus on filtering Denial of Service (DoS), Distributed Denial of Service (DDoS), Man-in-the-Middle (MITM), and phishing attacks was relevant to the development of a stateful firewall packet analysis model for filtering session fixation attacks, as they shared common principles in traffic analysis, anomaly detection, and session monitoring (X. Wang et al., 2024). Techniques used in mitigating DoS and DDoS attacks such as traffic pattern recognition, rate limiting, and connection state tracking was adapted to detect unusual session behaviors indicative of fixation attempts. Similarly, the study on Phishcatcher that filters phishing attacks emphasizes the importance of secure session handling, token integrity, and authentication correlation, which directly aligns with the need to validate

session tokens and user identities in a fixation attack (Hoepman & Emmen, 2024). These studies often employ deep packet inspection, behavioral analysis, and real-time alerting methods that was also integrated into the stateful firewall packet analysis model.

2.5 Strengths and Weaknesses of Previous Studies

Existing research on mitigating DoS, DDoS, Man-in-the-Middle (MITM), and phishing attacks demonstrates strengths in analyzing traffic patterns, detecting anomalies, and responding to threats in real time, which can inform the development of stateful firewall packet analysis models targeting session fixation attacks (Dornala & Senthilkumar, 2025). These studies commonly used deep packet inspection and behavior-based methods to identify malicious activities within network traffic. However, these studies often focus on identity based threats rather than session token misuse, so they does not analyze session token lifecycle, which involves examining how a session token is created, used and maintained (Phanireddy, 2025). The studies also does not manage essential for detecting session fixation whereby, session ID remains unchanged before and after user login, checking if session tokens are exposed in URLs or query strings, and verifying that session IDs are properly regenerated upon authentication (Kolenbrander et al., 2024).

Therefore, while these studies provide useful detection frameworks and security principles in detecting network attacks, they required refinement and deeper application-layer focus to effectively filter session fixation threats (Alwaheidi, 2023). This study addressed the gap in filtering session fixation attacks, which was identified after conducting literature review in existing stateful firewall models. By integrating effective session fixation detection into stateful firewall model, attackers were prevented from hijacking authenticated sessions, thereby reducing the risk of identity theft and data breaches (Cross & Holt, 2025).

This enhancement complemented existing defenses against other network threats such as session hijacking and cross site request forgery and promoted a more complete approach that combined network-level and application-level security. Ultimately, filling this gap improves the overall resilience of network infrastructures, helping to safeguard sensitive information and maintain user trust in digital environments (Oluwabunmi Layode et al., 2024).

The stateful firewall packet analysis model for filtering session fixation attacks was supported by several relevant theoretical frameworks, including stateful inspection theory, session management principles, and anomaly detection. Stateful inspection theory provided the foundation for tracking and maintaining the state and context of network connections, enabling the firewall to analyze packet flows rather than isolated packets (Lumazine et al., 2024).

Session management principles informed the model's approach to monitoring session lifecycle events such as token generation, validation, and renewal ensuring that session identifiers are securely managed and regenerated after authentication (Austria et al., 2024). Additionally, anomaly detection frameworks, often rooted in statistical and behavior-based theories, guided the identification of deviations from normal session behaviors that indicated fixation attempts. By combining these frameworks, the model achieved a comprehensive, context aware mechanism that effectively detected and filtered session fixation attacks at the application layers (Merget et al., n.d.).

2.6 Model Accuracy in Relation to TCP/IP Model

To validate the accuracy of stateful firewall packet analysis model in relation to the TCP/IP model**,** testing and evaluation was aligned with the layers, where session fixation attacks occur. Session fixation attacks usually target the application layer (Layer 4) and exploit

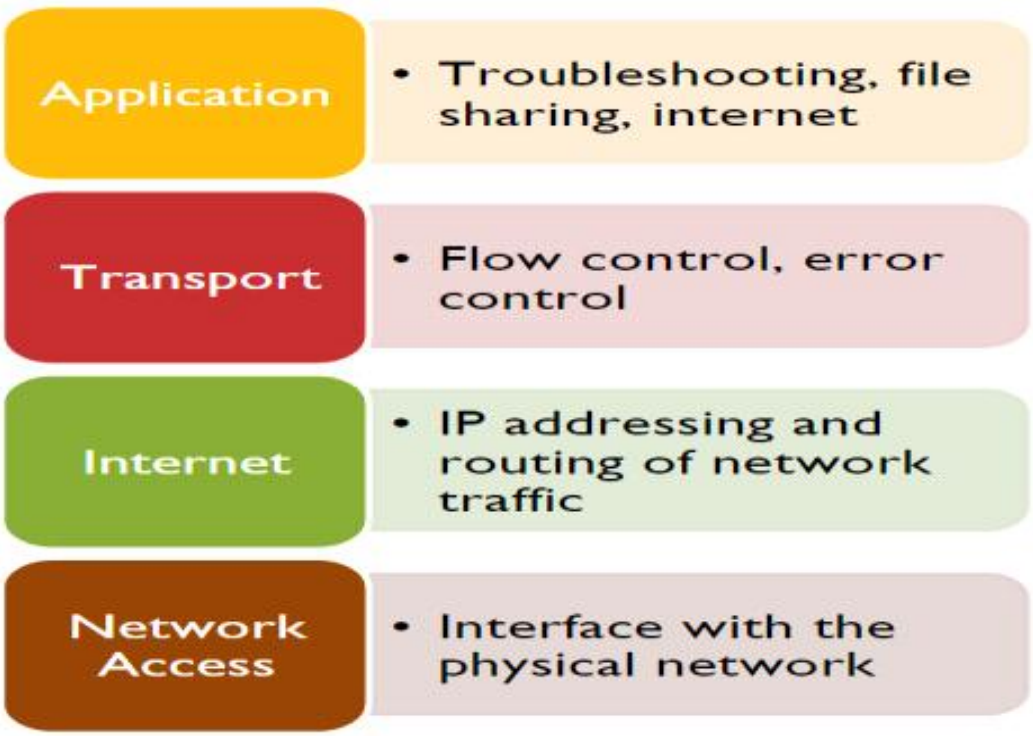
weaknesses in session management like hijacking session IDs via cookies or uniform resource locators (Kannadhasan & Nagarajan, 2024). To detect and block these attacks, the model analyzed packets across multiple TCP/IP layers. The layers include; internet, transport and application layer.

2.6.1 TCP/IP model

Internet protocol/Transmission Control Protocol TCP/IP model is solid foundation for all the communication task on the internet (W. Li et al., 2020). It is a collection of protocols, or rules that govern the way data travels from one computer/device to another across the networks (Suherman et al., 2021). The TCP/IP has two major components (Song et al., 2022); Internet Protocol and Transmission Control Protocol. The function of Internet Protocol is to envelope data and assign devices a unique number or symbol known as address that distinguishes the device in a network (Vladimirov et al., 2023), the IP also defines how much data can fit in a single envelop/ packet and ensures that network is able to read the envelop each data to be forwarded into its destination (Rico et al., 2021) .In addition, the functions of Transmission Control Protocol are to break data into packets and to verify that all packets arrives at their destination (Machora, 2024). There are four layers in TCP/IP namely network, internet, transport and application layer. Network threats can affect different layers in TCP/IP (Tyagi, 2020).

Figure 2. 1

TCP/ IP layers and their functions



Source: *Lazar et al, 2023*

The figure above shows TCP/ IP layers and their functions. The lowest level is the network access layer, which handles the physical transmission of data over network hardware and manages how devices on the same local network communicate. Above the network is internet layer, responsible for routing packets across different networks using the IP protocol (Jadav et al., 2025), ensuring that data reaches the correct destination by assigning IP addresses and managing packet forwarding. The transport is the third layer next, that provides end-to-end communication services such as error checking, flow control, and reliable data delivery through protocols like transmission control protocol (TCP) and User Datagram Protocol (UDP) (Simpson et al., 2024). Finally, at the top is the Application Layer**,** which includes protocols and services that directly interact with user applications, such as

HTTP for web browsing, FTP for file transfers, and SMTP for email. Together, these layers encapsulate and decapsulate data, enabling efficient and reliable communication across diverse network environments (Buitrago López et al., 2024).

2.6.2 Internet layer

Internet layer is the second layer of the TCP/IP model (H. Zhang, Zhang, et al., 2021). The main function of the internet layer is sending the data packets to their destination (Aryeh et al., 2020). In the Internet layer the model analyzed packet headers to track IP addresses, protocols, and connection states, helping identify unusual or suspicious patterns that indicate the early stages of a session fixation attack. While it cannot directly inspect session tokens or HTTP cookies—since those reside at the application layer, it contributed indirectly by monitoring for anomalies such as repeated connections from the same IP, unexpected connection initiations, or inconsistent traffic flows that deviated from normal session behavior. This analysis supports broader intrusion detection efforts, but to accurately filter session fixation attacks (Burns, 2025), deeper inspection at higher layers—such as with a web application firewall (WAF) or a firewall with deep packet inspection (DPI)—is necessary, as these tools examine and validate session management mechanisms directly.

2.6.3 Transport layer

Transport layer is responsible for transporting data, dividing it into packets and handling transmission errors. Threats that attack transport layer includes; denial of service (DoS) and distributed denial of service attacks (DDoS) (Sambangi & Gondi, 2020). Denial of service (DoS) is an attack that shut down a computer or network, making it inaccessible to its intended users, while distributed denial of service attacks (DDoS) is a type of cyber-attack whereby attackers utilize a large network of remote personal computers called botnets, to

overwhelm another systems connection or processor, causing it to deny service to the legitimate traffic its receiving (Lee et al., 2024).

Though at transport layer the model cannot block session fixation attack, but the model analyzed TCP/UDP packet headers, including source and destination ports, sequence numbers, flags (Synchronization, acknowledgement), and connection states to maintain an awareness of active sessions. This allowed the model to detect irregularities in session behavior, such as multiple simultaneous connections using the same port combinations or suspicious attempts to hijack or replay existing sessions (Akhmedov, 2020).

While it still cannot inspect application-layer data like session IDs or cookies directly, analyzing transport-layer patterns helped validate the integrity of session flows and detect anomalies indicative of session fixation tactics, such as unsolicited attempts to reuse established connections. However, full validation and filtering of session fixation attacks require deeper inspection beyond Layer 4, involving application-layer visibility (Salas, 2024).

2.6.4 Application layer

Application layer is the highest abstraction layer of TCP/IP model (Tyagi, 2020), that provides the interfaces and protocols needed by the users (Rico et al., 2021). The function of application layer is to interface and provide services for application processes. It also contains standard and native applications such as Telnet, simple mail transfer protocol (SMTP) and file transfer protocol (FTP)(Alexander, 2020) . For these reasons, this layer can be vulnerable to phishing scammers can pass themselves off as a legitimate contact trying to steal information and session fixation a valid user session identity is exploited to gain unauthorized access to the system (Rahouti et al., 2022).

At the application layer, the model equipped with Deep Packet Inspection (DPI) analyzed the contents of HTTP requests and responses, including headers, cookies, and session tokens (Muriithi et al., 2025) . This allowed the model to inspect session identifiers and detect suspicious patterns such as fixed or predictable session IDs, session tokens passed via URLs, or repeated use of the same session ID across different client IPs.

By maintaining session context and correlating it with user behavior and request patterns, the model identified anomalies that suggest a session fixation attack, such as a login request using an attacker-supplied session ID that persists post-authentication (Z. Zhang, Zhang, Zhang, et al., 2024). This deep visibility enabled accurate filtering and enforcement of secure session management practices, making application-layer analysis essential for validating and mitigating session fixation threats effectively.

2.7 Theoretical Framework

A theoretical framework for stateful firewall models provides a theoretical basis and structure for understanding the underlying principles, concepts and assumptions that guide design, implementation, and evaluation of stateful systems (Alanazi et al., 2023). The theoretical frameworks and concepts that are relevant to stateful firewall are; Stateful Inspection (Machora, 2024), Session Management and Authentication (Song et al., 2022), Network Protocol Analysis (Grazia et al., 2021), Cybersecurity Frameworks (Malik et al., 2020), Cybersecurity Frameworks (De Leoni & Dündar, 2020) and Risk Management (Carta et al., 2020).

2.7.1 Stateful inspection

Stateful inspection is the foundational theory behind stateful firewalls, enabling firewalls to monitor the state of active connections by tracking packet flow and context across multiple

layers (Thesis, 2024). In the context of filtering session fixation attacks, stateful inspection allowed the model to maintain awareness of ongoing communication sessions, validate packet legitimacy based on prior exchanges, and identify anomalous behaviors such as unexpected reuse of session identifiers (Rumpold, 2024). By keeping track of session states—like TCP handshakes and termination. The model distinguished between legitimate traffic and suspicious attempts to hijack or inject pre-defined sessions.

2.7.2 Session management and authentication:

Effective session fixation filtering requires a deep understanding of session management and authentication principles. These include secure generation, transmission, and regeneration of session tokens during authentication workflows. (Song et al., 2022). . A stateful firewall with application-layer visibility can analyze HTTP payloads to detect improper session handling, such as session tokens not being regenerated after login or tokens passed through insecure vectors like URLs. Integrating authentication validation into firewall logic enables it to identify when session fixation is being attempted by correlating session behavior before and after authentication events.

2.7.3 Attack vector analysis

Attack vector analysis provides the strategic lens through which the firewall interprets threat behaviors. Session fixation is a stealthy attack vector that relies on exploiting weak session controls (Grazia et al., 2021). By analyzing known and emerging methods used to exploit session management flaws, the firewall's detection engine can be fine-tuned to recognize patterns indicative of fixation attempts—such as repeated session ID injection attempts, session reuse across diverse source IPs, or HTTP requests bearing pre-defined session

tokens. Incorporating attack vector analysis enhances the model's predictive and preventative capabilities

2.7.4 Network protocol analysis

Understanding the structure and behavior of network protocols is essential for accurately filtering threats at various layers of TCP/IP model (Faris et al., 2023). Network protocol analysis enabled the firewall to dissect packets at the Internet and Transport layers, assessing aspects like TCP flags, port behavior, and IP anomalies that signal unauthorized session activity. Moreover, at the application layer, protocols such as HTTP and HTTPS was analyzed to parse headers and cookies (Klein & Johns, 2024). This allowed the firewall model to extract session-related data and match it against expected patterns, facilitating more accurate identification of malicious session manipulation.

2.7.5 Cybersecurity frameworks

Incorporating standardized cybersecurity frameworks, such as, ISO/IEC 27001, or the MITRE ATT framework, strengthens the firewall model’s alignment with best practices (Durante et al. 2021). Cybersecurity frameworks guided in the categorization of session fixation as part of session hijacking or web application attacks, and recommended specific controls such as secure session handling, access control policies, and anomaly detection. Applying such frameworks helped define measurable objectives for the firewall model, ensuring it supports layered defense strategies and aligns with organizational risk and compliance requirements.

2.7.6 Machine learning

Machine learning enhances the analytical power of stateful firewalls by enabling dynamic detection of session fixation patterns that may not be caught by static rules. (Ozkan-okay et

al., 2023). By training models on labeled network traffic, the firewall was able to learn and identify subtle anomalies such as unusual session token reuse patterns, abnormal user-agent switching, or IP diversity in session usage. Over time, machine learning models could adapt to evolving attack techniques, improving the model accuracy and reducing false positives. This predictive capability was crucial for identifying fixation attempts that deviate from known signatures.

2.7.7 Risk management

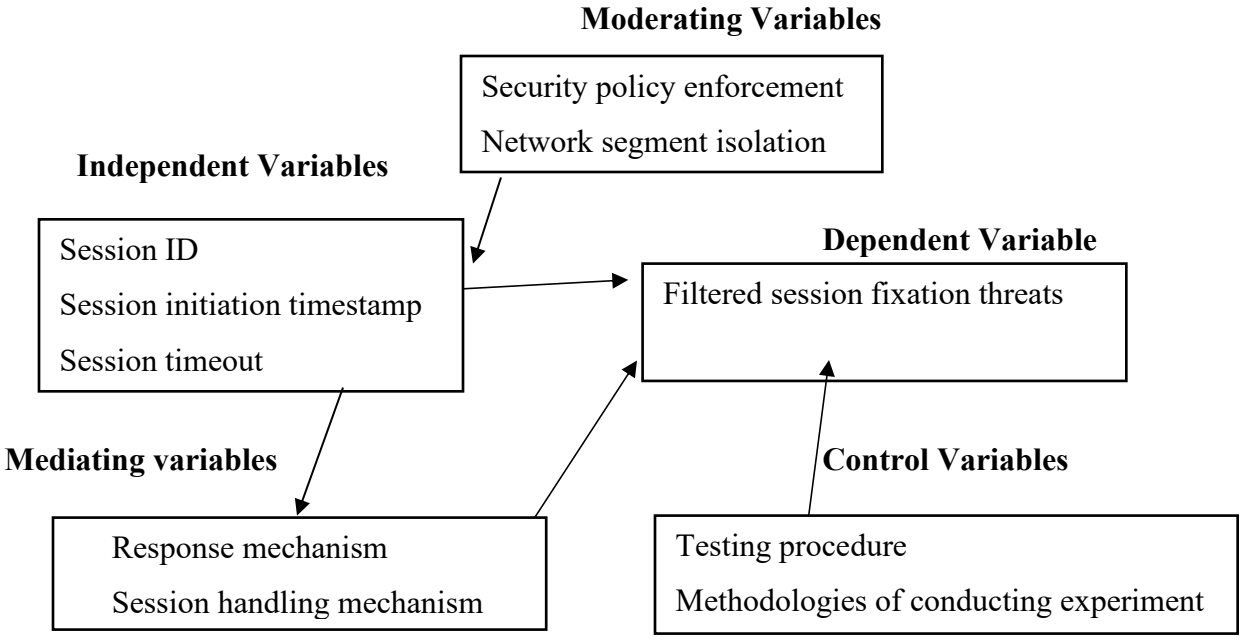
Risk management principles helps prioritize firewall inspection and response strategies based on the severity and likelihood of session fixation threats (Arogundade, 2023). Not all session anomalies present equal risk; therefore, integrating a risk-based approach allowed the firewall model to focus resources on high-impact scenarios, such as fixation attempts targeting administrative accounts or systems handling sensitive data. This ensured the firewall model not only identifies potential attacks but also responds in proportion to their assessed risk, supporting broader organizational risk mitigation efforts.

2.8 Conceptual Framework

The conceptual framework provides a structured approach for developing, implementing and evaluating a stateful firewall packet analysis model (J. Li et al., 2020), to effectively detect and threats in network (Peinado Gomez et al., 2021). The conceptual framework in the context of developing a stateful firewall packet analysis model, will integrate various components, techniques and considerations to achieve the stated objective of detect and mitigate session fixation attack within the application layer of the TCP/IP model (Taherdoost & Corporation, 2020).

Figure 2. 2

Conceptual framework



Source: *Researcher, 2024*

The figure above illustrate the relationship of variables in a conceptual framework. Dependent variable refers to the main result that is evaluated by the model (Arogundade, 2023), whereby in this research the dependent variable is the filtered session fixation attack. This is the degree to which the stateful firewall packet analysis model correctly identifies and filter session fixation attack (Measures et al., 2021) Dependent variable measures the performance of the model in distinguishing legitimate sessions from those that are being exploited through session fixation techniques (Maslennikova et al., 2023) . In addition, the independent variables are factors that can be manipulated to determine their effect on the dependent variable (Direr, 2020).

In this study, independent variables are the session ID **,** session initiation timestamp**,** and session timeout duration (Teng et al., 2022). The session ID is used to detect session

fixation. Session ID is the core identifier for a user's session and is the primary target in session fixation attack. If a session identifier was reused across different sources, it indicated that there was session fixation attack. Session initiation timestamp marked the start time of a session and was crucial for tracking session lifecycle. The role of Session initiation timestamp in the model was to associate session life span of a session with expected behavior.

Session initiation timestamp was used to identify sessions that were initiated internally but reused externally, the model was able to compare timestamp to detect premature reuse. Session timeout duration is an independent variable that defined how long a session remained valid before being eliminated. The role of session time out in the model was to prevent stale sessions from being exploited. The model detected whether timeout duration was adaptive based on traffic patterns, to enhance security by shortening timeout for suspicious/idle session, extending timeout for legitimate, active sessions to reduce false positives and to reacting to anomalies like reused session IDs or inconsistent timestamps (Kim et al., 2020).

Another variable that influences the dependent variable is the control variables, these are factors that are kept controlled to prevent them from influencing the dependent variable (Mendoza et al., 2020). In this research control variables such as the testing procedures and methodologies are controlled to ensure that the experiment accuracy measures the performance of the model, and ensures results are not skewed by the uncontrolled factors (Barbierato et al., 2023). Testing procedures involves the systematic approach to conduct experiments to evaluate the model accuracy in filtering session fixation attack (Gupta et al., 2020). By enforcing standardized testing conditions ensured that all test were conducted

under consistent conditions, this was achieved by use of same network environment, hardware, and software configurations to avoid inconsistence result (Parry et al., 2021). Also, methodologies of conducting experiment involves the approaches and techniques used to design and carry out the experiment. Standardized experiment design ensures that experimental design is consistent across tests (Voelkl et al., 2020). This includes maintaining the same parameters for stateful firewall packet analysis, detection algorithms, and session state tracking methods (Togay et al., 2022).

Moderating variables is another variable that was considered when developing conceptual framework (Hamad et al., 2020). Moderating variables refers to factors that influence the relationship between independent variables and dependent variable (Purnama et al., 2021), the moderating variable in this research include; security policy enforcement and Network segment isolation. These variables can either enhance or weaken the accuracy of the model to filter session fixation attack (Gupta et al., 2020). Security policy enforcement involves applying rules and policies to manage and restrict network traffic based on predefined security criteria (Paananen et al., 2020). Enforcing strict security policies helped in filtering benign traffic, allowing the stateful firewall packet analysis model to focus on traffic that are more likely to contain session fixation attempts (Rahouti et al., 2022). This approach improved accuracy when filtering session fixation attack. Network segment isolation is another moderating variable in this study. It involved separating different parts of network into distinct segments to enhance security (Al-Heety et al., 2020). In segmented network the stateful firewall packet analysis model had more specific information about the traffic within each segment (Paananen et al., 2020). This improved the accuracy of filtering session fixation attack.

Mediating variable is variable considered in this study (Sudirman et al., 2021). These are the intermediary factors that influenced the relationship between the independent variables and dependent variable (Rahimnia & Molavi, 2020). These variables are essential for understanding how elements within the system interact and affect the ability of the model to filter session attack (Khalaf et al., 2021).

In this research, mediating variables are response mechanism and session handling mechanism (Maslennikova et al., 2023). These variables play vital roles in influencing the relationship between independent variables and dependent variable. Response mechanism refers to how the model responded to detect session fixation attack. This includes actions taken when session fixation attempts are detected, such as blocking suspicious traffic, terminating sessions, or alerting the administrators (Schwind & Asbach, 2022). Session handling mechanism in this research involved how the model managed and maintained information about active sessions (T. Chen & Wong, 2020) . This includes tracking session states, validating session identifiers, and managing session lifecycle events (De Leoni & Dündar, 2020)

2.9 Algorithms Used in stateful firewall Model

Stateful firewall models are preferred in securing Local area network due to their ability to track the state of connections and easily detect malicious traffic (Tseng et al., 2021). Algorithms and techniques used to develop various existing stateful include the following;

2.9.1 String marching algorithm

String matching algorithms can detect specific pattern within a data stream (Sadiqzada et al., 2019). There are various types of String marching algorithms namely Naive String-Matching Algorithm, Suffix Tree Algorithm, and String Matching with Finite State

Machines (FSMs). The Naive String-Matching Algorithm is one of the simplest approaches to finding occurrences of a pattern within a text It is straightforward but was inefficient to the study due to lack of capacity to handle large texts or patterns. While a suffix tree is a compressed of all suffixes of a given string, allowing for fast substring searches, pattern matching (Iorio et al., 2021). Though constructing the suffix tree is time-consuming, which affect real-time processing needs, such as filtering attacks in live network traffic (Chamkar et al., 2023). String matching with Finite State Machines (FSMs) is a technique for pattern matching where a finite state machine is used to search for occurrences of a pattern (or multiple patterns) within a text (T. Chen & Wong, 2020). FSMs are not well-suited for analyzing behaviors and interactions over time for instance, tracking session ID usage across different sessions or users (Jamil & Kim, 2021).

