Enhancement WAF Model using Deep Learning and Machine Learning Techniques

Supervised by: Dr. Heba Osama, Eng. Saja Tarek

Monica Medhat Maurice Gendy, Martina Milad Ragheb Bishay, Samer Raafat Samir Youssef, Kevin Emad Rizk Abdelmessih

November 2023

Proposal Version	Date	Reason for Change
1.0	15-November-2023	Proposal First version's specifications are defined

Table 1: Document version history

GitHub: https://github.com/MonicaMedhat20/Web-Application-Firewall-WAF-Enhancement-.git

Abstract

Nowadays, due to the spread and growth of the technology worldwide, this has led to the increase in the number and types of web attacks which results in various security issues for internet users. Web Application Firewall (WAF) has been introduced to block and filter these web attacks by blocking any malicious actions or requests that the website might be exposed to. In addition WAF helps in protecting web applications from a variety of application layer attacks such as cross-site scripting (XSS), and SQL injection. Our target and added value in this project is to enhance the performance and the run time of the proposed WAF by using Machine learning and Deep learning techniques. Not only by comparing similar algorithms with each other; but also by comparing different algorithms with each other (comparing machine learning algorithms with deep learning algorithms) in order to achieve higher accuracy.

1 Introduction

In the last decade technology has been evolving in a huge way which has seen a vast amount of web applications that deliver services over the internet and enable organizations to offer rich services over the internet. It's generally known that one of the common difficulties in the field of delivering services over the Internet is protecting computers and networks. However, this has introduced the concept of cyberattacks and the challenges in securing web applications against various numbers of attacks which raise several security problems for internet users. In fact, this has introduced the concept of Web Application Firewalls (WAFs) to enhance security against various types of attacks and malicious requests. [8]

1.1 Background

Web applications are now the primary targets for cyber-attacks causing financial losses and distribution of services. Web Application Firewalls help to protect web applications from various attacks such as cross-site scripting (XSS) where malicious scripts can be executed randomly by an attacker by injecting data into a web page [10], and SQL injection; in these attacks, a server's SQL queries are maliciously executed, allowing unauthorized users to access and retrieve restricted data kept in databases [12]. In addition, WAF enhances the performance of traditional WAF in detecting and preventing web attacks, and addresses the limitations of traditional WAF in detecting new and unknown attacks, these traditional WAFs typically rely on pre-defined rules and patterns to detect known threats and block them. However, they may not be effective in detecting zero-day attacks (new or unknown attacks). The proposed WAF uses Deep Learning techniques which involves training the system to detect and identify new and unknown patterns with higher accuracy than the traditional WAFs used before. Moreover, WAF proposed a layered architecture to enhance the accuracy of threat detection and improve the response time in detecting and preventing attacks in a short time.

1.2 Motivation

Nowadays server attacks have become popular more than ever, the WAF entered the market in the 1990s. Recently, the OWASP top 10 list has been spreading all over the internet such as SQL injection, XSS, improper server design and configurations, and Zero-day attacks. The WAF ensures and puts an end to all malicious web traffic and only legitimate traffic is passed to your web server according to the set of standards and configuration rules and protocols of the WAF.

1.2.1 Academic

As an academic organization whether you are a school or university security is considered the top and most important thing nowadays as everything around us from e-learning and home schools which are conducted online connected by schools' websites or online university courses on its website. Fortunately, the WAF provides protection for web applications of your school or university from a variety of application layer attacks such as cross-site scripting (XSS), SQL injection, cookie poisoning, network spoofing, and any kind of masquerade attack which threatens the website transactional processes.

1.2.2 Business

As a business founder or worker whether you have a small or big business you care so much about the security and confidentiality of your projects, websites, and anything it wants to share online in the internet for the whole world to see and interact with whether it is for marketing, e-commerce website, or applications. Luckily, the WAF helps block spyware, viruses, worms, phishing, cross-site scripting, SQL injections, zero-day, and other various other types of cyber-security attacks. The outcome is better website performance and increased security.

1.3 Problem Statement

Our main problem is to enhance the accuracy of the traditional WAF by training and testing datasets to detect known malware attacks and newly discovered attacks (zero-day attacks) and to provide quick response time to attack detection and prevention.

2 Project Description

The aim and goal of this project is to protect and safeguard the websites from any malicious attacks coming from outsiders or insiders whose aim is to break down the system and cause unavailability. The project's main target is to enhance the security of Web Application Firewalls (WAFs) against various types of attacks and malicious requests using Machine Learning (ML) and Deep Learning (DL) techniques by gathering diverse and representative datasets of web application traffic, including normal and malicious requests, to train and evaluate these requests which help in distinguishing between the normal and anomalous ones. It also provides countermeasures to zero-day attacks that keep them up to date. As a result, the attack detection overhead time is reduced, making websites faster and more secure.

2.1 Objectives

- Addresses the limitations of traditional WAFs in detecting known, new, and unknown attacks (the zero-day attacks).
- Proposes a layered architecture of WAF to improve the accuracy of threat detection.
- Proposes solutions to reduce the attack detection of overhead time using intelligent algorithms that can predict and prevent web attacks.
- Provides blocking or prevention from different classes of attack types patterns that consistently manage to bypass the WAFs.

2.2 Scope

Our main focus will be on the Web Applications.

1. It is easier for end users as they don't need to install an application (online usage).
2. Users have the opportunity to use the website from different browsers.
3. Web applications are efficient in development for businesses as they are relatively simple and cost-effective.
4. Web applications store the data on a server so there is no need to install them on a hard drive
5. It is beneficial for companies as there are no storage limitations online.

2.3 Project Overview

The WAF's main purpose is to monitor traffic using a filter/block policy which can detect safe requests and unsafe requests. The safe ones are approved and sent to the destination servers while the unsafe ones are blocked by WAF and have been ceased.

The data-set filtration's main purpose is to preprocess the data which is to simplify it and split it for training and testing. In the training and testing data phases, by using deep and machine learning techniques, then, begin evaluating the models that have been conducted from both training and testing to produce results of the data set.

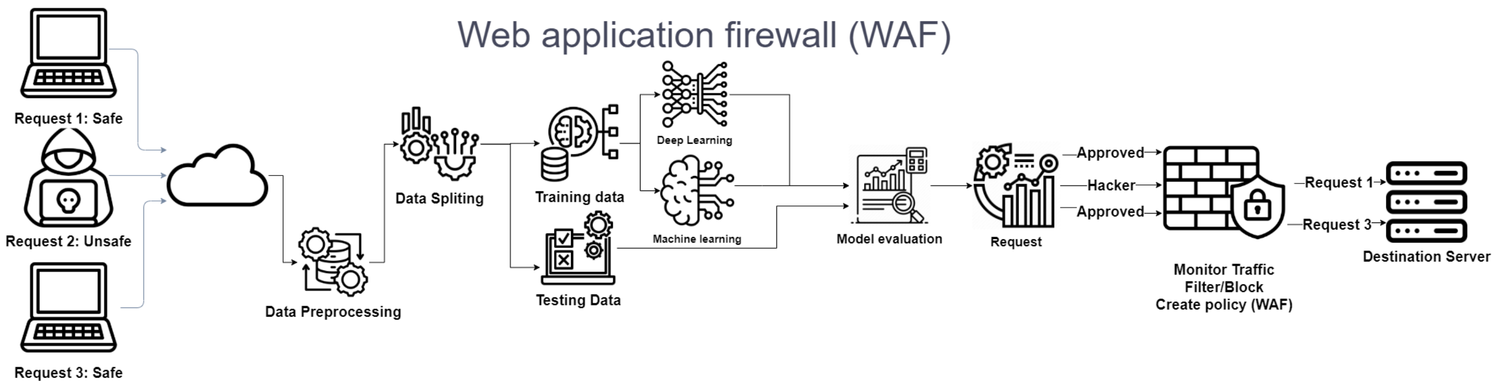


Figure 1: WAF system overview

2.4 Stakeholders

2.4.1 Internal

Team Leader: Monica Medhat and Team Members: Martina Milad, Samer Raafat, Kevin Emad.

2.4.2 External

External Stakeholders:

- Students who want to improve the security of the web applications they use related to universities or schools.
- Developers/Organizations who want to improve the security of the organization's used web applications.

3 Similar Systems

3.1 Academic

- **Montarulial et al.**[13]The ModSecurity WAF is mostly unsuccessful because it does not construct any model of the traffic observed by the protected web services, resulting in many false alarms, and it is readily defeated by adversarial SQLi attacks. Through adversarial training, the suggested AdvModSec methodology enhances robustness to adversarial SQLi assaults while improving the trade-off between detection and false positive rates. Based on empirical findings, AdvModSec enhances the adversarial robustness of vanilla ModSecurity by up to 42% and increases its detection rate by 21%. The approach offers a first tangible illustration of how adversarial machine learning may be applied to improve the resilience of WAFs against adversarial attacks. Along with promising future research possibilities focused at reinforcing classical rule-based solutions with machine learning-based approaches, and bridging the gap between these two worlds, the study also analyses related work and the limitations of the suggested approach. The research comes to the conclusion that web threats, such as DDoS, XSS, and SQL injection attacks, can be successfully identified and stopped by the suggested deep learning-based WAF. The suggested approach was able to identify novel and unidentified attacks and performed more accurately than conventional WAFs. Additionally, the study found a number of attack-indicating characteristics and factors that can be applied to raise threat detection accuracy.When compared to conventional WAFs, the suggested layered architectural paradigm increased threat detection accuracy. The paper suggests training the WAF and improving its performance with deep learning algorithms, such as CNNs and LSTMs. The paper's overall findings demonstrate how deep learning techniques can be used to strengthen web application security and fend against cyberattacks.