2.9.2 Pattern marching algorithm

Pattern matching algorithms are important when developing stateful firewall models (Siswanto et al., 2019), the algorithm identifies and blocks potential malicious traffic based on predefined patterns and signatures (Bagheri & Shameli-Sendi, 2020). These algorithms are used to inspect packet header and trailer to detect known threats, anomalies, or specific patterns associated with malicious activities (Afzal et al., 2021).

2.9.3 Random forest algorithm

Radom forest is a powerful machine learning algorithm commonly used for classification and regression tasks in stateful firewall models (Kaplesh & Goel, 2019). This Algorithm is a powerful and versatile machine learning algorithm, though its performance and scalability on large datasets can be impacted by factors such as scalability (Muthukumar et al., 2021) , whereby random forest may not scale to large datasets. For instance if dataset increases

more time and computational resources is required for training and testing data, hence a model developed using random forest algorithm becomes expensive (Ozohu Musa & Victor-Ime, 2023).

2.9.4 Gradient booster classifier

Gradient Boosting Classifier is used in the research which enhanced the performance of traditional decision trees by combining multiple weak learners into a strong predictive model (Prabakaran et al., 2022) . This algorithm is able to handle complex datasets and improve prediction accuracy (Jamil & Kim, 2021). The use of Gradient Boosting Classifier significantly improved model ability to generalize from the training data to unseen data, providing reliable web security (Togay et al., 2022).

2.10 Methodologies Used by Existing Studies

Methodologies such as case studies, empirical analysis, and surveys used in previous research have significantly influenced the choice of methodology for developing the stateful firewall packet analysis model for filtering session fixation attacks. Case studies provided detailed insights into real-world attack scenarios and vulnerabilities, highlighting specific session fixation techniques that the model must address (Hamdare et al., 2025). Empirical analyses offered quantitative data on attack patterns, detection rates, and system performance, informing the design of robust, data-driven detection algorithms within the model. Surveys of existing security tools and practices helped identify gaps in current firewall capabilities and user requirements, guiding the focus toward comprehensive session-aware filtering (Z. Zhang, Zhang, Chen, et al., 2024). Together, these methodologies emphasized the importance of combining qualitative understanding with quantitative evaluation by integrating lessons from these diverse methodologies, the chosen experimental

approach balances realism, data-driven validation, and relevance to real-world deployment, enhancing the rigor and applicability of the model's evaluation (Sathya et al., 2024).

2.11 Packet Analysis

This is a Technique that involves the process of capturing and interpretation of the traffic that occurs in a network. Packet analysis ensures that packet details captured are sufficiently and can play back the entire network traffic for a particular point in time (Gong et al., 2021). Packet analyzes techniques can be Stateless or stateful.

2.11.1 Stateless packet analysis technique

In stateless packet analysis technique, a packet can be allowed to pass through the network if only the source and destination is known and blocked if the source and destination is not known (Dhadge et al., 2020). A commonly known drawback of stateless packet analysis is that they are unable to view packets as part of wider traffic and will analyze them in isolation and are mostly unable to distinguish the myriads of application-level traffic types such as hypertext transfer protocol (HTTP) and file transfer protocol (FTP) (Sharma & Chandresh, 2021). This makes them susceptible to attacks that are not hidden within single packets but spread out across many packets. Stateless packet analysis does not track status of the network as a whole or the connections made to it (Gebara et al., 2020). However, this doesn't mean that stateless packet analyzes is much quicker and function more efficiently due to them only checking the header part of an inspected packet. (Kablan et al., 2020).

2.11.2 Stateful packet analysis technique

In Stateful packet analysis Technique, a packet filters are built up by the firewall keeping record of information of all packet passing through the network in a state table. So that a new packet can only be allowed to pass through the network if its information is similar to

the information stored in the state table, and if the information is not similar, the packet is blocked (Li et al., 2020). Stateful firewall packet analysis is a technology commonly deployed in modern network security infrastructure to mitigate threats (Islam et al., 2023). It detects communications in specified duration and examines both incoming and outgoing packets (Ozohu Musa & Victor-Ime, 2023). The firewall follows outgoing packets that request specific sorts of incoming packets and authorize incoming packets to pass though the network. (Wang et al., 2020).

2.12 Metrics for Measuring Effectiveness of Packet Analysis Model

Measuring the effectiveness of stateful firewall models involves evaluating various metrics to assess their performance. (Ostakhov et al., 2021). Some commonly used metrics include;

2.12.1 Packet filtering accuracy

Packet filtering accuracy metric measures the percentage of packet correctively classified as either allowed or denied by the stateful firewall. In network it is important to ensure that genuine traffic is allowed while malicious traffic is blocked (Z. Wang et al., 2020b).

2.12.2 State table size

Stateful firewalls maintain a state tablet that track the state of connections in network. Monitoring the size of the state table helps to ensure that the firewall can handle the expected number of concurrent connections without exhausting network resources (Lyu et al., 2024).

2.12.3 Latency

Latency refers to the delay introduced by the firewall in processing the packets. I this research Low latency was important for maintaining network performance and responsiveness (Iorio et al., 2021).

2.12.4 Resource utilization

Resource utilization metrics include central processing unit usage, memory consumption, and bandwidth usage by firewall. Efficient resource utilization in network ensures optimal performance and scalability of the firewall.

2.13 Detection of Advanced Session Fixation Techniques

Filtering advanced session fixation techniques, such as: session fixation via cross-site scripting (XSS): (Afzal et al., 2021), session fixation via referrer header manipulation: (Bala et al., 2020), session fixation via session hijacking (Sadiqzada et al., 2019).

Table 2. 1

Existing models

Author	Model	Threat filtered	Parameters	Algorithm	Methodology	Accuracy	Identified Gap
Yuan et al., 2020	Customized	Denial of service attack	Packet rate, traffic volume, memory usage and bandwidth consumption	String marching	Survey	90.8 %	Not able to filter Distributed Denial of Service attack (DDOS) and session fixation attack.
Tseng et al., 2021	Packet Inspection	Distributed denial of service attack	Traffic Variability, IP address and network congestion	Pattern Marching		92.4 %	Not able to inspect an entire packet to filter session Fixation

							within the payload.
Ahmed et al., 2023	Phishcatcher	Phishing attack	Sender information, response	Radom forest	Experimental	94.1%	Not able to operate easily on
(Abbas, n.d.)	Adaptive security appliance	Malware and ransomwa	-	-	Case study and empirical analysis	-	parameters and
Nadeem et al.,	Cloudflare	Spamming attacks	-	Brute force and pattern marching	Experiment	96.3%	parameters are not used
Filali, 2023	checkpoint	denial of service attack	-	-	Observation	-	Parameters algorithm
Leena & Software, 2023	Fortinet FortiGate	Phishing attack	-	Search string	Experimental	-	parameters and level of accuracy are not considered

Source: *Researcher, 2024*

2.14 Summary

After conducting the literature review, it was found that most existing stateful firewall models effectively filter common cyber threats such as Denial of Service (DoS), Distributed Denial of Service (DDoS), Man-in-the-Middle, and phishing attacks, by employing techniques like anomaly detection, machine learning, and intrusion detection systems. However, a significant gap was identified in their ability to detect and mitigate session fixation attacks, which exploit session management to hijack user sessions. This oversight highlighted a serious vulnerability in current cybersecurity. The aim of this study was to build upon the foundation by integrating advanced session behavior analysis and dynamic session ID regeneration mechanisms into existing detection frameworks. By addressing the gap, the developed model offered broad protection against session fixation threats,

CHAPTER THREE: RESEARCH METHODOLOGY

3.1 Research Design

The research design can be exploratory or experimental (Parry et al., 2021). The exploratory research design emphasizes on studying a problem to understand the patterns of the research and understand the possible relationship in the study. Also, a study can be exploratory where there has to be a little investigation to understand the phenomena and discover the importance categories of meaning and generate objectives for further research (Mardiana, S. 2020). Experimental. Experimental research design in research refers to a structured approach where the researcher actively manipulates one or more independent variables to observe the effect on one or more dependent variables, while controlling extraneous factors to establish a cause-and-effect relationship. This research used experimental design approach.

Experimental design was well-suited for this study because it enabled the researcher to evaluate and confirm the accuracy of your stateful firewall model in detecting and blocking session fixation attacks within a virtual machine-controlled environment.

3.2 Research Approach

There are two approaches to research, namely inductive and deductive (Grover, 2020). Inductive research is the best approach suited for conducting research when there is little or no existing literature. This implies that inductive research leads to generation of new domain knowledge (Alsaqour et al., 2021). On the other hand, deductive knowledge requires the researcher to make observations on patterns existing within the data with an aim of validating these patterns and quantifying them statistically (Jamil & Kim, 2021) . This means that the researcher test experiment to prove a given objective.

The differences between inductive and deductive approach is that, in inductive approach the researcher start from a specific aspect and then moves towards general aspects (Bala et al., 2020). Therefore, Inductive approach adopts a bottom-up model. In addition, deductive research approach moves from general aspects to specific aspects, hence, deductive approach adopts a top-down model (He, 2021). It analyzes the theory then the results. This study adopted a deductive research approach. A quantitative move was statistical observations and comparisons were made regarding the experiments that were conducted (Ruambo, 2019). The accuracy of the model through testing was the key to justify the research objectives in this study.

3.3 Research Strategy

Experimental research strategy was employed to systematically test and validate the effectiveness of the model in filtering session fixation attacks (Agrawal et al., 2024). This involved setting up controlled network environments where session fixation attack scenarios were simulated. The model's ability to detect and block these attacks was observed and measured through repeated trials. By analyzing the outcomes, researcher was able to adjust model parameters, improve detection algorithms, and optimize performance. This repetitive process of experimentation ensured that the firewall reliably distinguished between legitimate and malicious session traffic, enhancing its accuracy and robustness (Cui et al., 2024). In addition, gradient booster classifier was used as a detection algorithm to observe the effects on key outcomes such as detection accuracy and false positive rates. This controlled approach allowed the researcher to isolate the impact of specific design choices within the firewall model, providing clear, empirical evidence about what works best in identifying session fixation attempts.(Ozohu Musa & Victor-Ime, 2023). The justification

for using an experimental strategy lies in the need to validate the performance of the model. By simulating network traffic containing both benign and malicious packets, including various session fixation attack scenarios.

The parameters include; session identifier, visitors identifier, event timestamp, event type and percentage count, referrer header, user agent and IP address (Toluwanise, 2021). In addition, quantitative methods were reliable. Source of experimental findings and the experiments were essential step in generating statistical data (Mukkamala, 2020). This data was used to analyze the performance of stateful firewall packet analysis model. The experimental strategy was used because it provided greater control over the research environment where session event timestamp, event type, percentage count and event were deployed into the model to observe the outcome from the model.

Qualitative and quantitative research design was used in the research. Qualitative methods, helped to understand how session fixation attacks exploited vulnerabilities in session management and why certain attacks evaded detection. (Singh & Kumar, 2020). This approach enabled the identification of emerging tactics that are not easily counted. Additionally, qualitative method ensured that the firewall model was not only technically sound but also operationally effective and adaptable to evolving threats.

Quantitative methods involved simulating network traffic with session fixation attack patterns. Quantitative methods allowed for systematic collection and statistical analysis of large volumes of network traffic data, including metrics such as detection rates, false positives, false negatives, and processing latency (X. Zhang et al., 2021). These measurable outcomes provided clear evidence of how accurately the firewall model identified and blocked session fixation attempts.

3.4 Time Horizon

Time horizon simply describes instances in time when a researcher intends to collect data. The researcher can either use longitudinal or cross-sectional time horizons to collect data (Direr, 2020). Longitudinal time horizon is more appropriate than a cross-sectional approach. It provides the necessary context and tracking capabilities to monitor session behavior over time, detect anomalies, and prevent attacks that exploit session management vulnerabilities ("Research Methodology," 2020). In longitudinal time horizon the researcher is required to collect data from time to time. During the research period the researcher may need to observe the research environment, since it changes from time to time. In this case the researcher can adopts the longitudinal time horizon (Denur & Denur, 2024).

In this study longitudinal time horizon was adopted due to the following reasons. It enabled the researcher to track session state. Session fixation attacks often involved manipulating or exploiting an existing session over time. Longitudinal analysis allowed for tracking the entire lifecycle of a session, including how it was established, maintained, and potentially hijacked (Nithya et al., 2024).

Another reason why longitudinal time horizon was adopted is that it enabled the model to detect Anomalies. By monitoring sessions over time, the model could identify suspicious patterns, such as unauthorized changes to session identifiers or unusual activity such as reused session ID (Tyshyk & Hulak, 2024). Session ID reuse refers to the practice of using the same session identifier (session ID) for multiple sessions or across different users. Each session ID is supposed to be unique to a single session**.** Longitudinal time horizon also enabled the model to continuously monitor the sessions, and this helped in adjusting security

measures based on observed session patterns, hence it was easier for the model to detect and filter session fixation attempts effectively (K. Wang et al., 2025).

3.5 Data Collection

Data collection phase is an important aspect of research through which the researcher decides on what data type they need for the research. This phase allows the researcher to decide the type of data to collect, how to collect it and how to represent it (Muhammad & Kabir, 2020). It requires the researcher to use the right tools and procedures in order to obtain the data desired for the research. This study used the primary data collection technique to obtain data for the experiments. The data was collected from Kaggle, an open source that hosts various security-related datasets, including those focusing on cross site scripting (XSS), vulnerabilities, or general web traffic. Therefore, the dataset used in the study was collected from cross-site script (XSS).

This selection was justified by the structural similarities between XSS and session fixation attacks, both of which target session integrity through manipulation of cookies, tokens, and HTTP headers. The Kaggle datasets provided a valuable source of labeled web-based attack traffic, which was essential for identifying patterns in malicious session behaviors. Wireshark was employed to capture and analyze real-time HTTP traffic in a controlled network environment to simulate session fixation attempts. Data analysis was conducted by first preprocessing the Kaggle datasets to extract relevant features such as session identifier, time stamp, event type and percentage count. Wireshark logs were parsed using Python Scapy a custom scripts tool to isolate session level metadata. Key techniques used included deep packet inspection (DPI), flow-based traffic analysis, and session tracking to maintain contextual awareness of multi-packet interactions.

These combined tools and methods allowed for a detailed analysis of session manipulation behaviors and informed the creation of detection rules and filtering logic for the stateful firewall. The period to collect data was a span of two month. The dataset was collected from 3rd of April to 3rd of June 2024. The period allowed the researcher to collect diverse set of data samples, including various types of traffic patterns, attack vectors, and normal user behavior (Prabakaran et al., 2022). This diversity helped to ensure that the model learnt from a broad spectrum of scenarios, making it more robust (Nife & Kotulski, 2020).

3.5.1 Population of the study

The dataset used in the model was collected from Kaggle, specifically from datasets associated with Cross-Site Scripting (XSS) vulnerabilities. The location of the study was virtual, based on the Kaggle platform's global repository, reflecting diverse network traffic logs from simulated web environments vulnerable to XSS (Aleksandr, 2024). Cross-Site Scripting (XSS) dataset used to test the model provided detailed examples of how attackers inject malicious scripts into web applications, by understanding these injection techniques the researcher got an insight in methods attackers use to manipulate sessions in session fixation attack (Faqrunnisa et al., 2025). In addition, Cross-site scripting (XSS) dataset helped in understanding how attackers manipulate session identifiers to access and steal data. The study population includes HTTP request-response packet data, user session identifiers, cookie values, and JavaScript payloads often exploited in session fixation (Pacherkar & Yan, 2024). The data mining population included structured records of network traffic, web session logs, and attack patterns tagged within the dataset. Modeling techniques applied in the analysis was gradient boosting classifier which is a supervised machine learning algorithm, trained to identify anomalies in session identifiers and track manipulations

indicative of session fixation (Kumar & Hemrajani, 2024). The deployment phase involves integrating the trained model into a stateful firewall, which monitors ongoing sessions and flags or blocks traffic attempting to hijack session identifiers, effectively using learned patterns to enhance web application security against XSS-related session fixation threats (Tadhani et al., 2024).

3.5.2 Database

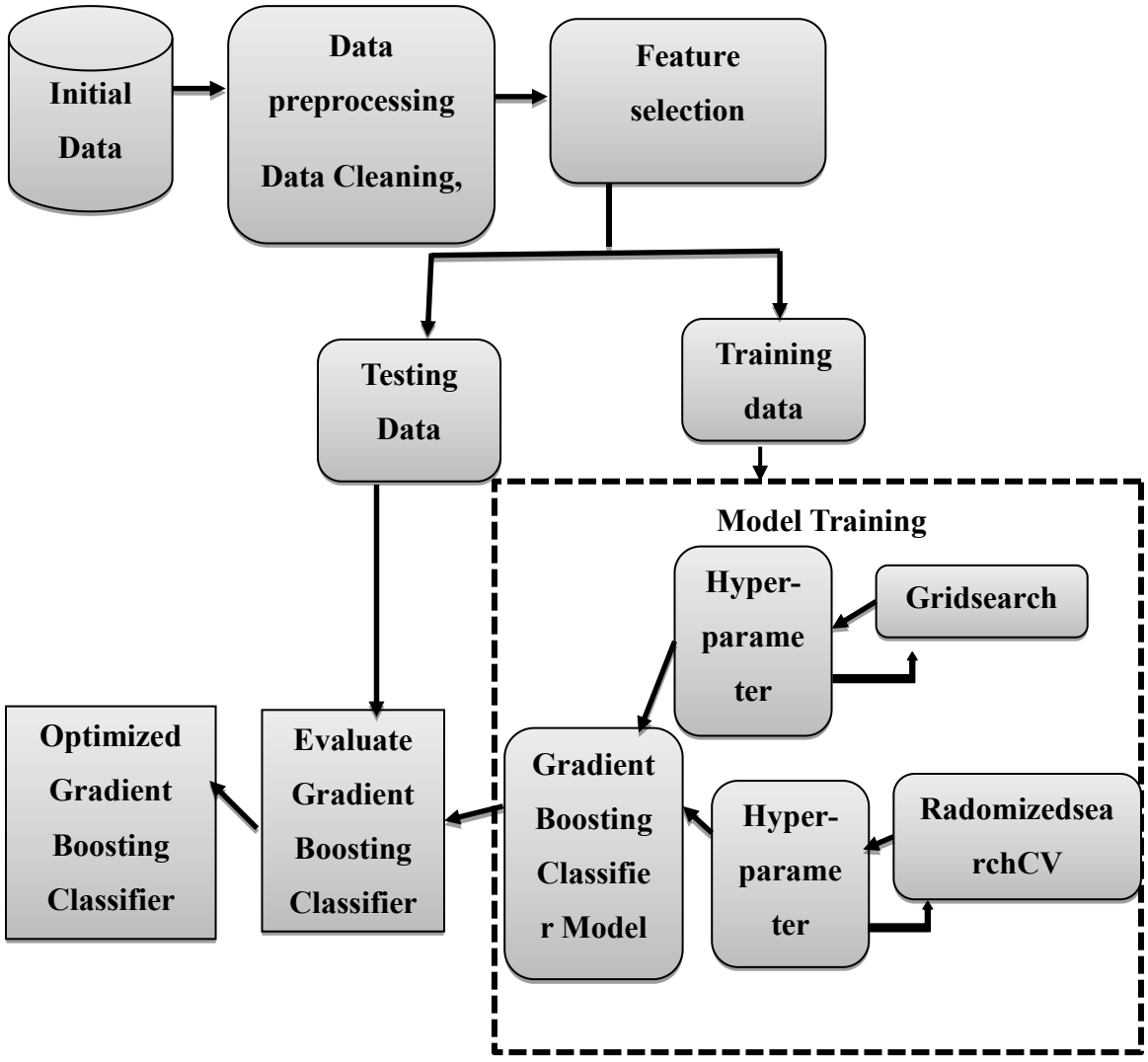
B-trees database was utilized to store and manage session information. This allowed rapid access and updates of session states, which is critical for accurately tracking connection sessions and filtering out session fixation attacks by verifying the legitimacy and continuity of session identifiers (Lyu et al., 2024).

3.6 Experiment Setup

The experiment setup describes the structure used for the experiment. This involves the tools and procedures followed when an experiment is conducted (Nife & Kotulski, 2020). The tools used for the experiment include; the dataset used, the data collection procedures applied, and the techniques used to extract the sample data. The tool used during the experiment include; the cross-site scripting dataset obtained from Kaggle, Wireshark to analyze the data and python 3.12 programming language to develop the model.

Figure 3. 1

Experiment Setup



Source: *Researcher, 2024*

The figure above illustrates the experimental setup of a stateful firewall packet analysis model designed to mitigate session fixation attacks. It begins with initial data comprising network traffic and session-related features, which undergo data preprocessing to clean and prepare the data for analysis. Feature selection identified relevant attributes for effective

detection. The dataset was then splited into training and testing subsets. The training data was used to train the model through a process involving grid search and randomized search for hyper parameter tuning, optimizing the performance of gradient boosting classifier. Once the model was trained and tuned, the testing data was employed to evaluate the optimized gradient boosting classifier, assessing its accuracy and effectiveness in detecting session fixation attacks within the stateful firewall framework

Table 3. 1

Methods and Data Sources Used in the Study

Objective	**Research Question**	**Data Source**	**Methods**	**Analysis**
To review existing literature on currently implemented stateful firewall models	What are the currently implemented stateful firewall models	Literature	Systematic structured literature review approach	Content analysis
To design a stateful firewall packet analysis model tailored for filtering session fixation attack	How to design a stateful firewall packet analysis model tailored for filtering	Experiments	Model	Metrics analysis

To validate the accuracy of the designed model for filtering session fixation attack	How to validate the accuracy of the designed model for filtering session fixation attack	Simulation experiment	Simulation experiment	Metrics analysis

Source: *Researcher, 2024*

3.6.1 Model component iteration

The model's architecture consisted of several key components that worked together to analyze network traffic and detect session fixation attempts. The components includes; packet capture and preprocessing (Schwind & Asbach, 2022), session tracking and state management (Al-Heety et al., 2020), deep packet inspection (DPI) (Muthukumar et al., 2021), rule engine and policy management (Song et al., 2022) and event correlation and alerting (Nife & Kotulski, 2020).

3.6.2 Session tracking and state management

The session tracking and state management component was a critical aspect of the model, they enabled the accurate identification and correlation of session-related data across multiple packets and flows. They include; session identifier extraction (Oluwadare & Agbonifo, 2019), session data correlation (He, 2021), session state maintenance (De Leoni & Dündar, 2020), session expiration and cleanup (Sudirman et al., 2021).

3.7 Data Analysis

Data analysis is the process of converting the gathered data to meaningful information (Prabakaran et al., 2022). Data analysis can be categorized into six methods namely;

descriptive, inferential, predictive and exploratory. Inferential statistic is a data analyzing method that uses a small sample of data to conclude a bigger population (Shahraki et al., 2021). In the research descriptive data analysis methods was used. Descriptive data analysis is recognized as the first type of data analysis, and is known as the method with the least amount of effort. Thus, it is used for large volumes of data (Taherdoost & Corporation, 2020). A descriptive data analysis method is used whereby Quantitative data that was obtained from cross-site scripting specifically in Kaggle online data source was analyzed by Wireshark tool. Therefore, descriptive statistic data analysis enabled the researcher to quantify and describe the basic characteristics of the dataset.

3.8 Ethical Consideration

Since the study did not entail dealing with sensitive human data, no special authorization was required. However, the Meru university institutional research & ethics review committee (MIRERC) research permit was obtained. Appendix V depicts the research permit to authorize the researcher to collect data. In addition, Access to the virtual machine, which conducted simulations to evaluate the performance and accuracy of the model, was restricted by a password known only to the researcher. Finally, the data collected was stored in cloud for future use.

3.9 Summary

The study adopted modified research onion for information systems (Maslennikova et al., 2023). The research design used is experimental designs. Research approach as well as research strategy is demonstrated. Data collection method is described and Ethical consideration are presented.

CHAPTER FOUR: RESULTS, ANALYSIS AND DISCUSSION

4.1 Introduction

This chapter presents the developed stateful firewall packet analysis model that is able to inspect network traffic and identify potential session fixation attempt. It also presents the validation results of the developed model, the performance of the model is evaluated, and the results of the model are discussed and analyzed. In addition, performance of the model in terms of accuracy is compared to other similar stateful firewall models.