- **Dawadi et al.**[11] The suggested deep learning-based WAF is successful in identifying and stopping online threats, such as DDoS, XSS, and SQL injection assaults which are the most attacks that face the websites, according to the paper's conclusion. The suggested approach was able to identify novel and unidentified attacks and performed more accurately than conventional WAFs. In addition, the study found a number of attack-indicating characteristics and factors that can be applied to raise threat detection accuracy. When compared to conventional WAFs, the suggested layered architectural paradigm increased threat detection accuracy. The paper suggests training the WAF and improving its performance with deep learning algorithms, such as CNNs and LSTMs. The paper's overall findings demonstrate how deep learning techniques can be used to strengthen web application security and fend against cyberattacks.

- **Applebaum et al.**[3] The author mentions the limitations of signature-based web application firewalls and the potential of machine learning-based approaches to overcome these limitations. Signature-based web application firewalls use only a pre-defined set of rules and patterns in order to detect known threats. However, they may not be effective in detecting zero-day attacks (new or unknown attacks).In addition, this paper also provides a review of machine-learning-based WAF methods and identifies open issues. The authors conclude that machine-learning-based methods have advantages over signature/rule-based methods as they can address the vulnerability to zero-day attacks and can be easier to configure and keep up to date. However, the effectiveness of machine-learning-based WAFs in protecting current attack patterns targeting web application frameworks is still an open area for further investigation. The paper also lists applicable datasets, such as CSIC2010, for testing and training machine learning-based algorithms for web application firewalls.

- **Alghawazi et al.** [7] suggested deep learning architecture based on an RNN autoencoder is a viable solution for detecting SQL injection attacks on online applications, according to the article. The testing findings indicated that the suggested strategy outperformed other current approaches for identifying SQL injection attacks in terms of accuracy and F1-score. The authors also note that the dataset utilised in this study was quite limited, and they urge that future studies extend the dataset and use the models in real-world circumstances. The study gives a detailed assessment of the literature on ML and DL strategies for detecting SQL injection attacks and examines the possibility of future research employing more complicated architectures for the RNN autoencoder.

- **Seyyar et al.** [5] In the categorization of anomalous and normal requests, the suggested strategy in the study based on BERT and deep learning approaches produced a success rate of over 99.98% and an F1 score of over 98.70%. The BERT model has a strong representational capability of URLs, which greatly benefits high performance web attack detection. The authors employed an aggregate dataset created by combining three separate datasets in both the training and validation phases. As a result, their methodology is capable of successfully generalising data from numerous datasets.

- **Prandl et al.** [1] explains the various harmful malware types that web application developers and internet security experts should be aware of. Finding out how effective free WAFs are generally and whether or not they can be thought of as competitive alternatives to proprietary solutions are its main objectives. It also examines the harmful kinds of patterns that routinely get past the free WAFs and how successful they are at blocking them. Three methods are utilized to check: Guardian, Webknight, and ModSecurity. The study concluded that ModSecurity is the best technique for free WAFs since it is very good at thwarting frequently occurring attacks like XSS and SQL injection and can handle a sizable number of attacks while producing a negligible number of false alerts.

- **Gandotra et al.** [2] explains how to use iptables to protect against Application Layer and Web-based attacks without causing a prolonged system degradation that maintains the system's speed and network quality. Its main objective is to evaluate the effectiveness and speed of Iptables rules for Web-based attack detection and mitigation, as well as other application layer attacks. Techniques like SQL injection and cross-site scripting are used to test web-based attacks, whereas HTTP flooding, FTP flooding, and FTP bounce scanning are used to test application layer attacks. The study came to the conclusion that iptables can be utilized as an Application Layer Firewall with the right attack signatures since it is effective at identifying attacks.

3.2 Summary of Critical Analysis

Similar Systems Comparsion Table				
Paper ID	**Objective**	**Method**	**Results**	**Future Work and missing step**
Dawadi et al. 2023 [11]	Propose a tiered WAF architecture to increase threat detection accuracy.	Data collection, feature extraction, deep learning and model evaluation	DL-WAF outperforms TTS in accuracy and effectiveness in detecting web attacks.	the suggested model might be expanded to identify attacks in other forms of network traffic including email and instant messaging, but types of attacks may occur that the model does not recognize.
Montarulial et al. 2023 [13]	To address the issue of WAF's basic technique, this article developed AdvModSec, a robust machine learning model.	AT and ML algorithms improve the detection of (WAFs) against adversarial SQL injection.	AdvModSec improves the detection of the ModSecurity by 21%, and its adversarial 42%	Assessing the transferability of adversarial SQLi attacks optimised on ModSecurity using commercial solutions based on CRS, but it lacked some aspects that could be used.

Table 2: Similar Systems Comparsion Table

Similar Systems Comparsion Table				
Paper ID	**Objective**	**Method**	**Results**	**Future Work and missing step**
Applebaum et al. 2021 [3]	The authors want to see how well firewalls defend against other forms of attacks, such as SQL injection, cross-site scripting, and other OWASP Top Ten web application security threats.	Signature-based approaches, Machine Learning approaches	Machine Learning - based WAFs are used instead of signature-based rules achieving 98.8% accuracy in identifying new attacks.	More testing of open-source WAFs and algorithms based on ML.
Alghawazi et al. 2023 [7]	The purpose of this article is to assess the performance of the suggested RNN autoencoder-based architecture for identifying SQLIAs.	RNN autoencoder, ANN, CNN	CNN showed the highest accuracy 96% followed by RNN and ANN with accuracy 94% and F1-score 92% while CNN F1-score is 49% lower than RNN and ANN.	The use of a more complicated architecture for the RNN autoencoder to identify SQLIAs is being investigated, but the dataset used is somewhat small.

Similar Systems Comparsion Table				
Paper ID	**Objective**	**Method**	**Results**	**Future Work and missing step**
Seyyar et al. 2022 [5]	The goal of this study is to present a novel attack detection system based on BERT and deep learning to address the vulnerability of information in web applications targeted by attackers.	BERT (Bidirectional Encoder Representations from Transformers)	In the categorization of anomalous and regular requests, the success rate is above 99.98% and the F1 score is over 98.70%.	The proposed model only runs on the Linux platform, but the authors intend to modify it to run on Android, iOS, and Mac platforms in the future. However, the research does not investigate the suggested approach's potential constraints, such as the computing costs of training and deploying the model in real-time systems. .

Similar Systems Comparison Table				
Paper ID	**Objective**	**Method**	**Results**	**Future Work and missing step**
Prandl et al. 2015 [1]	the efficiency of the current freeware WAF solutions and whether or not they are a viable substitute for WAFs that are purchased commercially.	ModSecurity, WebKnight and Guardian	ModSecurity prevented 52% of forged packets, while Guardian only stopped 5%.Both WebKnight and ModSecurity blocked 99.93% of malicious attacks.	The paper didn't include many tests and results with different methods and there is only one source of framework was used for the testing.
Gandotra et al. 2020 [2]	Understanding the keywords and the techniques for creating the attacks	Methods used in the experiment for web-based attacks: SQL Injection (SQLi) and Cross-Site Scripting (XSS) Methods used in the experiment for Application Layer Attacks: HTTP Flooding, FTP Flood and FTP Bounce Scan	Iptables, an Application Layer-Firewall, demonstrated a 1% performance degradation in web server protection against Web attacks.Its highest CPU utilization was 8.35 for FTP Flood attacks, indicating its excellent performance in detecting application layer attacks compared to Intrusion Detection Systems.	The suggested technique will be evaluated for false alarms, however there are only a few instances available for testing.

3.3 Business Applications

1. **Imperva WAF:** [6] Web application firewall (WAF) hosted in the cloud, Imperva offers DDoS protection, load balancing, failover, and content delivery network (CDN) services. Additionally, it offers defense against the OWASP Top Ten attack types. It functions by altering the protected application's DNS. Additionally, it has a number of inbuilt WAF rules that may be customized using an editor to defend against various kinds of attacks. Additionally, Imperva offers two-factor authentication that works with any website without requiring modifications to the application code.

2. **AWS WAF:** [4] It stands for Amazon Web Services, a cloud computing platform that Amazon offers. It provides a wide range of services, one of which is a WAF service that aids in protecting web applications from frequent attacks that can endanger their security or impact their availability. Control over access to online apps is possible thanks to this managed service. Additionally, it can be used to build unique rules for certain applications that prevent common pattern attacks like SQL injection and cross-site scripting (XSS). Additionally, AWS WAF interfaces with AWS Shield and Amazon CloudFront to offer a comprehensive security solution for web applications.