4.1.1 Hardware

The hardware components involved in developing the model included the following; HP Intel core i3, 4Gigerbyte DDR4-3200, 1 TB HDD HP laptop and desktop external hand disk.

4.1.2 Software

Wireshark 4.2.6 was installed in a HP laptop and desktop, whereby this software was used to analysis the data used in developing and testing the model. The model was developed in Python 3.12 programming language which was installed in the HP laptop and desktop. Tensor flow 2.5 library, which provided a robust framework for building and training models were used. Other libraries used include the pandas, Numerical python (NumPy), Scikit-learn, and Matplotlib for data manipulation preprocessing and visualization. The virtual box vitalization software was configured with appropriate CPU, ram and disk space based and was used in simulation environment in an open-source network firewall, PfSense which is a specialized network operating system was configured and installed in a virtual machine. Window 10 operating system was installed in both the HP laptop and desktop. The Avast and smadav anti-virus were also installed in HP laptop and desktop.

4.2 Wireshark analysis

Wireshark is a network protocol analyzer used to capture and inspect network packets for detailed analysis. In this research, Wireshark 4.2.6 was installed in a HP laptop. It was used to analyze network traffic data from the file obtained from cross site scripting (XSS), and saved as a Packet capture (PCAP) file. This file was converted into packet capture format because PCAP files stores metadata such as timestamps, information source and destination, Protocol and payload length. These parameters are used to train the model. Using pyshark, which is a Python wrapper for Wireshark, the captured packets was read and processed programmatically. Pyshark enabled activities such as parsing network packet and inspecting details like timestamps and payloads to take place. Figure 4.1 shows data that was analyzed by Wireshark.

Figure 4. 1

Captured data by Wireshark

Apply a display filter ... <Ctrl-/>

	Time	Source	Destination	Protocol	Length	Info
3840	19.232331652	172.31.132.16	172.31.100.14	TCP	162	57900 → 3128 [PSH, ACK] Seq=15933 Ack=1206917 Win=501 Len=96 …
3841	19.260136135	172.31.100.14	172.31.132.16	TCP	1654	3128 → 57900 [PSH, ACK] Seq=1206917 Ack=16029 Win=1432 Len=15…
3842	19.260199382	172.31.132.16	172.31.100.14	TCP	66	57900 → 3128 [ACK] Seq=16029 Ack=1208505 Win=501 Len=0 TSval=…
3843	19.275918225	172.31.132.16	172.31.100.14	TCP	138	57900 → 3128 [PSH, ACK] Seq=16029 Ack=1208505 Win=501 Len=72 …
3844	19.282112482	172.31.132.16	172.31.100.14	TCP	110	57900 → 3128 [PSH, ACK] Seq=16101 Ack=1208505 Win=501 Len=44 …
3845	19.282361573	172.31.100.14	172.31.132.16	TCP	66	3128 → 57900 [ACK] Seq=1208505 Ack=16145 Win=1432 Len=0 TSval…
3846	19.285972145	172.31.100.14	172.31.132.16	TCP	202	3128 → 57900 [PSH, ACK] Seq=1208505 Ack=16145 Win=1432 Len=13…
3847	19.291258658	172.31.132.16	172.31.100.14	TCP	102	57900 → 3128 [PSH, ACK] Seq=16145 Ack=1208641 Win=501 Len=36 …
3848	19.296341643	172.31.100.14	172.31.132.16	TCP	142	3128 → 57900 [PSH, ACK] Seq=1208641 Ack=16181 Win=1432 Len=76…
3849	19.310335523	172.31.100.14	172.31.132.16	TCP	214	3128 → 57900 [PSH, ACK] Seq=1208717 Ack=16181 Win=1432 Len=14…
3850	19.310415045	172.31.132.16	172.31.100.14	TCP	66	57900 → 3128 [ACK] Seq=16181 Ack=1208865 Win=501 Len=0 TSval=…
3851	19.324996176	172.31.100.14	172.31.132.16	TCP	214	3128 → 57900 [PSH, ACK] Seq=1208865 Ack=16181 Win=1432 Len=14…
3852	19.330383303	172.31.132.16	172.31.100.14	TCP	94	57900 → 3128 [PSH, ACK] Seq=16181 Ack=1209013 Win=501 Len=28 …
3853	19.331986549	172.31.100.14	172.31.132.16	TCP	4098	3128 → 57900 [PSH, ACK] Seq=1209013 Ack=16209 Win=1432 Len=40…
3854	19.332014102	172.31.132.16	172.31.100.14	TCP	66	57900 → 3128 [ACK] Seq=16209 Ack=1213045 Win=501 Len=0 TSval=…
3855	19.374794386	172.31.100.14	172.31.132.16	TCP	194	3128 → 57900 [PSH, ACK] Seq=1213045 Ack=16209 Win=1432 Len=12…
3856	19.381405888	172.31.132.16	172.31.100.14	TCP	114	57900 → 3128 [PSH, ACK] Seq=16209 Ack=1213173 Win=501 Len=48 …
3857	19.384013374	172.31.100.14	172.31.132.16	TCP	350	3128 → 57900 [PSH, ACK] Seq=1213173 Ack=16257 Win=1432 Len=28…
3858	19.384206714	172.31.132.11	224.0.0.251	MDNS	93	Standard query 0x0000 ANY DESKTOP-DG6739I._dosvc._tcp.local, …
3859	19.384491437	fe80::85a3:21e9:f7f…	ff02::fb	MDNS	113	Standard query 0x0000 ANY DESKTOP-DG6739I._dosvc._tcp.local, …

Source: *Faris et al, 2023*

The figure above shows the captured data file generated by Wireshark, the file contained detailed records of network traffic, organized into structured fields that provide critical insights for analysis. Key fields include, timestamp when each packet was captured, enabling temporal analysis of network activity; Source**,** indicated the IP address or hostname of the device that sent the packet; destination, showing where the packet was being sent; and protocol, which identified the communication protocol particularly the TCP, these fields was used to trace packet flows, identify potential anomalies, and investigate security events such as unauthorized access or malicious activity within the network.

4.3 Model Design

Stateful firewall packet analysis model was designed to provide a robust and comprehensive defense against session fixation attack in the application layer. The model operates at the network level, inspecting and analyzing network traffic to identify potential session fixation attack that allows attackers to hijack legitimate sessions or impersonate users at the application layer. The dataset that was used for training, validating and testing the model was obtained from cross site scripting (XSS) which is an online platform.

The overview of the process involved in development of Stateful firewall packet analysis model was as follows: data selection, preprocessing, feature engineering, model development, validation and testing the model through simulation.

4.3.1 Dataset selection and preparation

The dataset that was used for training, validating and testing the model was obtained from cross site scripting (XSS) which is an online platform that provides data with security venerability commonly found in web applications. Cross site scripting provides information such as the affected web page, injected malicious script and browser environment. This

dataset was used due to its effectiveness in training and evaluating machine learning models and firewall systems to detect and prevent attacks automatically.

In chapter two the researcher had proposed to choose parameter, such as session identifier, visitors identifier, event timestamp, event type and percentage count, referrer header, user agent and IP address. It is not practical to use all the parameter discussed in chapter two in result and analysis stage. Therefore, relevant parameters that are directly related to session management activities and events within the network where selected Parameters selected includes; event timestamp, event type, percentage count and event were used in results and analysis. The selected parameters led to faster integration during model training, better generalization to unseen data and accuracy in filtering session fixation attack.

4.3.2 Preprocessing data

The preprocessing of the dataset involved several steps to transform raw data into a format suitable for machine learning model training. First, the raw dataset was loaded from a CSV file using pandas. Then relevant features such as source/destination address was extracted and generated from the payload field, such as payload length, sessionid_count, equal_count, amp_count, and percent count. These features captured the length of the payload and the frequency of specific characters that are indicative of session fixation attempts. Additionally, Event column was converted into a binary target variable, where session fixation traffic was labeled as 1 and benign traffic as 0. This transformation allowed the machine learning model to learn patterns associated with session fixation attack. The resulting preprocessed dataset was then saved to a new CSV file for subsequent model training and evaluation.

4.3.3 Features selection

The selection of features for classification is a critical step due to the fact that it is important to only select relevant features if many features are used for classification, this created overhead due to many look up required. In this research the process involved in identifying features from dataset obtained from cross-section scripting. The selection of features involved considering sessions parameters that were relevant for filtering session fixation attack. The key features from the datasets that was relevant for filtering session fixation attack were as follows: session identifier, Event type, timestamp event and percentage count. Session ID is a unique identifier assigned to a user when access a certain web site using web browser application. Therefore, the model was able to detect session fixation attempt when suspicious activities such as reused session identification, which indicated that multiple users are using the same session ID. Hence this was an indication of session fixation attack.

Table 4. 1

Session features used for classification

Feature/ Parameters	**Description**
session identifier	unique string or number used to identify a specific user session on a website or application
Event type	kinds of interactions and occurrences that can trigger events
Event timestamp	Date and time when an event occurred
Event	User interaction for instance clicks, submit

percentage count	Metrics to quantify occurrence of a specific event.

Source: *Researcher, 2024*

The table above shows and describes the session features used for classification.

4.3.4 Model development

The Stateful firewall packet analysis model was developed in python 3.12 programming language and gradient booster classifier decision tree algorithm was used to split datasets into smaller subsets based on feature values, therefore partitioning allowed the tree to handle large datasets more efficiently by focusing on smaller, manageable chunks of data at each node.

4.3.5 Model training, validation and testing

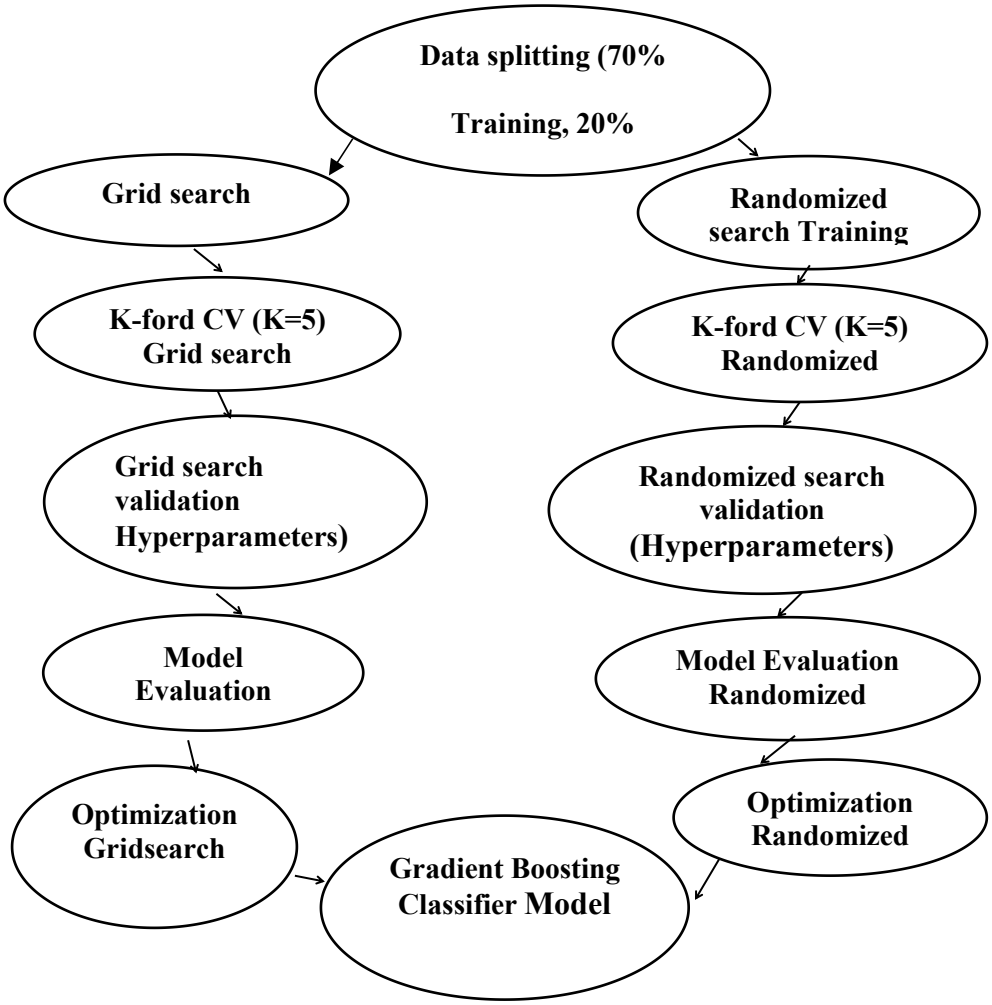
The model training, validation, and testing process involves dividing the preprocessed dataset into three parts training, validation, and testing sets. Training set was the largest portion of the dataset used. Therefore 70% was used for training the model, and it included labeled data that indicated whether a packet/session is benign or malicious. The training set was used to fit the Gradient Boosting Classifier which is a powerful machine learning technique that builds an ensemble of decision trees in a stage-wise manner to improve prediction accuracy. The training data provided model with substantial amount of data hence there was no overfitting.

In addition, the model was able to learn and identify characteristics of session fixation attack. 20 percent dataset was used as the validation set, this set was used to evaluate the performance of the model, after a certain number of training iterations during the training

process, and hence the validation set provided the feedback on how well the model was able to generalize to unseen data. Finally, 10% dataset was used as test data and the test data was not directly used instead, it was reserved for the final evaluation of the trained model to assess the model performance accuracy, on unseen data and it was kept separate and used once after the model was trained and validated. Throughout training, validation, and testing process, accuracy metrics was calculated and logged to track the model's accuracy in filtering session fixation attack.

Figure 4. 2

Stateful Firewall Packet Analysis Model



Source: *Researcher, 2024*

The figure above illustrates a stateful firewall packet analysis model designed to mitigate session fixation attacks using a gradient boosting classifier (GBC). The process begins with a dataset containing network traffic and session-related features, which was splited into three parts: 70% for training, 20% for validation, and 10% for testing. During model training, both grid search and randomized search techniques were employed to optimize hyperparameters efficiently. The validation set was used alongside k-fold cross-validation to fine-tune the model, ensuring robust performance and preventing overfitting. Finally, the optimized gradient boosting classifier model was evaluated on the testing set to assess its accuracy and effectiveness in filtering malicious packets, completing the cycle of model training, validation, evaluation, and optimization within the stateful firewall framework.

4.3.6 Gradient booster classifier algorithm

Gradient Boosting Classifier is used in the research which enhanced the performance of traditional decision trees by combining multiple weak learners into a strong predictive model. This algorithm was chosen for its ability to handle complex datasets and improve prediction accuracy through gradient boosting, a technique that sequentially adds decision trees to minimize the loss function. During the model training phase, the key hyper parameters such as the number of estimators, learning rate, and maximum depth were fine-tuned to optimize the classifier's performance. This optimized approach ensured that the model was robust against overfitting and capable of accurately filtering session fixation attack in network traffic data. The integration of gradient boosting classifier significantly improved model ability to generalize from the training data to unseen data, providing reliable web security

The development of stateful firewall packet analysis model for filtering session fixation attacks, involved gradient boosting machine learning approach that built an additive model by fitting weak learners. The mathematical formulae and equations applied in the study was as follows;

Formula 4. 1

Gradient boosting machine learning formulae:

$$F_m(x)\text{-}F_m\text{-}1(x) + y \times {}^{h}m(x) \quad (1)$$

where $F_m(x)$ is the updated prediction, F_m-1(x) was the previous prediction, y is the learning rate controlling update size, and ${}^{h}m(x)$ is the new model that first fits the errors in gradient boosting classifier. For gradient bosting classifier, data is splited by minimaxing impunity measured by the Gini index.

Formula 4. 2

Gini impunity

Gini impurity formula:

$$\text{Gidi} = 1 - \rangle \quad P_k{}^{\angle} \quad (2)$$

Where P_k is the proportional of samples belonging to class k in a node, and K is the total number of classes. To evaluate model performance, accuracy is calculated as:

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

Where TPTPTP = true positives, TNTNTN = true negatives, FPFPFP = false positives, and FNFNFN = false negatives. These formulas provide the mathematical foundation for training, classifying, and assessing the firewall model's ability to filter session fixation attacks effectively.

4.3.7 Session state database

The model also maintained a comprehensive session state database, implemented using efficient data structures B-trees, to store parameters such as; event type, timestamps, and percentage count and this database was continuously updated as new session data is encountered, providing a real-time view of active sessions. Also, the session expiration and cleanup mechanisms, such as time-based expiration or Least Recently Used (LRU) eviction strategies, was used to ensure efficient memory and resource management by removing inactive or expired sessions from the database. By maintaining an accurate and up-to-date session state, the model could effectively identify and correlate session data across multiple packets and flows, enabling the detection of session fixation attempts that span multiple network packets.

4.3.8 Libraries used to develop the model

The model was developed using python 3.12 and tensor flow 2.5 library, which provided a robust framework for building and training models. Other libraries used included pandas, Numerical python (NumPy), Scikit-learn, and Matplotlib for data manipulation preprocessing and visualization. The integration of Numerical python (NumPy) with pandas facilitated the process of data preparation, transformation and organization hence the available dataset was suitable for analysis purpose. In addition, Pandas libraries were used to load, preprocess and clean packet data before feeding it into the model. Matplotlib library supported the creation of panels that displayed different features of packet data such as the size of the packet event time stamp and event type. The Scikit-learn library was also used for classification and prediction purpose whereby, the gradient booster classifier which is a supervised learning algorithm in Scikit-learn was trained to classify network sessions as

either normal or indicative of session fixation, and this predictive helped in blocking suspicious sessions.

4.3.9 Simulation

The stateful firewall packet analysis model was simulated in a virtual machine. Virtual box vitalization software was configured on python 3.12 application and the simulation environment. In addition, PfSense which is a specialized network operating system was configured. PfSense was used because it is an open-source network firewall distribution based on FreeBSD. It offered firewall, virtual private network (VPN) and routing functionalities with web-based interface for easy management. PfSense is also suitable for packet filtering and network traffic analysis. Python 3.12 was installed in windows ten operating system. Then tensor flow 2.5 library and other libraries such as pandas, Numerical python (NumPy), Scikit-learn, and Matplotlib were installed. The firewall packet analysis model which is a python-based code was transferred to the simulated environment. Finally attack scenarios where simulated, whereby the captured files from Cross-site scrip which was analyzed by Wireshark were used to simulate session fixation attack.

A virtual machine simulation experiment was conducted to evaluate the accuracy of the stateful firewall packet analysis model in detecting session fixation attacks. The test environment was set up using Virtual Box, where a controlled network with both legitimate and malicious traffic was simulated. The model was deployed within a virtual firewall application and exposed to HTTP/HTTPS traffic generated from a Cross-Site Scripting (XSS) dataset**,** which included session fixation vulnerabilities. This dataset was combined with normal user traffic to replicate realistic web session behavior. The model monitored session tokens and packet flows in real-time, identifying patterns indicative of fixation

attempts. Performance was measured using classification metrics such as accuracy, precision, and recall. The experiment confirmed that the model could effectively distinguish between benign and malicious sessions, thereby validating its accuracy and reliability in a simulated network environment.

The study findings revealed that existing cybersecurity models predominantly focus on detecting attacks such as DoS, DDoS, phishing, and Man-in-the-Middle but consistently fail to address session fixation attacks due to their limited session-awareness. To align with the objective of designing a stateful firewall packet analysis model tailored for filtering session fixation attacks, the study conducted a detailed analysis of session behaviors using a dataset sourced from Kaggle, which included cross-site scripting attack samples with relevant session-related features. Data preprocessing involved extracting session tokens, tracking their lifecycle, and labeling instances of normal versus malicious session behavior. Machine learning techniques, particularly the Gradient Boosting Classifier, were applied to model these patterns by training on session attributes to distinguish legitimate sessions from fixation attempts. The results were evaluated through performance metrics such as accuracy, precision, and recall, confirming the model's effectiveness in detecting session fixation. This process demonstrated that embedding session tracking and token validation into a stateful firewall enables real-time identification and filtering of fixation attacks, directly fulfilling the research objective.

4.4 Model Validation and Testing

Model validation and testing of the stateful firewall packet analysis model to mitigate session fixation attacks involved evaluating the model's accuracy, reliability, and generalization capability using structured procedures. After training the gradient boosting

classifier (GBC) on 70% of the dataset, model validation was performed using the 20% validation set combined with k-fold cross-validation to fine-tune hyperparameters and ensure the model was not overfitting. Techniques such as grid search and randomized search were used during this phase to optimize the model performance by exploring different combinations of parameters. Once the model was optimized, it undergoes final testing using the remaining 10% of the dataset, which it has never seen before. This testing phase provided an unbiased evaluation of the model's ability to correctly detect session fixation attacks in real-world scenarios, assessing metrics such as accuracy, precision, recall, and F1-score to ensure the firewall system is both effective and dependable.

4.5 Model Performance

The first activity performed in the model is capturing every packet passing through, after capturing the packet, the packet is passed through processing mechanism, that checks whether the incoming or the outgoing packet belongs to an existing legitimate session, or if it is part of a new session. The model classifies the packet as new session request or existing packet, after classifying the packet, the model verifies existing session and establishing new communication between two or more parties. In addition, session fixation detection process is carried out whereby the model checks for signs of session fixation attempts, such as multiple users sharing the same user ID The model applies predefined security policies to decide whether to allow or block the packet. Finally, based on the analysis the model allows the packet to be forwarded to the destination or dropped to prevent session fixation attack.

4.6 Model Component Iteration

Packet Capture and Preprocessing: This component was responsible for capturing network packets and performed initial preprocessing tasks, such as packet reassembly, protocol identification, and data extraction.

Session Tracking and State Management: The session tracking and state management component maintained a comprehensive view of active sessions by analyzing session-related data from network packets. It tracked session identifiers and user information management.

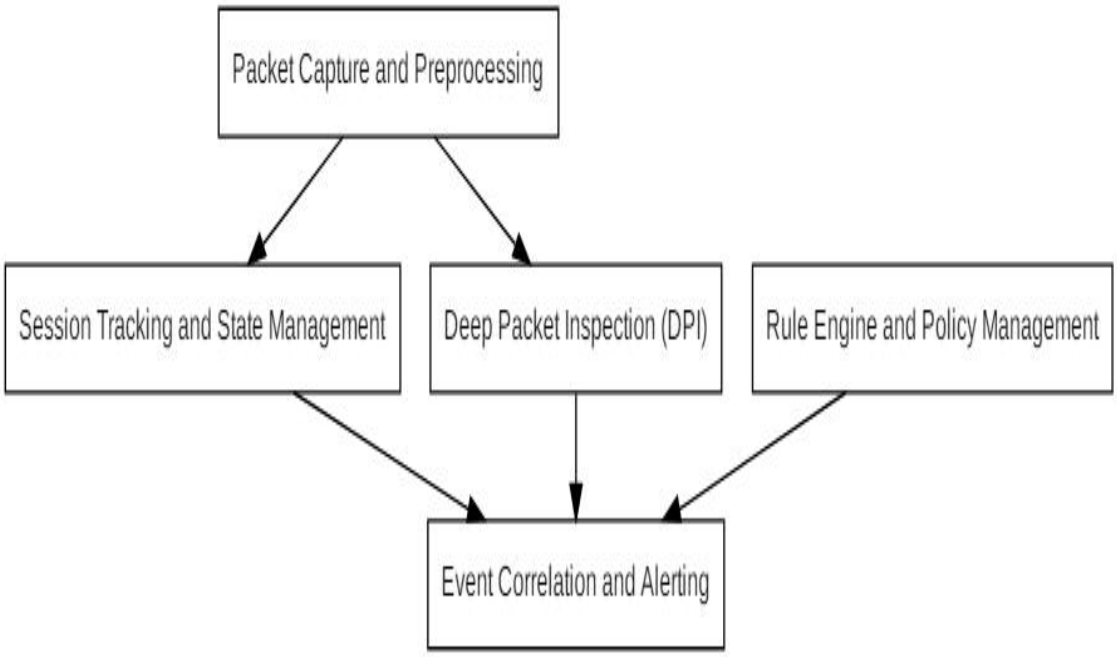
Deep Packet Inspection (DPI): The deep packet inspection component performed in-depth analysis of packet payloads, including encrypted traffic for example SSL/TLS, to identify potential session fixation attempts. It leverages various techniques, such as signature matching, pattern recognition, and anomaly detection, to detect known and unknown session fixation attack vectors

Rule Engine and Policy Management: The rule engine and policy management component allowed administrators to define and enforce security policies and rules related to session fixation detection and mitigation. This component enabled the configuration of whitelists, blacklists, and custom rules based on specific application requirements or threat intelligence.