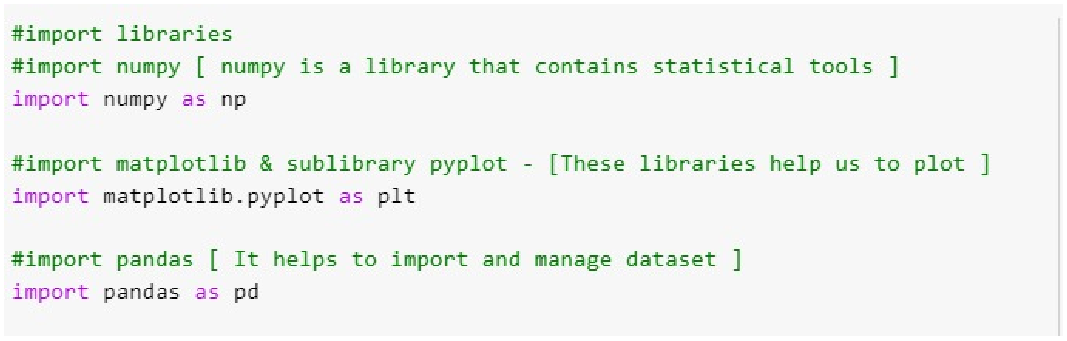
3. **Azure WAF:** [9] it is a security feature provided by Microsoft Azure that helps protect web applications from common web-based attacks such as SQL injection, cross-site scripting (XSS), and cross-site forgery. Located between the web application and the internet, this layer 7 firewall examines incoming traffic and prevents any fraudulent requests. Custom rules and policies can be set up in Azure WAF to satisfy certain security needs.

Figure 2: Business Applications

4 What is new in the proposed project?

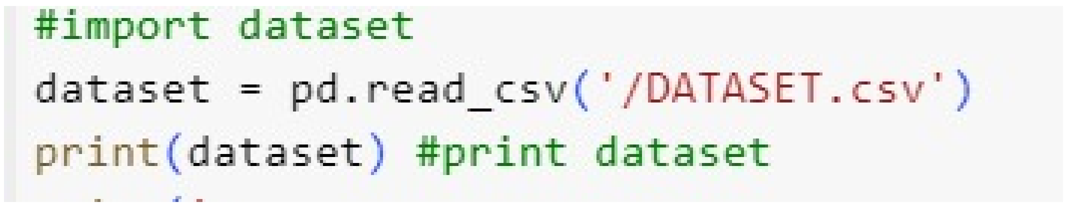
Based on the read research papers; most of the papers were comparing the ML algorithms with other ML algorithms or DL algorithms with other DL learning algorithms. Our target and added value in this project is to enhance the performance and the run time of the proposed WAF. Not only by comparing similar algorithms with each other; but also by comparing different algorithms with each other (comparing machine learning algorithms with deep learning algorithms) in order to achieve higher accuracy.

5 Proof of Concept



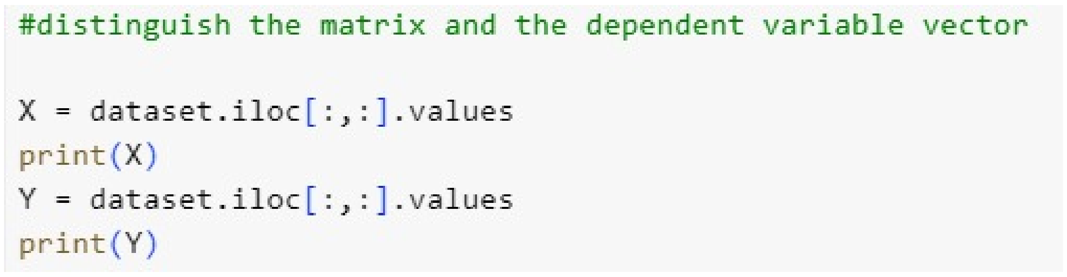
```
#import libraries
#import numpy [ numpy is a library that contains statistical tools ]
import numpy as np

#import matplotlib & sublibrary pyplot - [These libraries help us to plot ]
import matplotlib.pyplot as plt

#import pandas [ It helps to import and manage dataset ]
import pandas as pd
```

Figure 3: Libraries imports



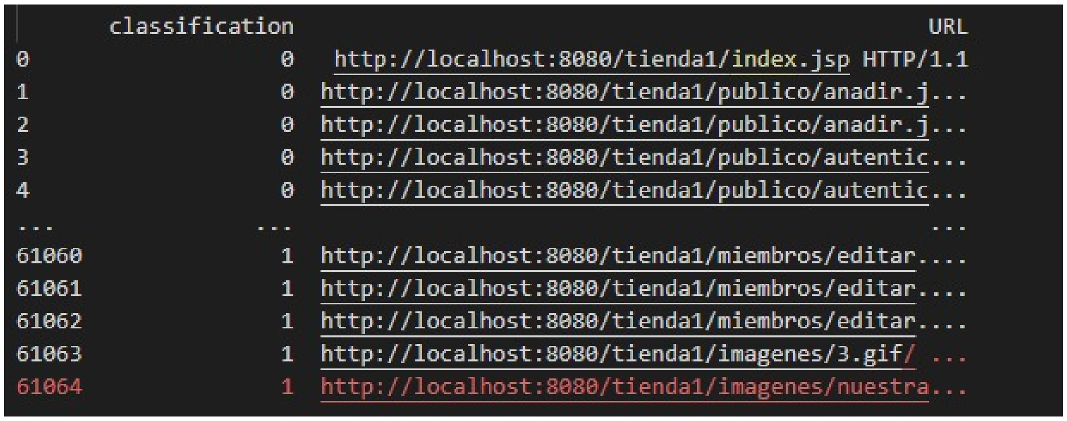
```
#import dataset
dataset = pd.read_csv('/DATASET.csv')
print(dataset) #print dataset
```

Figure 4: Dataset imports



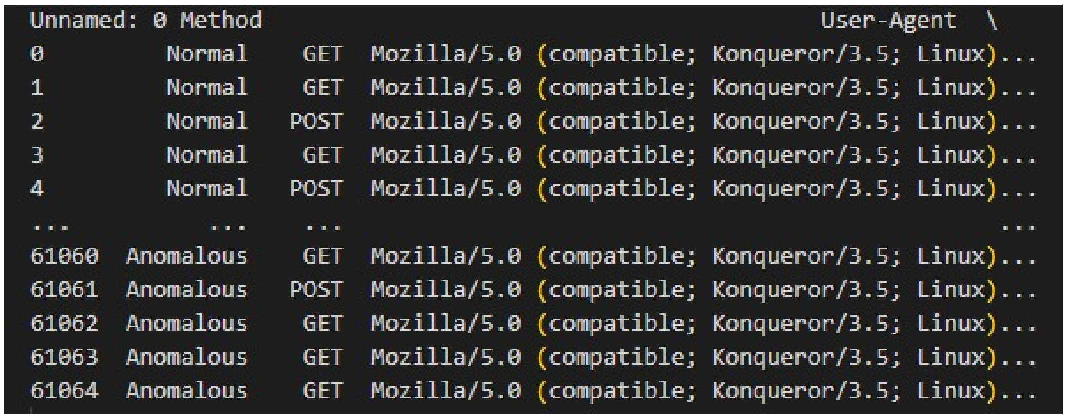
```
#distinguish the matrix and the dependent variable vector

X = dataset.iloc[:,:].values
print(X)
Y = dataset.iloc[:,:].values
print(Y)
```

Figure 5: Dataset X-Y axes



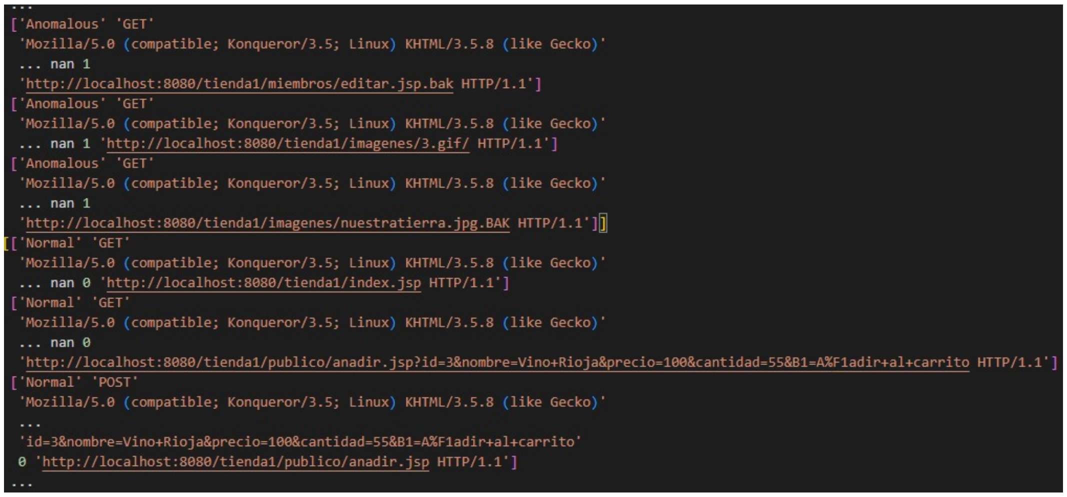
```
       classification                                              URL
0                   0   http://localhost:8080/tienda1/index.jsp HTTP/1.1
1                   0  http://localhost:8080/tienda1/publico/anadir.j...
2                   0  http://localhost:8080/tienda1/publico/anadir.j...
3                   0  http://localhost:8080/tienda1/publico/autentic...
4                   0  http://localhost:8080/tienda1/publico/autentic...
...               ...                                                ...
61060               1  http://localhost:8080/tienda1/miembros/editar....
61061               1  http://localhost:8080/tienda1/miembros/editar....
61062               1  http://localhost:8080/tienda1/miembros/editar....
61063               1  http://localhost:8080/tienda1/imagenes/3.gif/ ...
61064               1  http://localhost:8080/tienda1/imagenes/nuestra...
```