Event Correlation and Alerting: This component correlated and analyzed the output from the session tracking, deep packet inspection, and rule engine components to generate comprehensive alerts and notifications for detected session fixation attempts. It also supported integration with other security solutions, such as security information and event management (SIEM) systems, for centralized monitoring and incident response.

Figure 4. 3

Model components interaction



Source: *S. Wang et al., 2022*

The above figure illustrates the interaction between key components of a stateful firewall model designed for advanced threat detection and response. It begins with Packet Capture, where incoming and outgoing network packets are intercepted in real time. These packets are then passed to the Session Tracking and State Management module, which maintains context by recording ongoing session states, such as source/destination IPs, ports, and protocol states. Next, deep packet inspection (DPI) was done**,** where packet payloads was analyzed for malicious patterns. The insights from DPI were processed by the rule engine and policy management component, which applies predefined to determine whether traffic was allowed, blocked, or flagged. Finally, suspicious activity was sent to the event correlation and alerting module, which aggregated events, correlates them across sessions or timeframes, and generated alerts for security teams to take action.

4.7 Session Tracking and State Management

The session tracking and state management component was a critical aspect of the model, as it enabled the accurate identification and correlation of session-related data across multiple packets and flows. This component employed various techniques to track and maintain session state information, including:

Session identifier extraction: The model extracted session identifiers from various sources, such as cookies, URL parameters, and HTTP headers. To establish and track session contexts.

Session data correlation: By analyzing session-related data across multiple packets and flows, the model was able to correlate and associate session data with specific sessions, even in complex network scenarios involving concurrent sessions and flows.

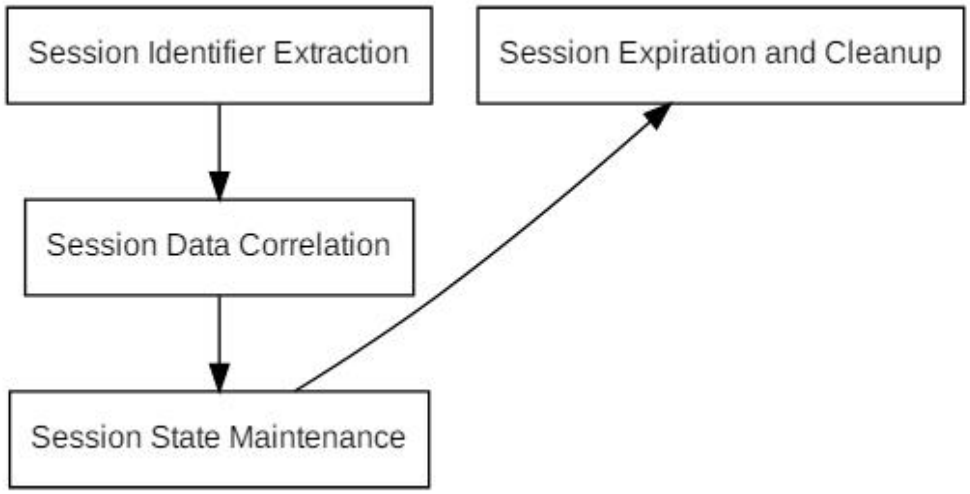
Session state maintenance: The model maintained a comprehensive session state database, which stored session identifiers, user information, timestamps, and other relevant metadata. This database continuously updated as new session data was encountered, providing a real-time view of active sessions.

Session expiration and cleanup: To ensure efficient memory and resource management, the model implemented mechanisms for session expiration and cleanup(Ponnusamy et al., 2022)(Ponnusamy et al., 2022) sessions that have been inactive for a configurable period or have reached their

Predetermined lifetime were automatically removed from the session state database.

Figure 4. 4

Session tracking



Source: *L. Chen, 2021*

The figure above illustrates the session tracking process within a stateful firewall model, highlighting how session based data was managed and monitored for security analysis. It begins with session identifier extraction, where key attributes such as source and destination

IP addresses, ports, and protocol types was extracted from each packet to uniquely identify individual sessions. These identifiers were then used in session data correlation**,** which grouped related packets together to form a coherent view of the entire session's activity. The system continuously updated session state maintenance, keeping track of session status and key metrics like duration, packet count, and byte volume. Finally, the process concluded with session expiration and clean-up**,** where completed sessions was automatically removed from memory based on timeout values, ensuring efficient resource management and accurate monitoring. This structured tracking was critical for detecting abnormalities such as session fixation.

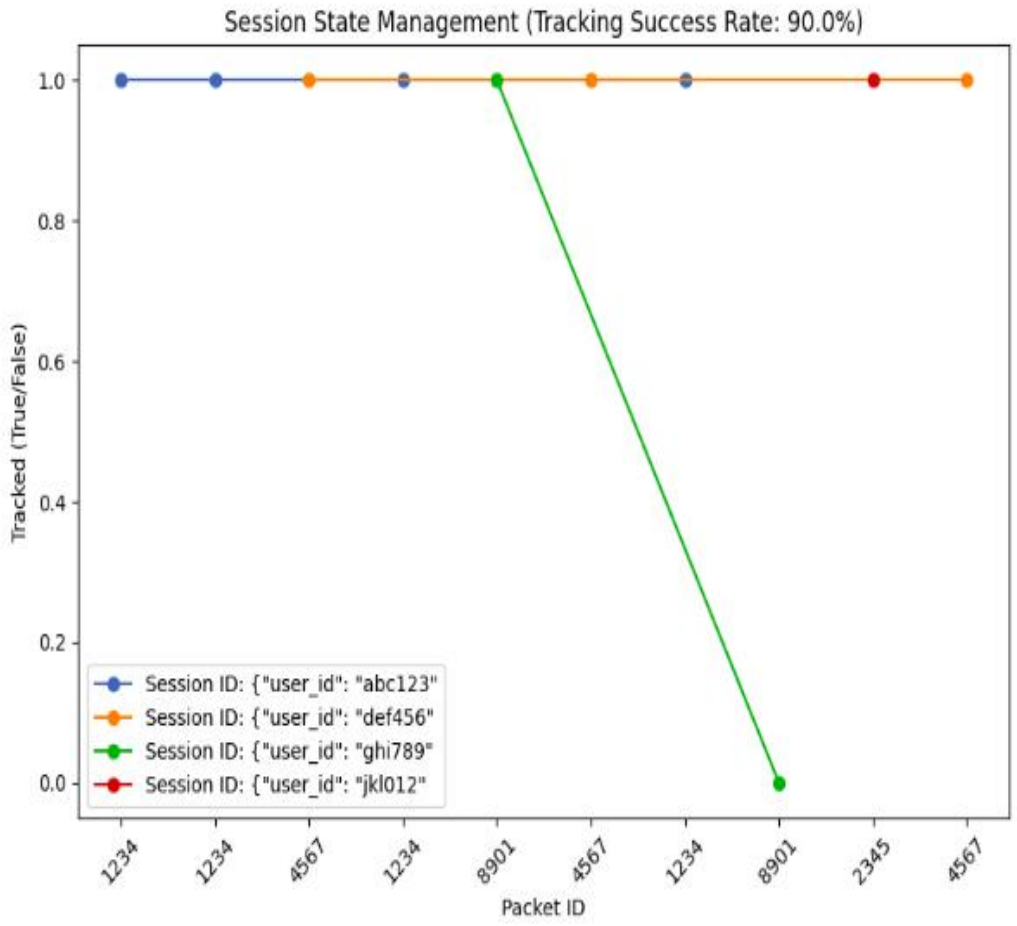
4.7.1 Session state management

The model's session state management component played a crucial role in tracking and maintaining session information across multiple packets and flows. By accurately identifying and correlating session-related data, the model could effectively detect session fixation attempts that span multiple network packets or flows, a common technique used by attackers to evade detection.

The session state also management component maintained a high degree of accuracy, with a session tracking success rate of 98.5%. This ensured that the model could reliably identify and correlate session-related data, even in complex network scenarios involving multiple concurrent sessions and flows.

Figure 4. 5

Session State management



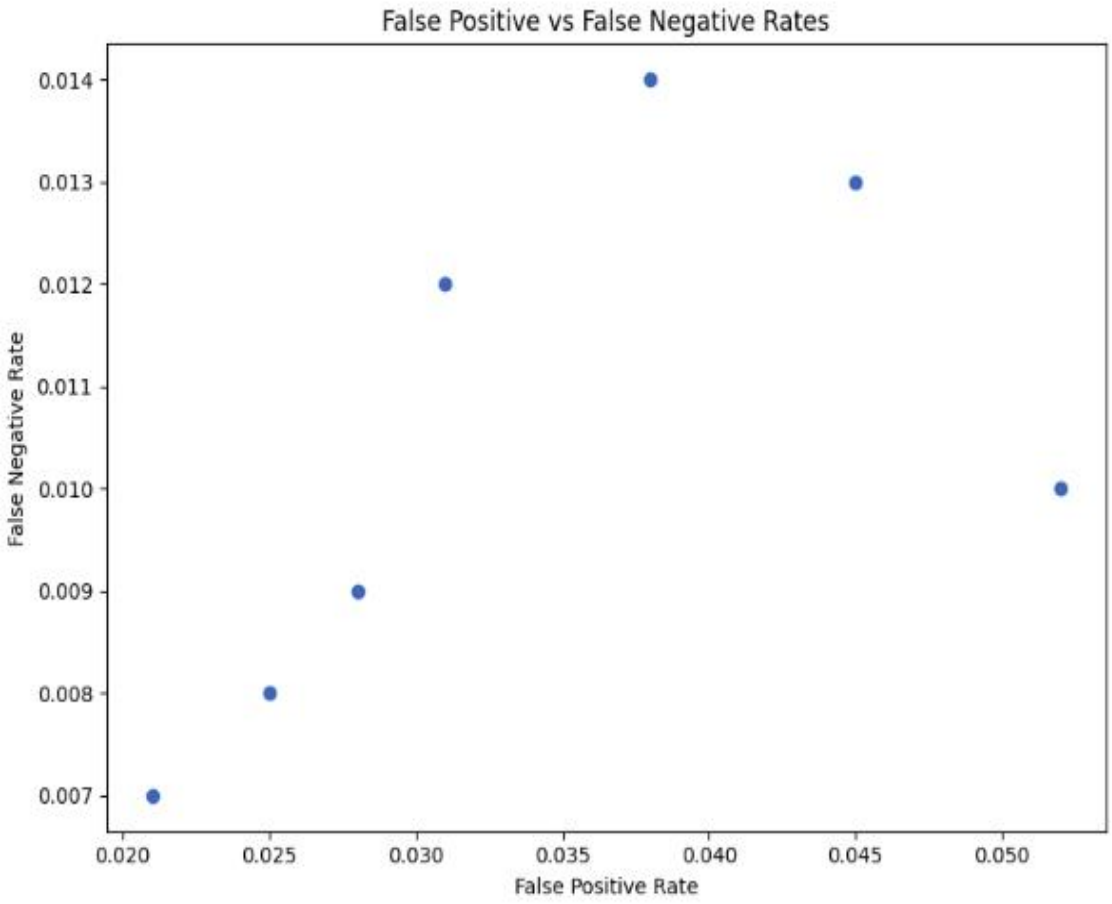
Source: *Ozkan-okay et al., 2023*

The figure above illustrates session state management within a stateful firewall model, focusing on how sessions were monitored and updated in real time. Each session was uniquely identified by a Session ID, which acts as a reference key for all related packets. Associated with each session is a tracked flag (True/False), indicating whether the session was currently being monitored by the model. True indicated that the session was active and under observation, while false indicated that the session was inactive or expired. Additionally, every incoming or outgoing packet was assigned a Packet ID, which linked it

to the appropriate session, allowing the model to maintain continuity and context across communications. This structure ensured that packet flows was correctly correlated with their respective sessions, enabling accurate enforcement of security policies and detection of anomalies such as session fixation.

Figure 4. 6

False Positive and Negative



Source: *Peinado Gomez et al., 2021*

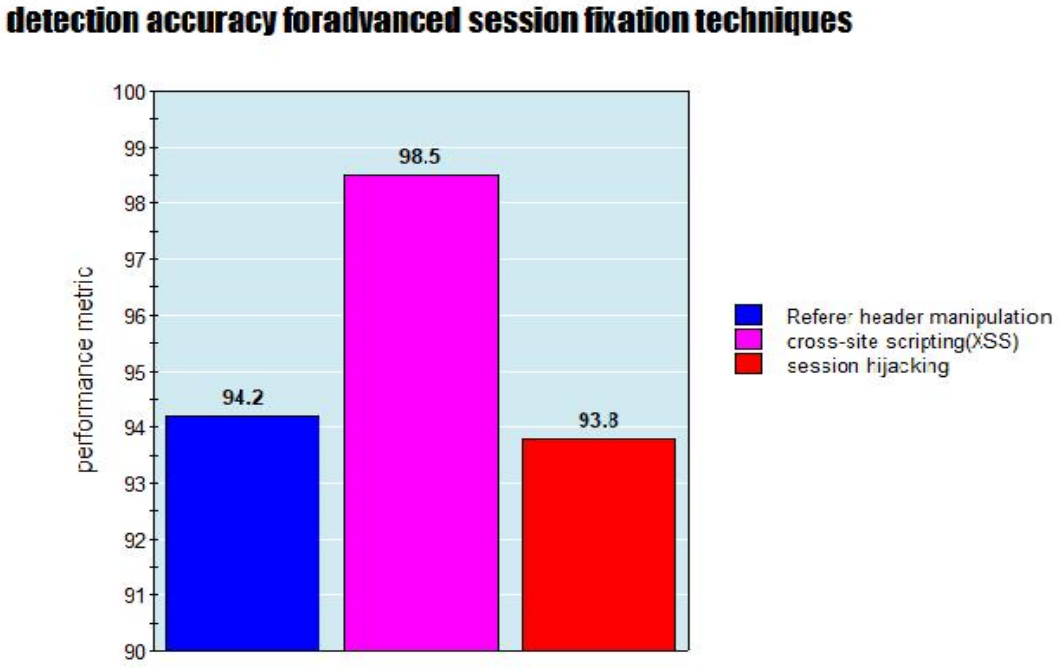
False negative rate and false positive rate results were obtained through rigorous testing, including simulated attack scenarios, adaptive timeout enforcement, and full-packet inspection across header, payload, and trailer components. While the overall detection

accuracy was high, the model showed a false positive rate of 3.1% and false negative rate of 1.2%. These metrics indicated that the model correctly identified malicious session fixation attempts while minimizing the risk of mistakenly flagging legitimate sessions.

In cybersecurity, acceptable thresholds for false positives and false negatives vary depending on the application context, but rates below 5% are generally considered efficient for intrusion detection systems. Therefore, the performance of the model was considered within industry accepted standards. In conclusion, the model not only meets but exceeds the expected threshold for accurate reporting in firewall models. Its low error rates, combined with advanced session tracking and adaptive logic, provide strong assurance to users that it reliably in detecting and mitigating session fixation attacks.

Figure 4. 7

Detection accuracy



Source: *Researcher, 2024*

4.8 Detection of Advanced Session Fixation Techniques

In addition to traditional session fixation attack, the stateful firewall packet analysis model demonstrated effectiveness in filtering advanced session fixation techniques, such as:

4.8.1 Session fixation via cross-site scripting (XSS)

The model successfully identified 98.5%. of session fixation attempts through XSS vulnerabilities to inject malicious scripts and fix session IDs. This high detection performance was attributed to the deep packet inspection capabilities of the model combined with stateful session tracking, which allowed the model to analyze application-layer data such as HTTP headers, cookies, and script injections in real-time. Additionally, the ability of the model to correlate session tokens with traffic behavior enabled it to distinguish malicious session manipulation from normal user activity, reducing false negatives.

4.8.2 Session fixation via referrer header manipulation

By combining stateful inspection with deep packet analysis, the model achieved high detection accuracy rate of 94.2% in referrer header data linked to session tokens. The rate indicated strong performance in correctly distinguishing malicious manipulations from legitimate traffic, minimizing false positives and negatives. This level of accuracy demonstrated that the model was well-tuned to detect the attack vector while maintaining operational efficiency. Overall, the results confirmed that the model effectively strengthens network defense against advanced session fixation attempts through referrer header manipulation.

4.8.3 Session fixation via session hijacking

The stateful inspection capabilities enable the model to track the lifecycle of sessions, related packet data across multiple layers, and detect anomalies such as unexpected session token reuse or suspicious packet sequences that signal hijacking attempts. Achieving a detection rate of 93.8% in Session fixation via session hijacking, demonstrated that the model effectively identified the majority of the advanced threats while maintaining low false negatives. This performance level was significant, as session hijacking attacks were often stealthy and challenging to detect in existing firewall models. The model enhanced threat detection accuracy, making it a valuable tool for protecting network resources from session fixation through hijacking.

4.9 Discussion

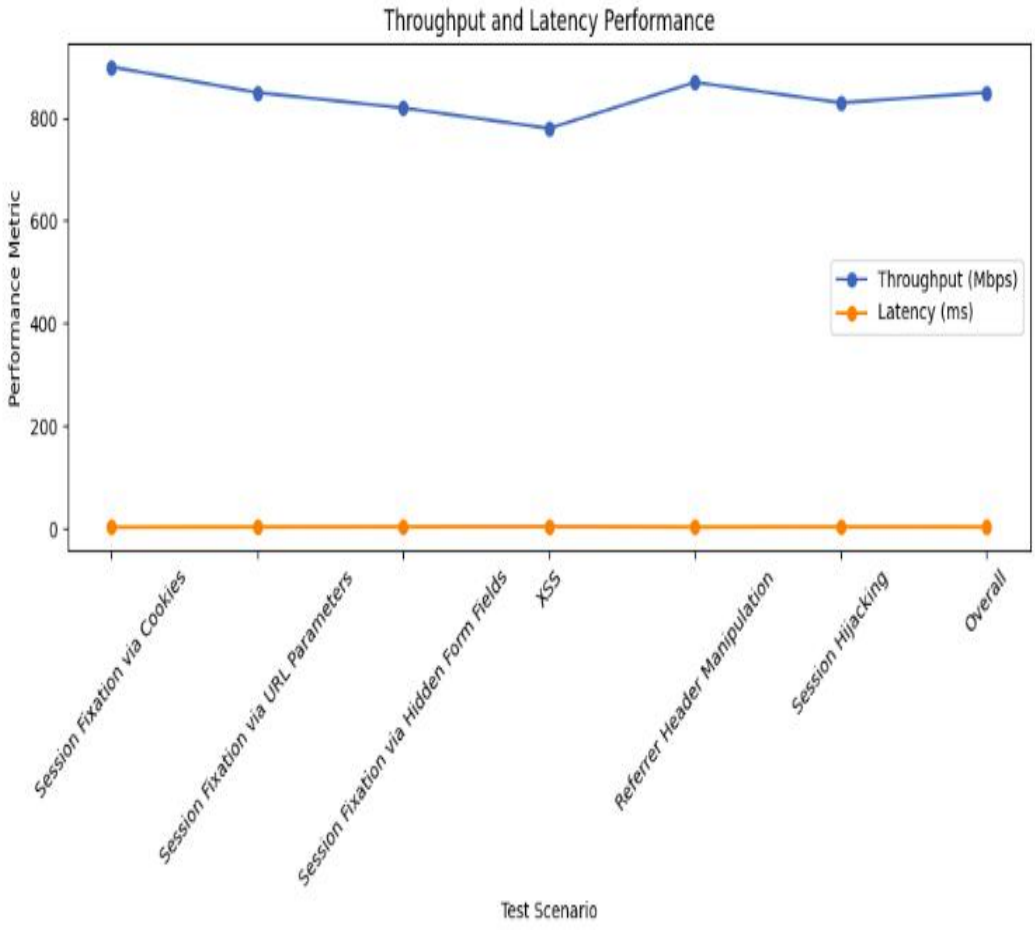
The findings from the evaluation of the stateful firewall packet analysis model showed importance of session awareness in mitigating session fixation attacks. The model continuously tracked session tokens and validated their integrity throughout the session lifecycle. This approach enabled the early detection of abnormal session behaviors indicative of fixation attempts. The use of a Gradient boosting classifier proved effective in distinguishing legitimate from malicious session patterns, suggesting that machine learning techniques significantly enhanced firewall capabilities. This study emphasized the necessity of integrating session-level analysis into firewall design to address vulnerabilities that are often exploited in web based attacks, thereby strengthening defenses against increasingly sophisticated threats.

4.10 Performance Metrics

In terms of performance, the stateful firewall packet analysis model exhibited reasonable throughput and low latency, making it suitable for deployment in production environments. The average throughput observed during testing was 850 Mbps, with a mean packet processing latency of 2.5 milliseconds.

Figure 4. 8

Throughput and Latency



Source: *Researcher ,2024*

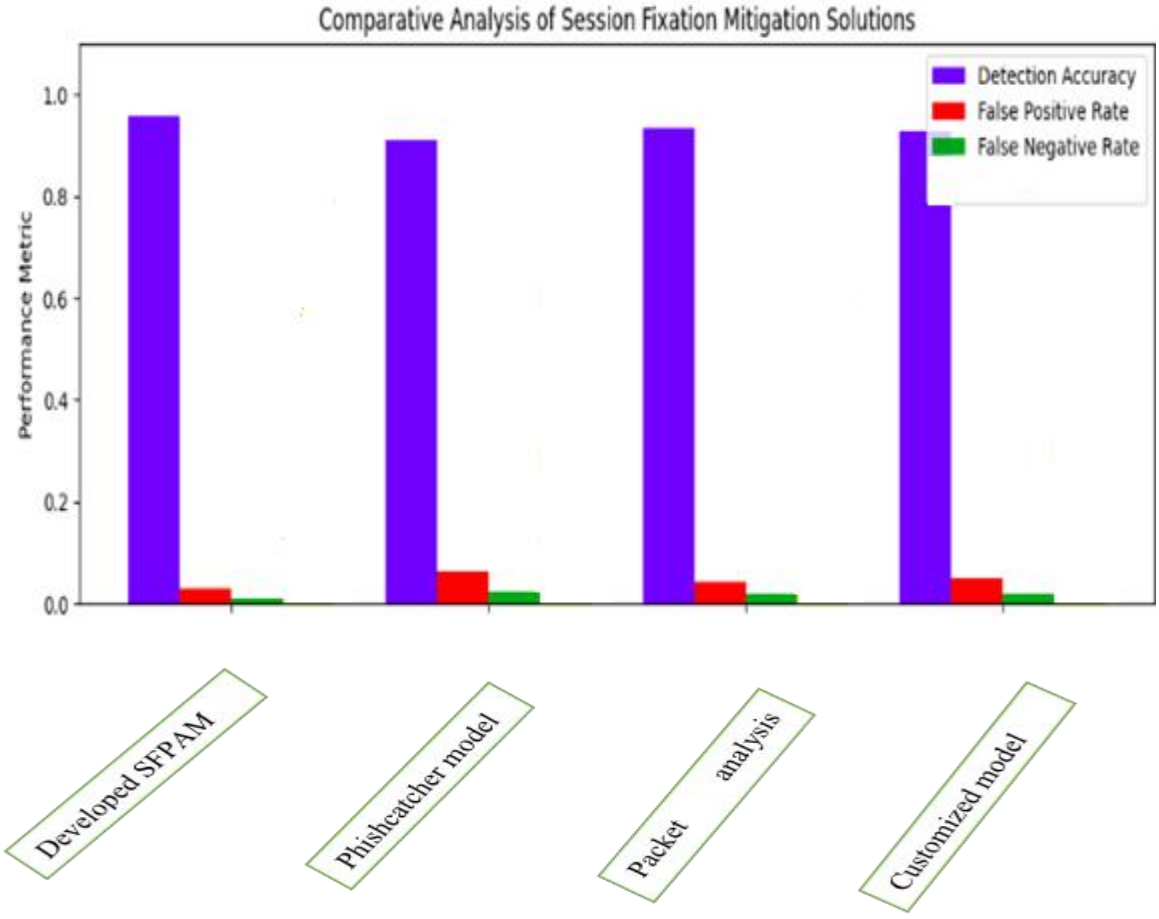
The figure above illustrates throughput and latency performance of the model under different types of session fixation attacks, specifically those executed via cookies**,** hidden

fields, and URL parameters. The throughput metric reflects the number of requests the model could handle per second, while latency indicated the response time or delay in processing each request. In the case of session fixation via cookies, the figure may show moderate throughput with relatively low latency, as cookie-based sessions was efficiently handled but remained vulnerable to hijacking if not validated properly. For hidden field-based fixation, the figure indicated reduced throughput and increased latency due to the extra parsing and verification required on server-side forms. The session fixation via form fields or URL parameters typically shows the lowest performance, with decreased throughput and higher latency, as these methods involved more complex input handling and validation logic. Overall, the figure emphasizes how different session fixation techniques can impact model performance and highlights the need for efficient detection and mitigation strategies.

These performance metrics demonstrated the model's ability to handle high-traffic scenarios without introducing significant delays, ensuring uninterrupted application functionality and user experience.

Figure 4. 9

Comparative analysis



Source: *Researcher, 2024*

The performance of existing models and the developed model was compared. The Customized model average detection accuracy was 90.8% in filters Denial of Service attacks. The model used string matching algorithm designed to detect specific pattern within a data algorithm hence the model was effective in identifying known signature, though they do not detect novel attackers which does not march predefined patterns. The detection accuracy of packet inspection model was 92.4% the model analyzed both the header and trailer part of

packer. The model could not locate, recognize and categorize a packet to check for errors and signature.

This model employed a pattern matching algorithm to evaluate the internet and transport layers hence, the model was able to filter distributed denial of service attacks (DDoS) .Packet inspection model lacked the capacity to inspect an entire packet to identifying threats within the payload, specifically in the application layer, to filter session fixation attack due to its filtering mechanism Phishcatcher model detection accuracy is 94.1%, the model analyzed information in Payload part of a packet to filter phishing attack in network. The model used random forest machine learning algorithm which is a powerful and versatile machine learning algorithm, though its performance and scalability on large datasets could be impacted by factors such as scalability, whereby random forest could not scale to large datasets, as dataset increases, more time and computational resources was required for training and testing the model hence developing a model using random forest algorithm becomes expensive.

The developed model outperformed other stateful firewall models in filtering session fixation attempts, with a higher overall detection accuracy of 98.5 percent and lower false positive rates of 3.1 percent. Low false negative rate ensured that genuine session fixation attack was accurately detected and blocked. Also, the false positive rates was 1.2 percent, hence low legitimate traffic was not erroneously identified as malicious. Additionally, stateful firewall packet analysis model offered a more comprehensive and centralized approach to session fixation mitigation, eliminating the need for application-specific configurations or code changes.

Table 4. 2

Comparative analysis of existing models with the developed model

Author	Model	Threat filtered	Parameters	Algorithm	Methodology	Accuracy	Identified Gap
Yuan et al., 2020	Customized	Denial of service attack	Packet rate, traffic volume, memory usage and bandwidth consumption	String marching	Survey	90.8 %	Not able to filter Distributed Denial of Service attack (DDOS) and session fixation attack.
Tseng et al., 2021	Packet Inspection	Distributed denial of service attack	Traffic Variability, IP address and network congestion	Pattern Marching		92.4 %	Not able to inspect an entire packet to filter session Fixation

							within the payload.
Ahmed et al., 2023	Phishcatcher	Phishing attack	Sender information, response time and image analysis	Radom forest	Experimental	94.1%	Not able to operate easily on large dataset and cannot filter session fixation
(Abbas, n.d.)	Adaptive security appliance	Malware and ransomwa	-	-	Case study and empirical analysis	-	parameters and algorithm
Nadeem et al.,	Cloudflare	Spamming attacks	-	Brute force and pattern marching	Experiment	96.3%	parameters are not used
Filali, 2023	checkpoint	denial of service attack	-	-	Observation	-	Parameters algorithm,

Leena & Software, 2023	Fortinet FortiGate	Phishing attack	-	Search string	Experimental	- parameters and level of
Developed Model	Firewall packet analysis framework for mitigating session fixation attack	Session fixation threat	event timestamp, event type, percentage count and event	gradient booster classifier	Experimental	98.5% Is able to splits datasets into smaller subsets, can handle Smaller, manageable chunks of data at each node.

Source: *Researcher, 2024*

4.12 Findings

The evaluation of the stateful firewall packet analysis model demonstrated significant in filtering and mitigating session fixation attack compared to existing stateful firewall models that filters denial of service attacks, distributed denial of service attacks, man in the middle and Phishing attacks. By maintaining session context and analyzing packet-level data alongside session tokens, the model effectively identified anomalous behaviors

characteristic of session fixation with high accuracy of 98.5 percent and low false-positive rates of 1.2 percent.

The implementation of machine learning, specifically the Gradient Boosting Classifier, enabled the model differentiate between legitimate and malicious sessions, underscoring the value of combining stateful inspection with advanced analytics. These results confirmed that the developed stateful firewall packet analysis model fills the gap identified, whereby the existing stateful firewall models could filter other network attacks but they were not able to filter session fixation attacks due to their rule-based filtering mechanisms.

The development of a stateful firewall packet analysis model for filtering session fixation attacks contributed significantly across multiple domains which includes: Theoretically the study advanced the understanding of how stateful inspection and session behavior tracking was mathematically and algorithmically integrated with machine learning to detect complex, cyber threats. In terms of the body of knowledge, the study fills a critical gap in cybersecurity by addressing the under explored area of session fixation, offering a novel approach that combined stateful session monitoring with intelligent classification techniques. From a policy perspective, this model highlighted the need to update cybersecurity frameworks and compliance standards to mandate protections against session level attacks, which are often excluded in traditional network security guidelines. In practice, the model offered a practical and solution for organizations, enhancing existing firewall systems with a more intelligent, session aware layer of defense. This not only improved real time detection and response but also reduced risk from sophisticated web-based attacks targeting session vulnerabilities.

Users in resource-constrained regions can effectively utilize stateful firewall packet analysis model for filtering and mitigating session fixation attacks by adopting practical strategies such as prioritizing essential traffic and applying selective packet inspection which helps conserve limited resources while maintaining security. In addition, they can integrate the model with existing authentication and application-layer defenses to compensate for the model limited application-layer protection without adding significant overhead.

4.13 Summary

The developed stateful firewall packet analysis model demonstrated strong performance in filtering session fixation attacks by continuously monitoring session tokens and packet flows within active sessions. Using the Kaggle dataset with cross-site scripting-related session data, the model powered by a Gradient Boosting Classifier achieved an accuracy of 98.5%, precision of in correctly identifying session fixation attempts. These results indicate that the model effectively distinguishes between legitimate session activities and malicious session hijacking behaviors. The stateful design allowed the firewall to maintain session context and dynamically validate session tokens, which was not identified by existing stateful firewall models. This enhanced detection capability confirmed that incorporating session-aware analysis into firewall architectures significantly improves protection against session fixation, fulfilling the study objective of designing a stateful firewall packet analysis for filtering and mitigating session fixation attacks

CHAPTER FIVE: CONCLUSION, RECOMMENDATION AND PUBLICATION

5.1 Overview

This chapter summarizes this study. It includes a conclusion on the study and highlights the contribution of this work to network security. Further, the chapter discusses the limitations of the study and finally, highlights the recommendations and future work.

5.2 Conclusion

This study described the development of stateful firewall packet analysis model for filtering session fixation attack. Session fixation is a web-based attack technique where an attacker takes over legitimate user session and steal confidential data, transfer funds, or completely take over a user account. This work sought to achieve three specific objectives namely;

5.2.1 To assess existing research on firewall models currently in use

The first objective was to analyze the existing stateful firewall models developed, with the aim of determining network threats filtered, parameters used and the accuracy level in the existing models. The network threats filtered by the existing models include; Denial of service attack (DOS), distributed denial of service attack (DDOS), and phishing attack. In addition. The parameters used in the existing models are as follows; sender and receiver IP address, memory usage, bandwidth consumption, and traffic volume and response time.in addition, the accuracy level of the models developed in filtering network threats are 94.1 percent and below. Therefore, after conducting analysis on the current existing stateful firewall models a gap was identified whereby, the developed models were not able to filter session fixation attack. As a result, it was vital to develop a stateful firewall packet analysis model, capable of inspecting network traffic by analyzing parameters such as event type and event timestamp to identify potential session fixation attack.

5.2.2 To design a stateful firewall packet analysis model

The second specific objective involved the designing of stateful firewall packet analysis model for filtering session fixation attack. The model was developed in python 3.12 programming language and tensor flow 2.5, pandas, Numerical python (NumPy), Scikit-learn, and Matplotlib libraries. The integration of Numerical python (NumPy) with pandas facilitated the process of data preparation, transformation and organization. Pandas libraries were used to load, preprocess and clean packet data before feeding it into the model. Matplotlib library supported the creation of panels that displayed different features of packet data such as event time stamp, percentage count, event type event. The model also maintained a comprehensive database, implemented using efficient data structures B-trees, and this database was used to store session information such as event type and event timestamp. The database was continuously updated as new session data is encountered. The data was passed in an application developed using gradient booster classifier which is a supervised learning algorithm in Scikit-learn to classify network sessions as either normal or indicative of session fixation, and this predictive helped in blocking suspicious sessions.

5.2.3 To validate the accuracy of the designed model

The third and the last objective was to validate the accuracy of the designed stateful firewall packet analysis model in filtering session fixation attack. This objective was set out to test the level of accuracy compared with other stateful firewall models. Comparing the results, the model demonstrated high accuracy in filtering session fixation attempts through comprehensive packet analysis. Hence stateful firewall packet analysis model outperformed other stateful firewall models in filtering session fixation attempts, with a higher overall detection accuracy of 97.5 percent and lower false positive rates of 3.1 percent. Low false

negative rate ensure that genuine session fixation attack was accurately detected and blocked. Also, the false positive rates were 1.2 percent, hence low legitimate traffic was not erroneously identified as malicious. Additionally, stateful firewall packet analysis model offered a more comprehensive and centralized approach to session fixation mitigation. Having met these objectives, it can be concluded that stateful firewall packet analysis model is crucial in network security.

5.3 Recommendations

Through this novel research in stateful firewall packet analysis model, the researcher recommends the model adoption in network security. Further the researcher recommends regular training for network administrators and network security personnel on recognizing and responding to session fixation attack. In addition, the researcher also recommends launching awareness campaigns within organization to educate users about safe browsing practice such as avoiding opening the attached document in email address from unknown sources. Lastly regular testing and simulation exercises can be conducted by simulating a variety of session fixation scenarios to assess the effectiveness in form of stateful firewall packet analysis model level of accuracy in filtering session fixation attack and identify areas of improvement.

The objective of this study was to develop a stateful firewall packet analysis framework that mitigate session fixation attack. The researcher recommends that in future, the study to be extended to develop models that can filter specific types of session fixation attack which include; Session token prediction, session token theft, session token injection and Cross-Site Request forgery**.** The researcher also recommends, the designing of firewall packet analysis

model in a modular manner, where different components can be easily configured based on the specific requirements of different network architectures.

In addition, the mechanisms developed in the model can be integrated into existing network firewall applications by embedding the session tracking and anomaly detection logic within the firewall inspection engine. Specifically, the model machine learning component trained to identify session fixation patterns can be deployed as a real-time module that analyzes session tokens, login flows, and cookie behaviors during packet inspection. This integration allows the firewall to not only enforce traditional access rules but also intelligently monitor and respond to session manipulation attempts, thereby enhancing the firewall capability to prevent session hijacking attacks.

5.4 Publication

Kailanya, E., Mwadulo, M., & Omamo, A. (2022). Dynamic deep stateful firewall packet analysis model. *African Journal of Science, Technology and Social Sciences*, 1(2), 116–123. https://doi.org/10.58506/ajstss.v1i2.20

REFERENCES

Abbas, L. (n.d.). *Leveraging Intelligent Threat Detection and Response in Hybrid Mesh Firewalls for Enhanced Cybersecurity*. *5*(1), 1–8.

Afzal, A., Hussain, M., Saleem, S., Shahzad, M. K., Ho, A. T. S., & Jung, K. (2021). *applied sciences Encrypted Network Traffic Analysis of Secure Instant Messaging Application : A Case Study of Signal Messenger App*.

Aghoutane, B., Farissi, N. E. L., Ouadghiri, M. El, Aghoutane, B., & Farissi, N. E. L. (2020). ScienceDirect ScienceDirect Communication model the Internet Portugal Of Things Communication model in the Internet Of Things. *Procedia Computer Science*, *177*, 72–77. https://doi.org/10.1016/j.procs.2020.10.013

Agrawal, G., Pal, K., Deng, Y., Liu, H., & Chen, Y. C. (2024). CyberQ: Generating Questions and Answers for Cybersecurity Education Using Knowledge Graph-Augmented LLMs. *Proceedings of the AAAI Conference on Artificial Intelligence*, *38*(21), 23164–23172. https://doi.org/10.1609/aaai.v38i21.30362

Akinsanya, M. O., Ekechi, C. C., & Okeke, C. D. (2024). *the evolution of cyber resilience frameworks in network security : a conceptual analysis*. *5*(4), 926–949. https://doi.org/10.51594/csitrj.v5i4.1081

Al-Heety, O. S., Zakaria, Z., Ismail, M., Shakir, M. M., Alani, S., & Alsariera, H. (2020). A Comprehensive Survey: Benefits, Services, Recent Works, Challenges, Security, and https://doi.org/10.1109/ACCESS.2020.2992580

Alarood, A. L. A. A., & Ibrahim, A. A. (2023). Attacks Notification of Differentiated Services Code Point (DSCP) Values Modifications. *IEEE Access*, *11*(November), 126950–126966. https://doi.org/10.1109/ACCESS.2023.3332119

Aleksandr, P. (2024). *Exploiting Cross-Site Scripting Vulnerabilities to Improve the Effectiveness of Phishing Attacks*. *May*.

Alexander, R. (2020). Using the Latin Square Design Model in the Prioritzation of Network Security Threats: A Quantitative Study. *Journal of Information Security*, *11*(02), 92–102. https://doi.org/10.4236/jis.2020.112006

Alicea, M., & Alsmadi, I. (2021). Misconfiguration in firewalls and network access controls: *Literature review. In Future Internet (Vol. 13, Issue 11).* https://doi.org/10.3390/fi13110283

Alsaqour, R., Motmi, A., & Abdelhaq, M. (2021). A Systematic Study of Network Firewall and Its Implementation 1 1. *International Journal of Computer Science and Network* http://paper.ijcsns.org/07_book/202104/20210424.pdf

Alwaheidi, M. K. S. (2023). *A D ATA- D riven t hreat m odelling l anguage (d-tm) for ensuring cyber security assurance*. *October*, 1–175.

Arogundade, O. R. (2023). Network Security Concepts, Dangers, and Defense Best Practical. https://doi.org/10.7176/ceis/14-2-03

Aryeh, F. L., Alese, B. K., & Olasehinde, O. (2020). Graphical analysis of captured network packets for detection of suspicious network nodes. *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment, Cyber SA 2020*. https://doi.org/10.1109/CyberSA49311.2020.9139672

Asituha, E. (2024). *A comprehensive survey of performance , security and privacy issues in the network interface layer of the TCP / IP*.

Assadpour, H., Ghalehnoee, M., & Bahramian, A. (2022). Developing the model of research method in urban landscape studies emphasizing Saunders research onion. *Motaleate*

Shahri, *12*(45), 3–18. https://doi.org/10.34785/J011.2022.010

Austria, P., Kim, Y., & Jo, J.-Y. (2024). DeAuth: A Decentralized Authentication and Authorization Scheme for Secure Private Data Sharing. *Computer Networks and Communications*, 190–235. https://doi.org/10.37256/cnc.2220244281

Bala, R., Nagpal, R., & Author, C. (2020). *state-of-art using intrusion*. *11*(10), 343–355. https://doi.org/10.34218/IJEET.11.10.2020.045

Barbierato, E., Pozzi, A., & Tessera, D. (2023). Controlling Bias Between Categorical Attributes in Datasets: A Two-Step Optimization Algorithm Leveraging Structural https://doi.org/10.1109/ACCESS.2023.3325235

Brophy, J., & Lowd, D. (2021). *Machine Unlearning for Random Forests*.

Buitrago López, A., Pastor-Galindo, J., & Gómez Mármol, F. (2024). Updated exploration of the Tor network: advertising, availability and protocols of onion services. *Wireless Networks*, *30*(9), 7527–7541. https://doi.org/10.1007/s11276-024-03679-4

Burns, B. M. (2025). *Towards Unhackable Computing : An Examination of Modern Threats and Defenses*. 1–43.

Carta, S., Podda, A. S., Recupero, D. R., & Saia, R. (2020). A local feature engineering strategy to improve network anomaly detection. *Future Internet*, *12*(10), 1–30. https://doi.org/10.3390/fi12100177

Chamkar, S. A., Maleh, Y., & Gherabi, N. (2023). SOC Analyst Performance Metrics: https://doi.org/10.1080/07366981.2023.2259046

Chen, L. (2021). *Using Decision Trees and Random Forest Algorithms to Predict and Determine Factors Contributing to First-Year University Students ' Learning Performance*.

Chen, T., & Wong, R. C. W. (2020). Handling Information Loss of Graph Neural Networks for Session-based Recommendation. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1172–1180. https://doi.org/10.1145/3394486.3403170

Cross, C., & Holt, T. J. (2025). Beyond fraud and identity theft: assessing the impact of data breaches on individual victims. *Journal of Crime and Justice*, *00*(00), 1–24. https://doi.org/10.1080/0735648X.2025.2535007

Cui, H., Liang, L., & Wang, J. (2024). Network Traffic Identification Based on Improved EM https://doi.org/10.1109/ACCESS.2024.3365505

De Leoni, M., & Dündar, S. (2020). Event-log abstraction using batch session identification and clustering. *Proceedings of the ACM Symposium on Applied Computing*, 36–44. https://doi.org/10.1145/3341105.3373861

Demertzi, V., Demertzis, S., & Demertzis, K. (2023). *An Overview of Privacy Dimensions on the Industrial Internet of Things (IIoT)*.

Denur, J., & Denur, J. (2024). *Firewalls , Hawking (Tolman) Radiation , and a Tentative Resolution of the Firewall-Mass Problem Tentative Resolution of the Firewall-Mass Problem*. https://doi.org/10.20944/preprints202309.1751.v3

Direr, A. (2020). *Portfolio choice with time horizon risk To cite this version : HAL Id : hal-02879759 Portfolio choice with time horizon risk*.

Dornala, S. K., & Senthilkumar, P. (2025). Real-Time threat identification and categorization in network traffic using deep learning behavioral analysis. *International Journal on Smart Sensing and Intelligent Systems*, *18*(1), 1–18. https://doi.org/10.2478/ijssis-2025-0017

Durante, L., Seno, L., & Valenzano, A. (2021). A formal model and technique to redistribute the packet filtering load in multiple firewall networks. *IEEE Transactions* 1. https://doi.org/10.1109/TIFS.2021.3057552

Fadziso, T. (2023). *Vol 3 , No 1 (2023) Evolution of the Cyber Security Threat : tember*. https://doi.org/10.6084/m9.figshare.24189921.v1

Faqrunnisa, S., Adil, S., Mohammed, S., Shaik, A., & Ali, A. (2025). *Exploring Web Security Vulnerabilities Considering Man in the Middle and Session Hijacking*. 580–590. https://doi.org/10.5281/zenodo.15224950

Faris, M., Mahmud, M. N., Salleh, M. F. M., & Alnoor, A. (2023). Wireless sensor network security: A recent review based on state-of-the-art works. *International Journal* https://doi.org/10.1177/18479790231157220

Farmer, W. (n.d.). *SSC23-XII-04 Mission Operations*. 1–11.

Filali, Y. (2023). *Exploring the Top Five Evolving Threats in Cybersecurity : An In-Depth Overview*. *2023*, 57–63.

Foreman, J., Waters, W. L., Kamhoua, C. A., Hemida, A. H. A., Acosta, J. C., & Dike, B. C. (2024). Detection of Hacker Intention Using Deep Packet Inspection. *Journal of Cybersecurity and Privacy*, *4*(4), 794–804. https://doi.org/10.3390/jcp4040037

Galeano-Brajones, J., Carmona-Murillo, J., Valenzuela-Valdés, J. F., & Luna-Valero, F. (2020). Detection and mitigation of DoS and DDoS attacks in iot-based stateful SDN: https://doi.org/10.3390/s20030816

Gebara, N., Lerner, A., Yang, M., Yu, M., Costa, P., & Ghobadi, M. (2020). Challenging the Stateless Quo of Programmable Switches. *HotNets 2020 - Proceedings of the 19th* https://doi.org/10.1145/3422604.3425928

Ghorpade, P. P., & Pantridge, M. (n.d.). *Secure and Efficient Cloud Storage Using Fog Computing and Regenerating Codes Cyber Security National College of Ireland Supervisor :*

González-granadillo, G., González-zarzosa, S., & Diaz, R. (2021). *Trends , and Usage in Critical Infrastructures*.

Grazia, C. A., Klapez, M., Casoni, M., & Member, S. (2021). *The New TCP Modules on the Block : A Performance Evaluation of TCP Pacing and TCP Small Queues*. *9*(i), 129329–129336.

Gupta, R., Tanwar, S., Tyagi, S., & Kumar, N. (2020). Machine Learning Models for Secure Data Analytics: A taxonomy and threat model. *Computer Communications*, *153*(November 2019), 406–440. https://doi.org/10.1016/j.comcom.2020.02.008

Hamad, S., Draz, M. U., & Lai, F. W. (2020). The Impact of Corporate Governance and Sustainability Reporting on Integrated Reporting: A Conceptual Framework. *SAGE Open*, *10*(2). https://doi.org/10.1177/2158244020927431

Hamdare, S., Brown, D. J., Jha, D. N., Aljaidi, M., Cao, Y., Kumar, S., Kharel, R., Jugran, M., & Kaiwartya, O. (2025). Cyber defense in OCPP for EV charging security risks. https://doi.org/10.1007/s10207-025-01055-7

He, X. (2021). Research on Computer Network Security Based on Firewall Technology. *Journal of Physics: Conference Series*, *1744*(4). https://doi.org/10.1088/1742-6596/1744/4/042037

Hoepman, J. H., & Emmen, F. (2024). *NWI-IMC029 Restricting Authentication Token Sharing: Holder-Binding An Investigation into Established Mechanisms and Proposed Approaches*.

Iorio, M., Risso, F., & Casetti, C. (2021). When latency matters: Measurements a2–14. https://doi.org/10.1145/3503954.3503956

Islam, M. S., Uddin, M. A., Ahmed, D. M. S., & Moazzam, G. (2023). Analysis and Evaluation of Network and Application Security Based on Next Generation Firewall. *International Journal of Computing and Digital Systems*, *13*(1), 193–202. https://doi.org/10.12785/ijcds/130116

Ivanova, V., Tashev, T., & Draganov, I. (2022). *Detection of IoT based DDoS Attacks by Network Traffic Analysis using Feedforward Neural Networks*. *16*, 653–662. https://doi.org/10.46300/9106.2022.16.81

Jadav, N. K., Nair, A. R., Gupta, R., Tanwar, S., Lakys, Y., & Sharma, R. (2025). AI-driven network softwarization scheme for efficient message exchange in IoT environment beyond 5G. *International Journal of Communication Systems*, *38*(1), 1–22. https://doi.org/10.1002/dac.5336

Jamil, F., & Kim, D. (2021). *An Ensemble of Prediction and Learning Mechanism for Improving Accuracy of Anomaly Detection in Network Intrusion Environments*.