Figure 6: Code 1



```
 Unnamed: 0 Method                                          User-Agent  \
 0          Normal    GET  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 1          Normal    GET  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 2          Normal   POST  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 3          Normal    GET  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 4          Normal   POST  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 ...           ...    ...                                                ...
 61060   Anomalous    GET  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 61061   Anomalous   POST  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 61062   Anomalous    GET  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 61063   Anomalous    GET  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
 61064   Anomalous    GET  Mozilla/5.0 (compatible; Konqueror/3.5; Linux)...
```

Figure 7: Code 2



```
...
['Anomalous' 'GET'
 'Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)'
 ... nan 1
 'http://localhost:8080/tienda1/miembros/editar.jsp.bak HTTP/1.1']
['Anomalous' 'GET'
 'Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)'
 ... nan 1 'http://localhost:8080/tienda1/imagenes/3.gif/ HTTP/1.1']
['Anomalous' 'GET'
 'Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)'
 ... nan 1
 'http://localhost:8080/tienda1/imagenes/nuestratierra.jpg.BAK HTTP/1.1']]
[['Normal' 'GET'
 'Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)'
 ... nan 0 'http://localhost:8080/tienda1/index.jsp HTTP/1.1']
['Normal' 'GET'
 'Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)'
 ... nan 0
 'http://localhost:8080/tienda1/publico/anadir.jsp?id=3&nombre=Vino+Rioja&precio=100&cantidad=55&B1=A%F1adir+al+carrito HTTP/1.1']
['Normal' 'POST'
 'Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)'
 ...
 'id=3&nombre=Vino+Rioja&precio=100&cantidad=55&B1=A%F1adir+al+carrito'
 0 'http://localhost:8080/tienda1/publico/anadir.jsp HTTP/1.1']
...
```

Figure 8: Code Results

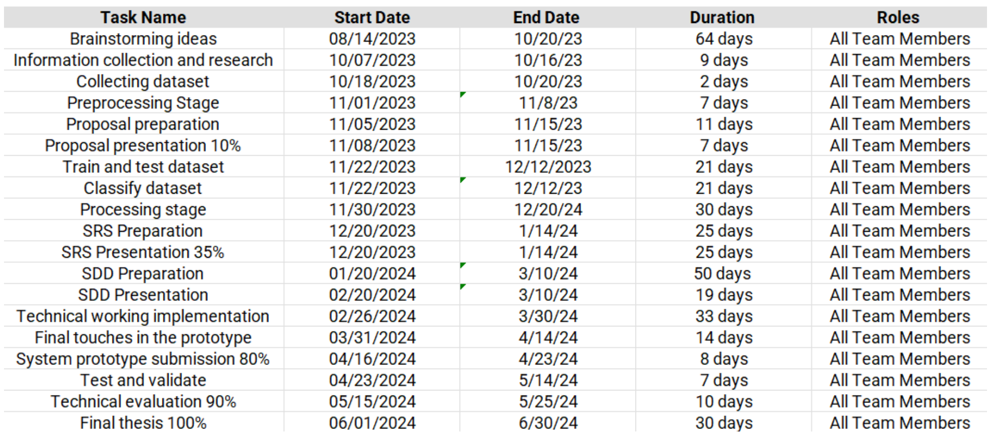
6 Project Management and Deliverables

6.1 Deliverables

This project's main purpose is to monitor traffic using a filter/block policy that can detect safe requests and unsafe requests. The safe ones are approved and sent to the destination servers while the unsafe ones are blocked by WAF and have been ceased.