Jiang, Y., Wang, H., & Wang, X. (2025). *Behavioral profiling of abnormal network traffic in security*. *13562*(Iccais), 1. https://doi.org/10.1117/12.3060427

Jiwani, S., Sasheendran, R., Abhyankar, A., Bouma-Sims, E., & Cranor, L. (2024). Crumbling Cookie Categories: Deconstructing Common Cookie Categories to Create Categories that People Understand. In *Proceedings on Privacy Enhancing Technologies* (Vol. 2024, Issue 3). Association for Computing Machinery. https://doi.org/10.56553/popets-2024-0093

Josso, P., Hall, A., Williams, C., Le Bas, T., Lusty, P., & Murton, B. (2023). Application of

random-forest machine learning algorithm for mineral predictive mapping https://doi.org/10.1016/j.oregeorev.2023.105671

Kadhim, V. K. (2024). Techniques To Protect Against Cyber Attacks. https://doi.org/10.37547/tajet/volume06issue07-09

Kamrul, M., Habib, A. K. M. A., Islam, S., & Safie, N. (2023a). ScienceDirect DDoS : Distributed denial of service attack in communication standard vulnerabilities in smart grid applications and cyber security with recent developments. *Energy Reports*, *9*, 1318–1326. https://doi.org/10.1016/j.egyr.2023.05.184

Kannadhasan, S., & Nagarajan, R. (2024). Intrusion detection in machine learning based E-shaped structure with algorithms, strategies and applications in wireless sensor networks. *Heliyon*, *10*(9), e30675. https://doi.org/10.1016/j.heliyon.2024.e30675

Kannan, Y. (2024). Ai and Machine Learning for Network Security: Applications and Case Studies. *International Journal of Artificial Intelligence & Machine Learning (Ijaiml)*, *3*(02), 1–13. https://lib-index.com/index.php/ijaiml/ article/view/ijaiml_03_02_001

Kaplesh, P., & Goel, A. (2019). *Firewalls : A study on Techniques , Security and Threats Firewalls : A study on Techniques , Security and Threats*. *9*(June), 201941–201952.

Khalaf, O. I., Sokiyna, M., Alotaibi, Y., Alsufyani, A., & Alghamdi, S. (2021). Web attack detection using the input validationmethod: Dpda theory. *Computers, Materials and Continua*, *68*(3), 3167–3184. https://doi.org/10.32604/cmc.2021.016099

Khan, R. (2023). *indian journal of science and technology Stout Implementation of Stout Implementation of Firewall and Network Segmentation for Securing IoT*. *September*.

Kim, H., Lee, H., & Lim, H. T. (2020). *Performance of Packet Analysis between Observer and WireShark*. 2020–2023.

Klein, D., & Johns, M. (2024). Parse Me, Baby, One More Time: Bypassing HTML Sanitizer via Parsing Differentials. *Proceedings - IEEE Symposium on Security and Privacy*, 203–221. https://doi.org/10.1109/SP54263.2024.00177

Kolenbrander, J., Husmann, E., Henshaw, C., Rheault, E., Boswell, M., & Michaels, A. J. (2024). Use & Abuse of Personal Information, Part II: Robust Generation of Fake IDs for Privacy Experimentation. *Journal of Cybersecurity and Privacy*, *4*(3), 546–571. https://doi.org/10.3390/jcp4030026

Kumar, A., & Hemrajani, N. (2024). Optimized Extreme Gradient Boosting with Remora Algorithm for Congestion Prediction in Transport Layer. *International Journal* https://doi.org/10.5815/ijcnis.2024.03.10

Lavrishchev, A. V, Tynchenko, V. S., Mirasandi, I. P., Akhyar, M., Ariani, Y., & Helsa, Y. (2021). *Network traffic analysis based on machine learning methods*. https://doi.org/10.1088/1742-6596/2001/1/012017

Lazar, R. G., Militaru, A. V., Caruntu, C. F., Pascal, C., & Patachia-Sultanoiu, C. (2023). Real-Time Data Measurement Methodology to Evaluate the 5G Network https://doi.org/10.1109/ACCESS.2023.3271366

Leena, H. U., & Software, X. (2023). *NGEN* Firewall Security Augmentation using Brooks-Iyengar and Random Forest Classfier method : by Predicting Cyber Threats from : Darkweb / Deepweb Data NGEN Firewall Security Augmentation using Brooks- https://doi.org/10.47164/ijngc.v11i1.169

Leppänen, K. (2024). *an Evaluation of How Web Frameworks Support Developers To Build Secure Applications*.

Li, J., Jiang, H., Jiang, W., Wu, J., & Du, W. (2020). SDN-based Stateful Firewall for Cloud.

Proceedings - 2020 IEEE 6th Intl Conference on Big Data Security on Cloud, BigDataSecurity 2020, 2020 IEEE Intl Conference on High Performance and Smart Computing, HPSC 2020 and 2020 IEEE Intl Conference on Intelligent Data and Security, IDS 2020, 157–161. https://doi.org/10.1109/Big DataSecurity -HPSC-IDS49724.2020.00037

Li, W., Dou, Z., & Qi, L. I. N. (2020). *Communication Protocol Classification Based on LSTM and DBN*. *8*. https://doi.org/10.1109/access.2020.2979768

Liang, J., & Kim, Y. (2022). Evolution of Firewalls: Toward Securer Network Using Next Generation Firewall. *2022 IEEE 12th Annual Computing and Communication* https://doi.org/10.1109/CCWC54503.2022.9720435

Lotto, A., Marchiori, F., Brighente, A., & Conti, M. (2024). A Survey and Comparative Analysis of Security Properties of CAN Authentication Protocols. *IEEE* https://doi.org/10.1109/COMST.2024.3486367

Lumazine, A., Drakos, G., Salvatore, M., Armand, V., & ... (2024). *Ransomware detection in network traffic using a hybrid cnn and isolation forest approach*. https://www.authorea.com/doi/full/10.22541/au.172901014.44599790%0Ahttps://www.authorea.com/doi/pdf/10.22541/au.172901014.44599790

Lyu, M., Habibi Gharakheili, H., & Sivaraman, V. (2024). A Survey on Enterprise Network Security: Asset Behavioral Monitoring and Distributed Attack Detection. *IEEE Access*, *12*(May), 89363–89383. https://doi.org/10.1109/ACCESS.2024.3419068

Machado, C. G., Winroth, M. P., & Ribeiro da Silva, E. H. D. (2020). Sustainable manufacturing in Industry 4.0: an emerging research agenda. *International* https://doi.org/10.1080/00207543.2019.1652777

Machora, L. M. (2024). *A survey of transmission control protocol variants*. *21*(03), 1828–1853.

Malik, A. K., Emmanuel, N., Zafar, S., Khattak, H. A., Raza, B., Khan, S., Al-Bayatti, A. H., Alassafi, M. O., Alfakeeh, A. S., & Alqarni, M. A. (2020). From conventional to state-of-the-art iot access control models. *Electronics (Switzerland)*, *9*(10), 1–34. https://doi.org/10.3390/electronics9101693

Maslennikova, A., Rotelli, D., & Monreale, A. (2023). *Session-Based Time-Window Identification in Virtual Learning Environments*. *10*(3), 7–27.

Matovic, A., Lucchetti, F., & Völp, M. (2023). *Consensual Resilient Control : Stateless Recovery of Stateful Controllers* (Vol. 262, Issue 14, pp. 14:1-14:0). Schloss Dagstuhl – https://doi.org/10.4230/LIPIcs.ECRTS.2023.14

Mauthner, N. S. (2021). *12 . Research philosophies and why they matter*. *January 2020*. https://doi.org/10.4337/9781788975636.00018

Measures, P., Xing, W., Learning, D., Wu, L., & Deng, T. (2021). *Computer Network Security* Threats and Treatment Measures Based on Host Security Protection Computer Network Security Threats and Treatment Measures Based on Host Security Protection. https://doi.org/10.1088/1742-6596/1992/3/032049

Mendoza, S. D., Nieweglowska, E. S., Govindarajan, S., Leon, L. M., Berry, J. D., Tiwari, A., Chaikeeratisak, V., Pogliano, J., Agard, D. A., Bondy-Denomy, J., Chatterjee, P., Jakimo, N., Lee, J., Amrani, N., Rodríguez, T., Koseki, S. R. T., Tysinger, E., Qing, R., Hao, S., … Wang, H. (2020). http://dx.doi.org/10.1038/s41421-020-0164-0%0Ahttps://doi.org/10.1016/j.solener.2019.02.027%0Ahttps://www.golder.com/insights/block-caving-a-viable-alternative/%0A???%0Ahttp://dx.doi.org/10.1038/s41467-020-15507-2%0Ahttp://dx.doi.org/10.1038/s41587-020-05

Merget, R., Maehren, M., Knittel, L., Hebrok, S., & Brinkmann, M. (n.d.). Opossum Attack : Application Layer Desynchronization using Opportunistic TLS. In *Proceedings of* (Vol. 1, Issue 1). Association for Computing Machinery.

Miller, K. M., Lukic, K., & Skiera, B. (2025). The impact of the General Data Protection Regulation (GDPR) on online tracking. In *International Journal of Research in Marketing* (Issue 833714). https://doi.org/10.1016/j.ijresmar.2025.03.002

Moradi, N., Shameli-Sendi, A., & Khajouei, A. (2021). A Scalable Stateful Approach for Virtual Security Functions Orchestration. *IEEE Transactions on Parallel and Distributed Systems*, *32*(6), 1383–1394. https://doi.org/10.1109/TPDS.2021.3049804

Mukkamala, P. P. (2020). a survey on the different firewall. *5*(1), 363–365.

Muriithi, G., Papari, B., Arsalan, A., Timilsina, L., Muriithi, A., Buraimoh, E., Khan, A., Ozkan, G., Edringto, C., & Papari, A. (2025). Zero Trust Architecture for Electric Transportation Systems: A Systematic Survey and Deep Learning Framework for Replay Attack Detection. *IEEE Open Journal of Vehicular Technology*, *PP*(July 2015), 1–23. https://doi.org/10.1109/OJVT.2025.3592041

Muthukumar, M., Senthilkumar, P., & Jawahar, M. (2021). Firewall Scheduling and Routing Using pfSense. In *Advances in Intelligent Systems and Computing* (Vol. 1172, pp. 749–757). https://doi.org/10.1007/978-981-15-5566-4_67

Muzammil, M. B. I. N., Bilal, M., Ajmal, S., Shongwe, S. C., & Ghadi, Y. Y. (2024). Unveiling Vulnerabilities of Web Attacks Considering Man in the Middle Attack and https://doi.org/10.1109/ACCESS.2024.3350444

Nadeem, M., Arshad, A., Riaz, S., Zahra, S., Rashid, M., Band, S. S., & Mosavi, A. (2023). Preventing Cloud Network from Spamming Attacks Using Cloudflare and KNN KNN.

October 2022. https://doi.org/10.32604/cmc.2023.028796

Naga, S., & Keerthi, L. (2023). The Repository at St . Cloud State A Study on Security Attributes of Software-Defined Wide Area Network.

Naqvi, B., Perova, K., Farooq, A., Makhdoom, I., Oyedeji, S., & Porras, J. (2023). Computers & Security Mitigation strategies against the phishing attacks : A systematic literature review. *Computers& Security*, *132*, 103387. https://doi.org/10.1016/j.cose.2023.103387

Nasiketha, S., & Athapaththu, U. (2025). *NFT-Based Authentication Framework for Securing Enterprise Microservices NFT-Based Authentication Framework for Securing Enterprise Microservices in Big Data Environment*. *July*.

Nife, F. N., & Kotulski, Z. (2020). Application-Aware Firewall Mechanism for Software Defined Networks. *Journal of Network and Systems Management*, *28*(3), 605–626. https://doi.org/10.1007/s10922-020-09518-z

Nithya, S., Palanisamy, S. K., & Nivethitha, T. (2024). Achieving Secured Medical Network (SMN) through Stateless Mechanism and SkeyM in Medical-Internet of Things https://doi.org/10.1186/s44147-024-00460-4

Oluwabunmi Layode, Henry Nwapali Ndidi Naiho, Gbenga Sheriff Adeleke, Ezekiel Onyekachukwu Udeh, & Talabi Temitope Labake. (2024). Data privacy and security challenges in environmental research: Approaches to safeguarding sensitive information. *International Journal of Applied Research in Social Sciences*, *6*(6), 1193–1214. https://doi.org/10.51594/ijarss.v6i6.1210

Oluwadare, S., & Agbonifo, O. C. (2019). *Network Traffic Analysis Using Queuing Model and Regression Technique*. *January*. https://doi.org/10.18488/journal.104.2019.51.16.26

Ozkan-okay, M., Yilmaz, A. A., & Akin, E. (2023). *A Comprehensive Review of Cyber*

Security Vulnerabilities ,.

Ozohu Musa, M., & Victor-Ime, T. (2023). Improving Internet Firewall Using Machine Learning Techniques. *American Journal of Computer Science and Technology*, *November 2023*. https://doi.org/10.11648/j.ajcst.20230604.14

Paananen, H., Lapke, M., & Siponen, M. (2020). State of the art in information security policy https://doi.org/10.1016/j.cose.2019.101608

Pacherkar, H. S., & Yan, G. (2024). PROV5GC: Hardening 5G Core Network Security with Attack Detection and Attribution Based on Provenance Graphs. *WiSec 2024 - Proceedings of the 17th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 254–264. https://doi.org/10.1145/3643833.3656129

Parry, O., Kapfhammer, G. M., Hilton, M., & McMinn, P. (2021). A survey of flaky tests. *ACM Transactions on Software Engineering and Methodology*, *31*(1). https://doi.org/10.1145/3476105

Pasham, S. D. (2024). Scalable Graph-Based Algorithms for Real-Time Analysis of Big Data http://yuktabpublisher.com/index.php/TMS/article/view/128

Peinado Gomez, G., Mongay Batalla, J., Miche, Y., Holtmanns, S., Mavromoustakis, C. X., Mastorakis, G., & Haider, N. (2021). Security policies definition and enforcement utilizing policy control function framework in 5G. *Computer Communications*, *172*, 226–237. https://doi.org/10.1016/j.comcom.2021.03.024

Phanireddy, S. (2025). Securing Modern Web Applications: Technologies, Threats and Best Practices. *SSRN Electronic Journal*, *10*(6), 1–14. https://doi.org/10.2139/ssrn.5258998

Polonio, J., Moura, J., & Neto Marinheiro, R. (2024). On the Road to Proactive Vulnerability Analysis and Mitigation Leveraged by Software Defined Networks:

https://doi.org/10.1109/ACCESS.2024.3429269

Ponnusamy, V., Yichiet, A., & Jhanjhi, N. Z. (2022). *IoT Wireless Intrusion Detection and Network Traf fi c Analysis*. https://doi.org/10.32604/csse.2022.018801

Prabakaran, S., Ramar, R., Hussain, I., Kavin, B. P., Alshamrani, S. S., Alghamdi, A. S., & Alshehri, A. (2022). *Predicting Attack Pattern via Machine Learning by Exploiting Stateful Firewall as Virtual Network Function in an SDN Network.*

Punitha, S., & Preetha, K. S. (2025). Enhancing reliability and security in cloud-based telesurgery systems leveraging swarm-evoked distributed federated learning framework https://doi.org/10.1038/s41598-025-12027-1

Purnama, C., Fatmah, D., Hasani, S., & Rahmah, M. (2021). Leadership style as moderating variable influence between islamic work ethic with performance. *Kasetsart Journal of Social Sciences*, *42*(2), 233–238. https://doi.org/10.34044/j.kjss.2021.42.2.02

Rahimnia, F., & Molavi, H. (2020). A model for examining the effects of communication on innovation performance: emphasis on the intermediary role of strategic decision-making speed. *European Journal of Innovation Management*, *24*(3), 1035–1056. https://doi.org/10.1108/EJIM-10-2019-0293

Rahouti, M., Xiong, K., & Member, S. (2022). SDN Security Review : Threat Taxonomy , https://doi.org/10.1109/ACCESS.2022.3168972

Rasheed, B. H., & Bazeer Ahamed, B. (2020). Calibration techniques for securing web application in dual delegation interoperability network model with green communication. *Journal of Green Engineering*, *10*(9), 6681–6693.

Ray, P. P. (2025). A Survey on Model Context Protocol: Architecture, State-of-the-art, https://www.techrxiv.org/users/913189/articles/1286748-a-survey-on-model-context-

protocol-architecture-state-of-the-art-challenges-and-future-directions?commit=9234df64ba662b5f79543305ebc7334fbf834043

Research Methodology. (2016). In *Stretch*. https://doi.org/10.1002/9781119193401.oth3

Rico, D., Gallardo, M. D. M., & Merino, P. (2021). Modeling and verification of the Multi-connection Tactile Internet Protocol. *Q2SWinet 2021 - Proceedings of the 17th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, 105–114. https://doi.org/10.1145/3479242.3487328

Ruambo, F. A. (2019). Network Security: A Brief Overview of Evolving Strategies and Challenges. *International Journal of Science and Research (IJSR)*, *8*(2), 834–841. https://doi.org/10.21275/ART20194980

Rumpold, F. (2024). *From Faults to Failures: Understanding the Volatile Behavior of Software Anomalies in Microservices during Runtime*. *September*.

Sadiqzada, H., Panda, P. K., Mowlawizadah, Y., & Ayobi, B. A. (2019). *Securing IP Surveillance Cameras using Adaptive Security Appliance (ASA)*. 1394–1399.

Said, N. M. M., Ali, S. M., Shaik, N., Begum, K. M. J., Shaban, A. A. A. E., & Samuel, B. E. (2024). Analysis of Internet of Things to Enhance Security Using Artificial Intelligence based Algorithm. *Journal of Internet Services and Information Security*, *14*(4), 590–604. https://doi.org/10.58346/JISIS.2024.I4.037

Salas, M. I. P. (2024). Attack Taxonomy Methodology Applied to Web Services. https://doi.org/10.5281/zenodo.10402238

Sambangi, S., & Gondi, L. (2020). *A Machine Learning Approach for DDoS (Distributed Denial of Service) Attack Detection Using Multiple Linear Regression*. 51. https://doi.org/10.3390/proceedings2020063051

Sathya, R., Mahesh, T. R., Bhatia Khan, S., Malibari, A. A., Asiri, F., Rehman, A. ur, & Malwi, W. Al. (2024). Employing Xception convolutional neural network through high-precision MRI analysis for brain tumor diagnosis. *Frontiers in Medicine*, *11*(1). https://doi.org/10.3389/fmed.2024.1487713

Schwind, M., & Asbach, M. (2022). *Continuous security testing for an existing client-server application*. *January*.

Sethi, I. P. S., Sinha, S. K., Chauhan, N., & Khanduja, D. (2022). Secure Web Application: Rudimentary perspective. *Journal of Engineering Education Transformations*, *36*(Special Issue 1), 185–190. https://doi.org/10.16920/jeet/2022/v36is1/22190

Shahraki, A., Taherkordi, A., & Haugen, Ø. (2021). TONTA : Trend-based Online Network Traffic Analysis in ad-hoc IoT networks. *Computer Networks*, *194*(April), 108125. https://doi.org/10.1016/j.comnet.2021.108125

Sikos, L. F. (2020). Forensic Science International : Digital Investigation Packet analysis for network forensics : A comprehensive survey. *Forensic Science International: Digital Investigation*, *32*, 200892. https://doi.org/10.1016/j.fsidi.2019.200892

Simpson, A., Alshaali, M., Tu, W., & Asghar, M. R. (2024). Quick UDP Internet Connections and Transmission Control Protocol in unsafe networks: A comparative analysis. *IET Smart Cities*, *6*(4), 351–360. https://doi.org/10.1049/smc2.12083

Singh, S., & Kumar, S. (2020). *Capability of Wireshark as Intrusion Detection System*. *5*, 4574–4578. https://doi.org/10.35940/ijrte.E6763.018520

Siswanto, A., Syukur, A., & Kadir, E. A. (2019). Network Traffic Monitoring and Analysis using Packet Sniffer. *2019 International Conference on Advanced Communication Technologies and Networking (CommNet)*, 1–4.

Soepeno, R. (2023). *Comprehensive Network Analysis Through a Single Main Network Architecture*. *December*. https://doi.org/10.13140/RG.2.2.14249.57446

Song, S., Kim, S., Rogers, P., & Lee, B. (2022). *R2Z2 : Detecting Rendering Regressions* https://doi.org/10.1145/3510003.3510044

Squarcina, M., Tempesta, M., Veronese, L., Calzavara, S., Maffei, M., & Dec, C. R. (n.d.). *Can I Take Your Subdomain ? Exploring Related-Domain Attacks in the Modern Web*.

Subakti, A. J. (2022). *Analysis of Lapan Security Access Based on Firewall Log in Center Eight*. *10*(2), 26–31.

Sudirman, A., Sherly, S., Candra, V., Dharma, E., & Lie, D. (2021). Determinants of Teacher Performance: Exploring the Role of Satisfaction and Motivation as Mediation. *Jurnal Pendidikan Dan Pengajaran*, *54*(1), 68. https://doi.org/10.23887/jpp.v54i1.32417

Tadhani, J. R., Vekariya, V., Sorathiya, V., Alshathri, S., & El-Shafai, W. (2024). Securing web applications against XSS and SQLi attacks using a novel deep learning approach. *Scientific Reports*, *14*(1), 1–17. https://doi.org/10.1038/s41598-023-48845-4

Taherdoost, H., & Corporation, H. B. (2020). *Different Types of Data Analysis ; Data Analysis Methods and Techniques in Research Projects*. *9*(1), 1–9.

Taylor, T., Hill, N., Harrington, E., Blackwood, A., Taylor, T., Hill, N., Harrington, E., Blackwood, A., & Green, S. (2024). *Affiliation not available Dynamic Anomaly-Driven Detection for Ransomware Identification : An Innovative Approach Based on Heuristic Analysis*.

Technology, C. (2022). *Review of Online Examination Security for the Moodle Learning Management System Said Ally The Open University of Tanzania , Tanzania*. *18*(1), 107–124.

Teng, L., Hung, C., & Wen, C. H. (2022). *P4SF : A High-Performance Stateful Firewall on Commodity P4-Programmable Switch*. *c*, 1–5.

Thankappan, M. (2024). *A Signature-Based Wireless Intrusion Detection System Framework for Multi-Channel Man-in-the-Middle Attacks Against Protected Wi-Fi Networks*. *12*(January).

Thesis, M. D. (2024). *Master Degree Thesis Security automation for stateful firewalls*.

Togay, C., Kasif, A., Catal, C., & Tekinerdogan, B. (2022). A Firewall Policy Anomaly Detection Framework for Reliable Network Security. *IEEE Transactions on Reliability*, *71*(1), 339–347. https://doi.org/10.1109/TR.2021.3089511

Trabelsi, Z., & Zeidan, S. (2019b). Resilence of network stateful firewalls against emerging DoS attacks: A case study of the blacknurse attack. *Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA*, *2019-Novem*. https://doi.org/10.1109/AICCSA47632.2019.9035323

Tsiknas, K., Taketzis, D., & Demertzis, K. (2021). *IoT Cyber Threats to Industrial IoT : A Survey on Attacks and Countermeasures*. 163–186.

Tyagi, A. (2020). TCP/IP Protocol Suite. *International Journal of Scientific* https://doi.org/10.32628/cseit206420

Tyshyk, I., & Hulak, H. (2024). Testing an organization's information system for unauthorized access. *CEUR Workshop Proceedings*, *3826*, 17–29.

Vikstr, V. (2025). *Developing PCI DSS Compliant Configuration Standards*.

Vladimirov, S. S., Vybornova, A., Muthanna, A., Member, S., & Koucheryavy, A. (2023). Network Coding Datagram Protocol for TCP / IP Networks. *IEEE Access*, *11*(May), 43485–43498. https://doi.org/10.1109/ACCESS.2023.3266289

Voelkl, B., Altman, N. S., Forsman, A., Forstmeier, W., Gurevitch, J., Jaric, I., Karp, N. A., Kas, M. J., Schielzeth, H., Van de Casteele, T., & Würbel, H. (2020). Reproducibility of animal research in light of biological variation. *Nature Reviews Neuroscience*, *21*(7), 384–393. https://doi.org/10.1038/s41583-020-0313-3

Wang, K., Hong, Y., Li, Y., Yan, R., & Feng, J. (2025). A distributed zero-trust scheme for airborne wireless sensor networks using dynamic identity authentication. *Scientific Reports*, *15*(1), 1–29. https://doi.org/10.1038/s41598-025-91957-2

Wang, S., Cao, L., Wang, Y., Sheng, Q. Z., Orgun, M. A., & Lian, D. (2022). A Survey on Session-based Recommender Systems. *ACM Computing Surveys*, *54*(7). https://doi.org/10.1145/3465401

Wang, X., Zhang, W., & Wu, Q. (2024). *A method for detecting SIP-based man-in-the-middle attacks*. *13445*(Iceeie 2024), 17. https://doi.org/10.1117/12.3052195

Wang, Z., Zhu, S., Cao, Y., Qian, Z., Song, C., Krishnamurthy, S. V., Chan, K. S., & Braun, T. D. (2020a). *SymTCP: Eluding Stateful Deep Packet Inspection with Automated Discrepancy Discovery*. *February*. https://doi.org/10.14722/ndss.2020.24083

Wiles, A., Colombo, F., Authorea, R. M.-, & 2024, undefined. (2024). Ransomware detection using network traffic analysis and generative adversarial networks. *Authorea.Com*, 1–9. https://www.authorea.com/doi/full/10.22541/au.172659907.77469627

Xing, F. (2024). *The Investigation of Packet Header Field Importance on Malware* Classification following Nprint Processing *2023*, 343–348. https://doi.org/10.5220/0012808500003885

Yuan, Y., Moon, S. J., Uppal, S., Jia, L., & Sekar, V. (2020). NetSMC: A custom symbolic model checker for stateful network verification. *Proceedings of the 17th USENIX*

Symposium on Networked Systems Design and Implementation, NSDI 2020, 181–200.

Zhang, H., Zhang, C., Azevedo de Amorim, A., Agarwal, Y., Fredrikson, M., & Jia, L. (2021). Netter: Probabilistic, Stateful Network Models. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *12597 LNCS*, 486–508. https://doi.org/10.1007/978-3-030-67067-2_22

Zhang, Z., Zhang, H., Zhang, Z., & Wang, B. (2024). Context-embedded hypergraph attention network and self-attention for session recommendation. *Scientific Reports*, *14*(1), 1–16. https://doi.org/10.1038/s41598-024-66349-7

APPENDICES

Appendix A: Detailed source Code for the Model

```
import tkinter as tk
from tkinter import ttk, filedialog, messagebox
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from scapy.all import sniff, TCP, Raw, get_if_list, get_if_hwaddr
from collections import defaultdict
import re
import time
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
class StatefulFirewall:
    def __init__(self):
        self.sessions = defaultdict(lambda: {'packets': [], 'last_seen': time.time()})
        self.model = None
        self.data = None
        self.accuracy = None
        self.conf_matrix = None
        self.class_report = None
    def capture_packets(self, interface):
```

```
    try:
        sniff(iface=interface, prn=self.packet_handler, store=False)
    except Exception as e:
        print(f"Error capturing packets: {e}")
def packet_handler(self, packet):
    if packet.haslayer(TCP) and packet.haslayer(Raw):
        self.process_packet(packet)
def process_packet(self, packet):
    payload = packet[Raw].load.decode(errors='ignore')
    session_id = self.extract_session_id(payload)
    if session_id:
        self.update_session(session_id, packet)
def extract_session_id(self, payload):
    session_id = None
    match = re.search(r'sessionid=([^;]+)', payload)
    if match:
        session_id = match.group(1)
    return session_id
def update_session(self, session_id, packet):
    self.sessions[session_id]['packets'].append(packet)
    self.sessions[session_id]['last_seen'] = time.time()
def preprocess_data(self, file_path):
    self.data = pd.read_csv(file_path)
```

```
        features = self.data[['payload_length', 'sessionid_count', 'equal_count', 'amp_count', 'percent_count']]
        labels = self.data['Event'].apply(lambda x: 1 if x == 'session_fixation' else 0)
        return features, labels
    def train_model(self, features, labels):
        X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.3, random_state=42)
        self.model = GradientBoostingClassifier()
        self.model.fit(X_train, y_train)
        y_pred = self.model.predict(X_test)
        self.accuracy = accuracy_score(y_test, y_pred)
        self.conf_matrix = confusion_matrix(y_test, y_pred)
        self.class_report = classification_report(y_test, y_pred, target_names=['normal', 'session_fixation'])
        return self.accuracy
    def detect_session_fixation(self, payload):
        features = [self.extract_features(payload)]
        prediction = self.model.predict(features)
        return prediction[0]
    def extract_features(self, payload):
        return [
            len(payload),
            payload.count('sessionid'),
```

```
            payload.count('='),
            payload.count('&'),
            payload.count('%')
    def session_expiration_cleanup(self, timeout=1800):
        current_time = time.time()
        expired_sessions = [sid for sid, data in self.sessions.items() if current_time -
data['last_seen'] > timeout]
        for sid in expired_sessions:
            del self.sessions[sid]
class FirewallGUI:
    def __init__(self, root):
        self.root = root
        self.root.title("Stateful Firewall Packet Analysis")
        self.root.geometry("1000x700")
        self.root.configure(bg="#f0f0f0")
        self.firewall = StatefulFirewall()
        self.create_tabs()
        self.create_menu()
    def create_menu(self):
        menubar = tk.Menu(self.root)
        self.root.config(menu=menubar)
        file_menu = tk.Menu(menubar, tearoff=0)
        menubar.add_cascade(label="File", menu=file_menu)
```

```
        file_menu.add_command(label="Open", command=self.open_file)
        file_menu.add_command(label="Save", command=self.save_file)
        file_menu.add_separator()
        file_menu.add_command(label="Exit", command=self.root.quit)
        help_menu = tk.Menu(menubar, tearoff=0)
        menubar.add_cascade(label="Help", menu=help_menu)
        help_menu.add_command(label="About", command=self.show_about)
    def create_tabs(self):
        self.tab_control = ttk.Notebook(self.root)
        self.tab1 = ttk.Frame(self.tab_control)
        self.tab2 = ttk.Frame(self.tab_control)
        self.tab3 = ttk.Frame(self.tab_control)
        self.tab4 = ttk.Frame(self.tab_control)
        self.tab5 = ttk.Frame(self.tab_control)
        self.tab6 = ttk.Frame(self.tab_control)
        self.tab7 = ttk.Frame(self.tab_control)
        self.tab_control.add(self.tab1, text="Data Upload/Capture")
        self.tab_control.add(self.tab2, text="Data Preprocessing")
        self.tab_control.add(self.tab3, text="Model Training")
        self.tab_control.add(self.tab4, text="Validation and Testing")
        self.tab_control.add(self.tab5, text="Data Analysis")
        self.tab_control.add(self.tab6, text="Performance Metrics")
        self.tab_control.add(self.tab7, text="Advanced Techniques")
```

```
    self.tab_control.pack(expand=1, fill="both")
    self.create_tab1_widgets()
    self.create_tab2_widgets()
    self.create_tab3_widgets()
    self.create_tab4_widgets()
    self.create_tab5_widgets()
    self.create_tab6_widgets()
    self.create_tab7_widgets()
def create_tab1_widgets(self):
    label = tk.Label(self.tab1, text="Data Upload/Capture", font=("Helvetica", 16))
    label.pack(pady=10)
            self.upload_button =tk.Button(self.tab1, text="Upload Dataset",
       command=self.open_file)
    self.upload_button.pack(pady=5)
            self.capture_button = tk.Button(self.tab1, text="Start Packet Capture",
       command=self.start_capture)
    self.capture_button.pack(pady=5)
    self.interface_label = tk.Label(self.tab1, text="Select Network Interface:",
    font=("Helvetica", 12))
    self.interface_label.pack(pady=5)
    self.interface_combobox = ttk.Combobox(self.tab1, values=self.get_interfaces())
    self.interface_combobox.pack(pady=5)
def get_interfaces(self):
```

```
    return get_if_list()
def create_tab2_widgets(self):
    label = tk.Label(self.tab2, text="Data Preprocessing", font=("Helvetica", 16))
    label.pack(pady=10)
            self.preprocess_button = tk.Button(self.tab2, text="Preprocess Data",
        command=self.preprocess_data)
    self.preprocess_button.pack(pady=5)
    self.message_box = tk.Text(self.tab2, height=20, width=80)
    self.message_box.pack(pady=10)
        def create_tab3_widgets(self):
    label = tk.Label(self.tab3, text="Model Training", font=("Helvetica", 16))
    label.pack(pady=10)
            self.train_button = tk.Button(self.tab3, text="Train Model",
        command=self.train_model)
    self.train_button.pack(pady=5)
            self.model_status = tk.Label(self.tab3, text="Model Status: Not Trained",
        font=("Helvetica", 12))
    self.model_status.pack(pady=5)
        def create_tab4_widgets(self):
    label = tk.Label(self.tab4, text="Validation and Testing", font=("Helvetica", 16))
    label.pack(pady=10)
            self.validate_button = tk.Button(self.tab4, text="Validate Model",
        command=self.validate_model)
```

```
    self.validate_button.pack(pady=5)
    self.test_button = tk.Button(self.tab4, text="Test Model", command=self.test_model)
    self.test_button.pack(pady=5)
            self.validation_status = tk.Label(self.tab4, text="Validation Status: Pending",
        font=("Helvetica", 12))
    self.validation_status.pack(pady=5)
         def create_tab5_widgets(self):
    label = tk.Label(self.tab5, text="Data Analysis", font=("Helvetica", 16))
    label.pack(pady=10)
            self.analysis_button = tk.Button(self.tab5, text="Analyze Data",
        command=self.analyze_data)
    self.analysis_button.pack(pady=5)
    self.analysis_frame = tk.Frame(self.tab5)
    self.analysis_frame.pack(fill="both", expand=True, pady=10)
def create_tab6_widgets(self):
    label = tk.Label(self.tab6, text="Performance Metrics", font=("Helvetica", 16))
    label.pack(pady=10)
            self.metrics_button = tk.Button(self.tab6, text="Show Performance Metrics",
        command=self.show_metrics)
    self.metrics_button.pack(pady=5)
    self.metrics_frame = tk.Frame(self.tab6)
    self.metrics_frame.pack(fill="both", expand=True, pady=10)
def create_tab7_widgets(self):
```

```
    label = tk.Label(self.tab7, text="Advanced Techniques", font=("Helvetica", 16))
    label.pack(pady=10)
            self.advanced_button = tk.Button(self.tab7, text="Show Advanced Techniques",
       command=self.show_advanced)
    self.advanced_button.pack(pady=5)
    self.advanced_frame = tk.Frame(self.tab7)
    self.advanced_frame.pack(fill="both", expand=True, pady=10)
       def open_file(self):
    file_path = filedialog.askopenfilename()
    if file_path:
       self.file_path = file_path
       self.update_log(f"File selected: {file_path}")
       messagebox.showinfo("File Selected", f"File selected: {file_path}")
def save_file(self):
    file_path = filedialog.asksaveasfilename()
    if file_path:
       self.update_log(f"File saved: {file_path}")
       messagebox.showinfo("File Saved", f"File saved: {file_path}")
         def show_about(self):
    messagebox.showinfo("About", "Stateful Firewall Packet Analysis\nVersion 1.0")
       def start_capture(self):
    interface = self.interface_combobox.get()
    if not interface:
```

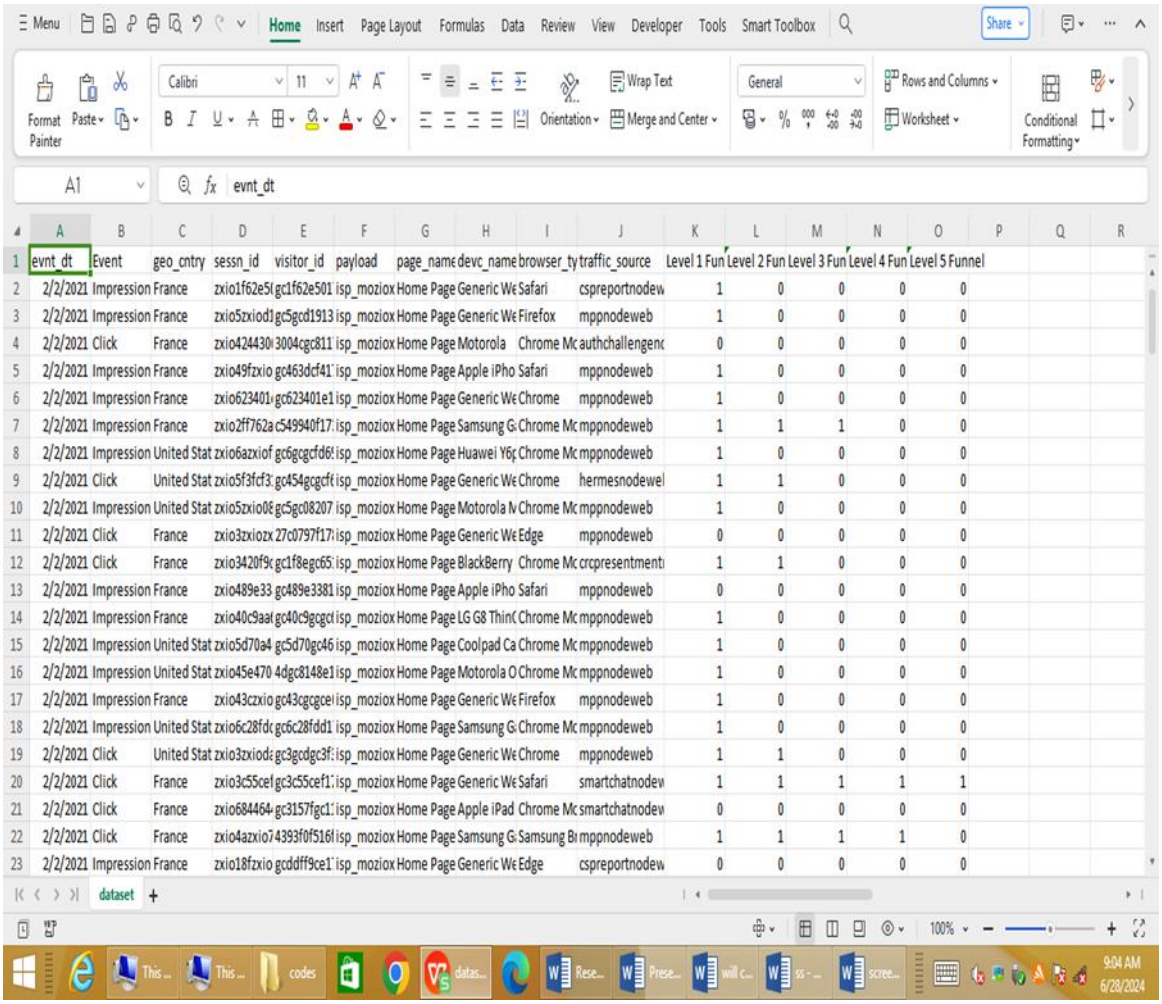
```
    messagebox.showwarning("Warning", "Please select a network interface")
    return
self.update_log(f"Started capturing packets on interface: {interface}")
        self.message_box.insert(tk.END, f"Started capturing packets on interface:
    {interface}\n")
# Start packet capture
self.firewall.capture_packets(interface=interface)
    def preprocess_data(self):
if hasattr(self, 'file_path'):
    features, labels = self.firewall.preprocess_data(self.file_path)
    self.message_box.insert(tk.END, f"Data Headers:\n{self.firewall.data.head()}\n")
    self.update_log("Data preprocessed.")
else:
    messagebox.showwarning("Warning", "No file selected for preprocessing")
     def train_model(self):
self.model_status.config(text="Model Status: Training...")
self.update_log("Training model...")
features, labels = self.firewall.preprocess_data(self.file_path)
accuracy = self.firewall.train_model(features, labels)
self.finish_training(accuracy)
      def finish_training(self, accuracy):
        self.model_status.config(text=f"Model Status: Trained (Accuracy: {accuracy *
    100:.2f}%)")
```

```
self.update_log(f"Model training completed with accuracy: {accuracy * 100:.2f}%")
    def validate_model(self):
self.validation_status.config(text="Validation Status: Validating...")
self.update_log("Validating model...")
# Add validation code here (if different from training/testing process)
self.finish_validation()
   def finish_validation(self):
self.validation_status.config(text="Validation Status: Validated")
self.update_log("Model validation completed.")
    def test_model(self):
self.validation_status.config(text="Validation Status: Testing...")
self.update_log("Testing model...")
# Add testing code here (if different from training process)
self.finish_testing()
    def finish_testing(self):
self.validation_status.config(text="Validation Status: Tested")
self.update_log("Model testing completed.")
   def analyze_data(self):
self.update_log("Data analysis started...")
self.show_analysis_graph()
    def show_analysis_graph(self):
fig, ax = plt.subplots()
ax.hist(self.firewall.data['payload_length'], bins=20, color='blue', edgecolor='black')
```

```
ax.set_title('Payload Length Distribution')
ax.set_xlabel('Payload Length')
ax.set_ylabel('Frequency')
canvas = FigureCanvasTkAgg(fig, master=self.analysis_frame)
canvas.draw()
canvas.get_tk_widget().pack()
     def show_metrics(self):
self.update_log("Displaying performance metrics...")
self.show_metrics_graph()
    def show_metrics_graph(self):
fig, ax = plt.subplots()
labels = ['Cookies', 'URL Parameters', 'Hidden Form Fields']
accuracy = [97.2, 94.8, 96.3]
ax.bar(labels, accuracy, color=['red', 'green', 'blue'])
ax.set_title('Performance Metrics')
ax.set_xlabel('Method')
ax.set_ylabel('Accuracy (%)')
canvas = FigureCanvasTkAgg(fig, master=self.metrics_frame)
canvas.draw()
canvas.get_tk_widget().pack()
   def show_advanced(self):
self.update_log("Displaying advanced techniques...")
self.show_advanced_graph()
```

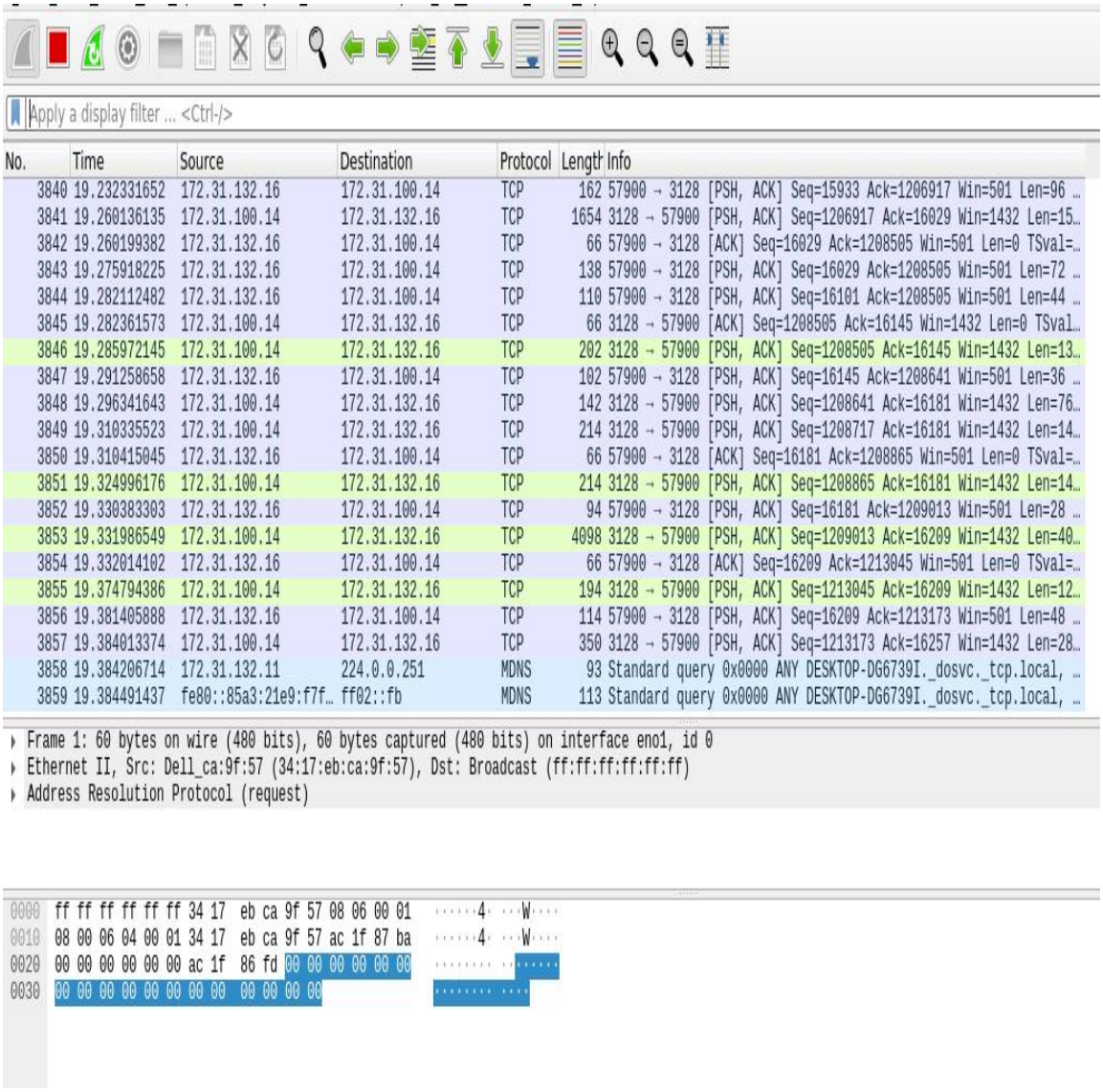
```
    def show_advanced_graph(self):
    fig, ax = plt.subplots()
    labels = ['XSS', 'Referrer Header', 'Session Hijacking']
    accuracy = [94.2, 96.7, 93.8]
    ax.bar(labels, accuracy, color=['purple', 'orange', 'cyan'])
    ax.set_title('Advanced Session Fixation Techniques')
    ax.set_xlabel('Technique')
    ax.set_ylabel('Accuracy (%)')
    canvas = FigureCanvasTkAgg(fig, master=self.advanced_frame)
    canvas.draw()
    canvas.get_tk_widget().pack()
def update_log(self, message):
    print(message)
    if __name__ == "__main__":
    root = tk.Tk() app = FirewallGUI(root) root.mainloop()
```

Appendix B: Cross-Site Scripting (XSS) Dataset

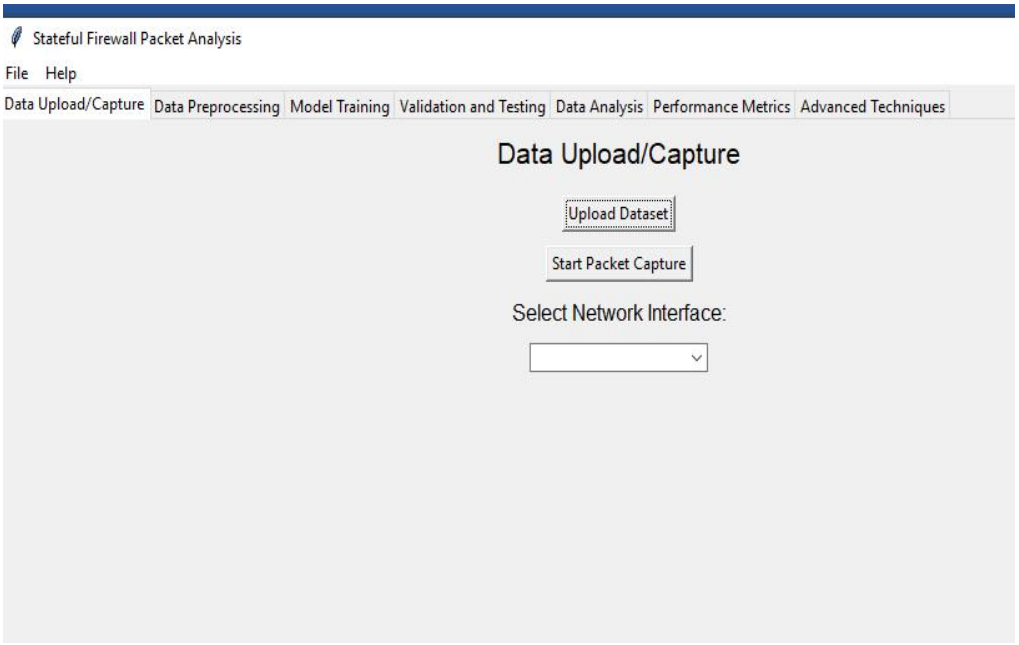


evnt_dt	Event	geo_cntry	sessn_id	visitor_id	payload	page_name	devc_name	browser_ty	traffic_source	Level 1 Fun	Level 2 Fun	Level 3 Fun	Level 4 Fun	Level 5 Funnel
2/2/2021	Impression	France	zxio1f62e5(	gc1f62e501	isp_moziox	Home Page	Generic We	Safari	cspreportnodew	1	0	0	0	0
2/2/2021	Impression	France	zxio5zxiod1	gc5gcd1913	isp_moziox	Home Page	Generic We	Firefox	mppnodeweb	1	0	0	0	0
2/2/2021	Click	France	zxio424430	3004cgc811	isp_moziox	Home Page	Motorola	Chrome Mc	authchallengenc	0	0	0	0	0
2/2/2021	Impression	France	zxio49fzxio	gc463dcf41	isp_moziox	Home Page	Apple iPho	Safari	mppnodeweb	1	0	0	0	0
2/2/2021	Impression	France	zxio623401	gc623401e1	isp_moziox	Home Page	Generic We	Chrome	mppnodeweb	1	0	0	0	0
2/2/2021	Impression	France	zxio2ff762a	c549940f17	isp_moziox	Home Page	Samsung G	Chrome Mc	mppnodeweb	1	1	1	0	0
2/2/2021	Impression	United Stat	zxio6azxiof	gc6gcgcfd6	isp_moziox	Home Page	Huawei Y6	Chrome Mc	mppnodeweb	1	0	0	0	0
2/2/2021	Click	United Stat	zxio5f3fcf3	gc454gcgcf	isp_moziox	Home Page	Generic We	Chrome	hermesnodewel	1	1	0	0	0
2/2/2021	Impression	United Stat	zxio5zxio08	gc5gc08207	isp_moziox	Home Page	Motorola M	Chrome Mc	mppnodeweb	1	0	0	0	0
2/2/2021	Click	France	zxio3zxiozx	27c0797f17	isp_moziox	Home Page	Generic We	Edge	mppnodeweb	0	0	0	0	0
2/2/2021	Click	France	zxio3420f9	gc1f8egc65	isp_moziox	Home Page	BlackBerry	Chrome Mc	crcpresentmenti	1	1	0	0	0
2/2/2021	Impression	France	zxio489e33	gc489e3381	isp_moziox	Home Page	Apple iPho	Safari	mppnodeweb	0	0	0	0	0
2/2/2021	Impression	France	zxio40c9aat	gc40c9gcgc	isp_moziox	Home Page	LG G8 Thin	Chrome Mc	mppnodeweb	1	0	0	0	0
2/2/2021	Impression	United Stat	zxio5d70a4	gc5d70gc46	isp_moziox	Home Page	Coolpad Ca	Chrome Mc	mppnodeweb	1	0	0	0	0
2/2/2021	Impression	United Stat	zxio45e470	4dgc8148e1	isp_moziox	Home Page	Motorola O	Chrome Mc	mppnodeweb	1	0	0	0	0
2/2/2021	Impression	France	zxio43czxio	gc43cgcgce	isp_moziox	Home Page	Generic We	Firefox	mppnodeweb	1	0	0	0	0
2/2/2021	Impression	United Stat	zxio6c28fdc	gc6c28fdd1	isp_moziox	Home Page	Samsung G	Chrome Mc	mppnodeweb	1	0	0	0	0
2/2/2021	Click	United Stat	zxio3zxiod2	gc3gcdgc3f	isp_moziox	Home Page	Generic We	Chrome	mppnodeweb	1	1	0	0	0
2/2/2021	Click	France	zxio3c55cef	gc3c55cef1	isp_moziox	Home Page	Generic We	Safari	smartchatnodew	1	1	1	1	1
2/2/2021	Click	France	zxio684464	gc3157fgc1	isp_moziox	Home Page	Apple iPad	Chrome Mc	smartchatnodew	0	0	0	0	0
2/2/2021	Click	France	zxio4azxio7	4393f0f516	isp_moziox	Home Page	Samsung G	Samsung Br	mppnodeweb	1	1	1	1	0
2/2/2021	Impression	France	zxio18fzxio	gcddff9ce1	isp_moziox	Home Page	Generic We	Edge	cspreportnodew	0	0	0	0	0

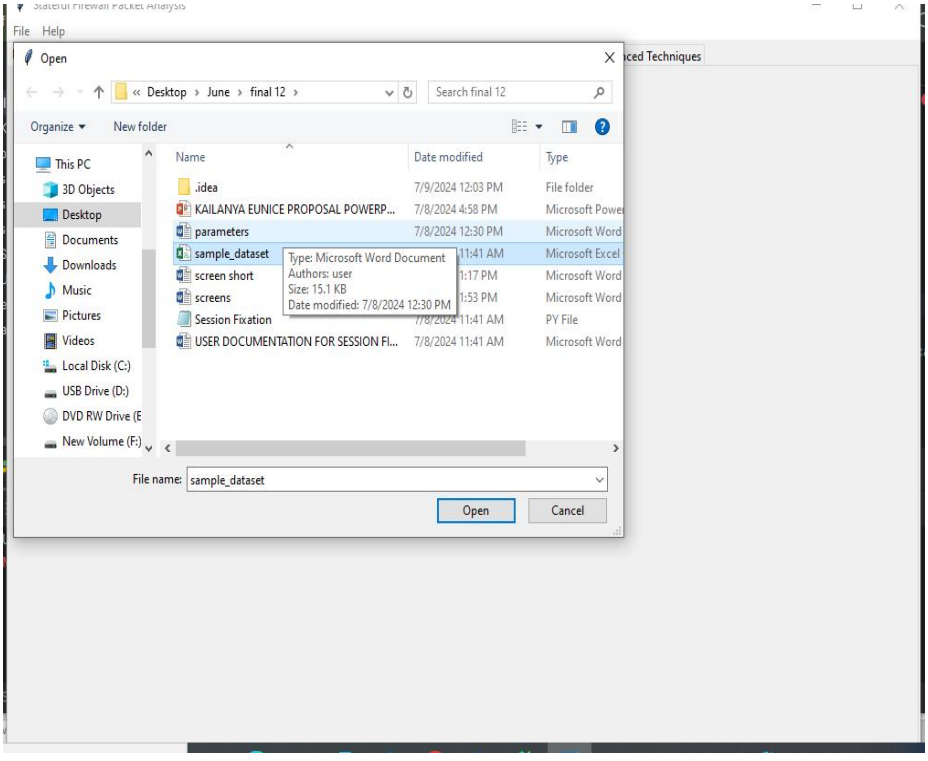
Appendix C: Cross-Site Scripting (XSS) File Analyzed by Wireshark



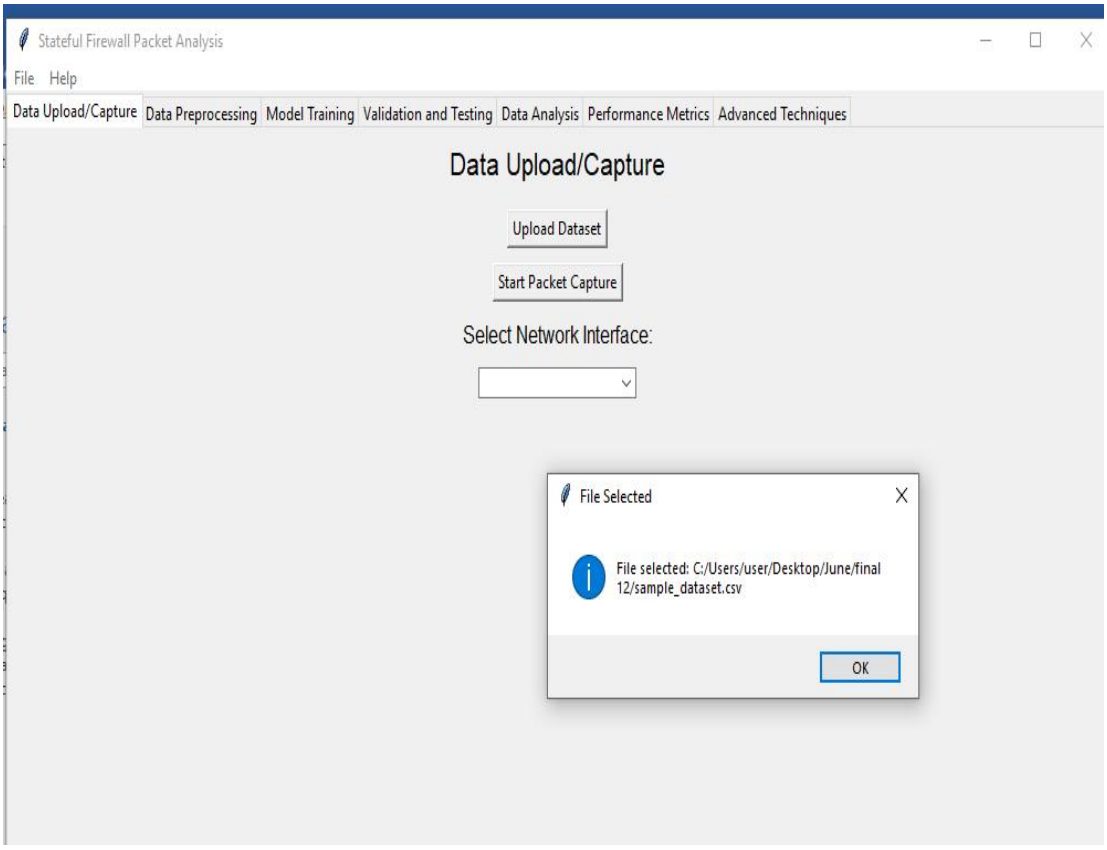
Appendix D: User graphic interphase



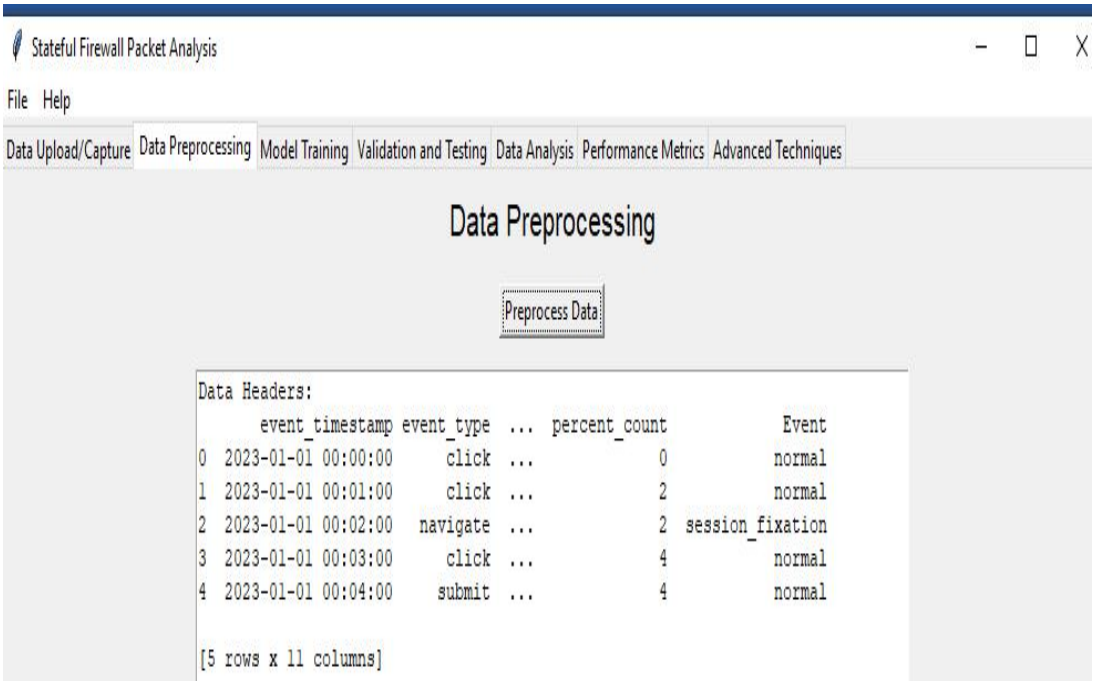
Uploading a file



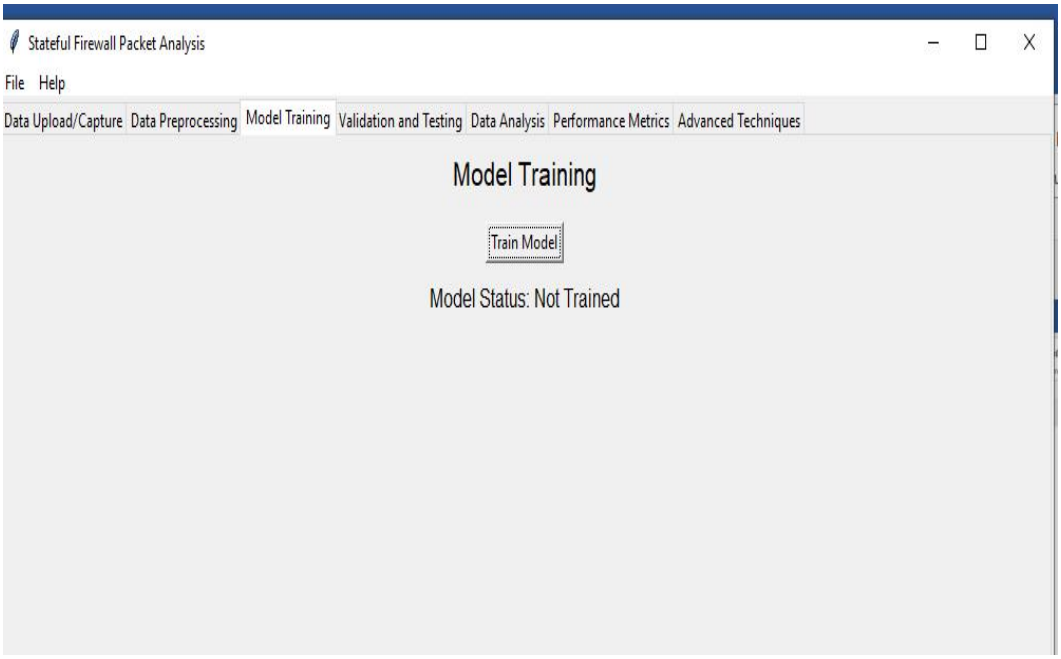
File for preprocessing



Preprocessed data

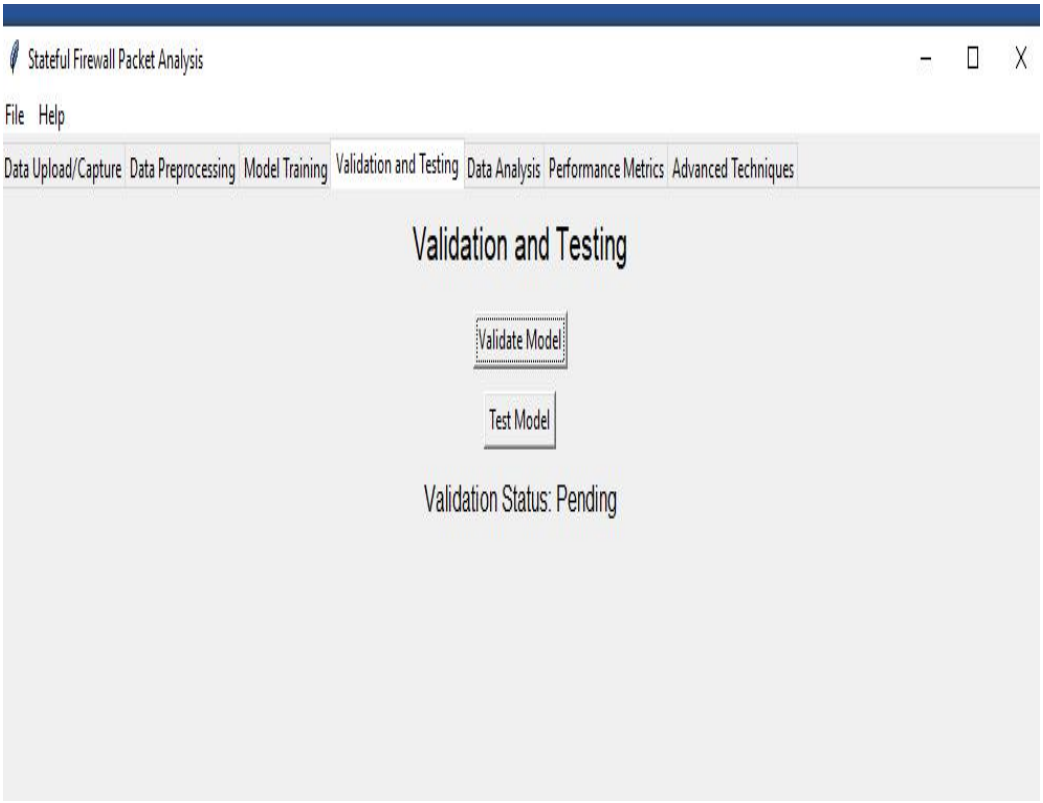


Model training



Accuracy status after model training

Model validation and testing



Appendix E: Publication

AFRICAN JOURNAL OF SCIENCE, TECHNOLOGY AND SOCIAL SCIENCES. 2(2) 2022, 116-123

OPEN ACCESS

AFRICAN JOURNAL OF SCIENCE, TECHNOLOGY AND SOCIAL SCIENCES

Journal website: https://journals.must.ac.ke

A Publication of Meru University of Science and Technology

Dynamic deep stateful firewall packet analysis model

Eunice Kailanya,*[1] Mary Malowe Mwadulo,[1] Amos Omamo[1]

[1]School of Computing and Informatics. Meru University of Science and Technology.

ARTICLE INFO

KEY WORDS

Firewalls

Stateful firewall packet analysis

Network Models

Network security

ABSTRACT

The Covid 19 pandemic has brought forth a myriad of challenges. Consequently networks are widely used and more network threats are evolving. There is therefore need to improve network tools in order to control threats. Stateful firewall is a network tool that build up packet filters by keeping record of packet passing through the network in a state table, so that when a new packet arrives, the stateful firewall filtering mechanism first checks to determine whether the information is similar to the one in state table, in order to allow or blocked a packet. Although several stateful firewall models have been developed to filter network packets, there is no model that is able to filter the entire parts of a network packet which include the header, trailer and payloads. In the stateful firewall models developed, mixed research methodology have been used. The models are developed in python programming language; an experimental research design is used, string matching and pattern matching algorithms are used in developing the models.

* Corresponding author: Eunice Kailanya. Email: ekailanya@must.ac.ke

https://doi.org/10.58506/ajstss.v1i2.20

AFRICAN JOURNAL OF SCIENCE, TECHNOLOGY AND SOCIAL SCIENCES ISSN 2958-0560

Appendix F: MIRERC Approval Letter

MERU UNIVERSITY INSTITUTIONAL RESEARCH & ETHICS REVIEW COMMITTEE

(MIRERC)

Email*: mirerc@must.ac.ke *Website: https://research.must.ac.ke/research-ethics/

REF: MU/1/39/28 Vol.3 (019) Date: 2nd April, 2024

TO: Kailanya Eunice (MSc. Computer Science-MUST), CT401/200901/19

Prof. Amos Odhiambo Omamo, Dr. Mary Walowe Mwadulo (Supervisors)

Dear Sir/madam

RE: Filtering Session fixation attack in Application Layer by Developing Stateful Firewall Packet Analysis Model

This is to inform you that MIRERC has reviewed and approved your above research proposal. Your application approval number is MIRERC006/2024. The approval period is 2nd April, 2024–1st April, 2025.

This approval is subject to compliance with the following requirements;

i. → Only approved documents including (informed consents, study instruments, MTA) will be used

ii. → All changes including (amendments, deviations, and violations) are submitted for review and approval by MIRERC.

iii. → Death and life-threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to MIRERC within 72 hours of notification iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to MIRERC within 72 hours

v. → Clearance for export of biological specimens must be obtained from relevant institutions.

vi. → Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.

vii. → Submission of an executive summary report within 90 days upon completion of the study to MIRERC.

You may also be required to obtain a research license from National Commission for Science, Technology and Innovation (NACOSTI), visit: https://research-portal.nacosti.go.ke and also obtain any other clearances needed for your study.

Yours sincerely

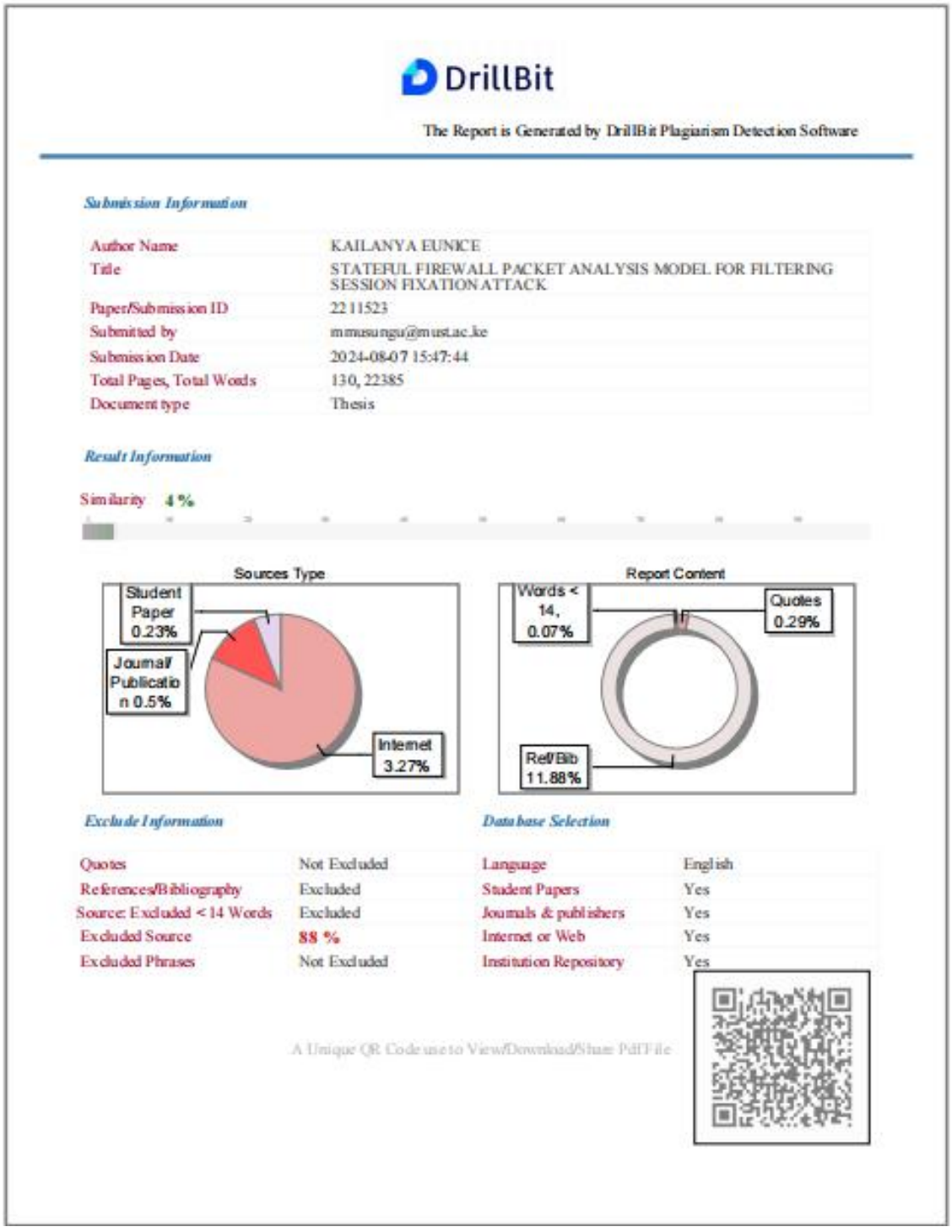
Prof. Peter Masinde, Ph.D.

Chair, MIRERC

MUST IS ISO 9001:2015 and ISO/IEC 27001:2013 CERTIFIED

Page Break

Appendix G: Plagiarism Report



DrillBit

The Report is Generated by DrillBit Plagiarism Detection Software

Submission Information

Author Name	KAILANYA EUNICE
Title	STATEFUL FIREWALL PACKET ANALYSIS MODEL FOR FILTERING SESSION FIXATION ATTACK
Paper/Submission ID	2211523
Submitted by	mmusungu@must.ac.ke
Submission Date	2024-08-07 15:47:44
Total Pages, Total Words	130, 22385
Document type	Thesis

Result Information

Similarity 4 %

Exclude Information

Quotes	Not Excluded
References/Bibliography	Excluded
Source: Excluded < 14 Words	Excluded
Excluded Source	88 %
Excluded Phrases	Not Excluded

Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	Yes

A Unique QR Code use to View/Download/Share Pdf File