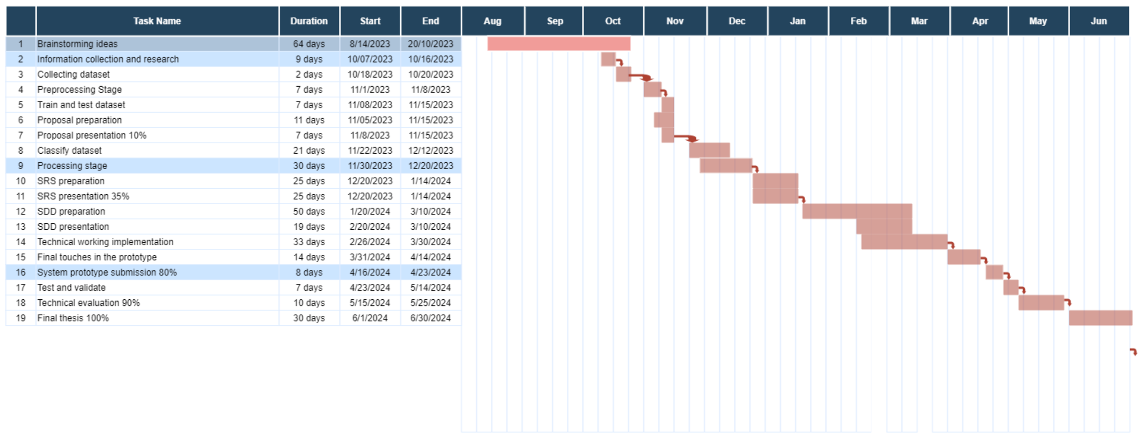
- The project will be submitted in 6 phases
  1. **Software proposal document.**
  2. **Software Requirements Specification:** the software nature is described.
  3. **Software Design Description:** how the system design is represented.
  4. **Thesis Document:** shows all the researches done to deploy the application.
  5. **Research papers:** multiple papers were read concerning WAF and submitted to the conferences.
  6. **The Web application.**

6.2 Time and Task plan



Task Name	Start Date	End Date	Duration	Roles
Brainstorming ideas	08/14/2023	10/20/23	64 days	All Team Members
Information collection and research	10/07/2023	10/16/23	9 days	All Team Members
Collecting dataset	10/18/2023	10/20/23	2 days	All Team Members
Preprocessing Stage	11/01/2023	11/8/23	7 days	All Team Members
Proposal preparation	11/05/2023	11/15/23	11 days	All Team Members
Proposal presentation 10%	11/08/2023	11/15/23	7 days	All Team Members
Train and test dataset	11/22/2023	12/12/2023	21 days	All Team Members
Classify dataset	11/22/2023	12/12/23	21 days	All Team Members
Processing stage	11/30/2023	12/20/24	30 days	All Team Members
SRS Preparation	12/20/2023	1/14/24	25 days	All Team Members
SRS Presentation 35%	12/20/2023	1/14/24	25 days	All Team Members
SDD Preparation	01/20/2024	3/10/24	50 days	All Team Members
SDD Presentation	02/20/2024	3/10/24	19 days	All Team Members
Technical working implementation	02/26/2024	3/30/24	33 days	All Team Members
Final touches in the prototype	03/31/2024	4/14/24	14 days	All Team Members
System prototype submission 80%	04/16/2024	4/23/24	8 days	All Team Members
Test and validate	04/23/2024	5/14/24	7 days	All Team Members
Technical evaluation 90%	05/15/2024	5/25/24	10 days	All Team Members
Final thesis 100%	06/01/2024	6/30/24	30 days	All Team Members

Figure 9: Time Plan Of The Proposed System



	Task Name	Duration	Start	End
1	Brainstorming ideas	64 days	8/14/2023	20/10/2023
2	Information collection and research	9 days	10/07/2023	10/16/2023
3	Collecting dataset	2 days	10/18/2023	10/20/2023
4	Preprocessing Stage	7 days	11/1/2023	11/8/2023
5	Train and test dataset	7 days	11/08/2023	11/15/2023
6	Proposal preparation	11 days	11/05/2023	11/15/2023
7	Proposal presentation 10%	7 days	11/8/2023	11/15/2023
8	Classify dataset	21 days	11/22/2023	12/12/2023
9	Processing stage	30 days	11/30/2023	12/20/2023
10	SRS preparation	25 days	12/20/2023	1/14/2024
11	SRS presentation 35%	25 days	12/20/2023	1/14/2024
12	SDD preparation	50 days	1/20/2024	3/10/2024
13	SDD presentation	19 days	2/20/2024	3/10/2024
14	Technical working implementation	33 days	2/26/2024	3/30/2024
15	Final touches in the prototype	14 days	3/31/2024	4/14/2024
16	System prototype submission 80%	8 days	4/16/2024	4/23/2024
17	Test and validate	7 days	4/23/2024	5/14/2024
18	Technical evaluation 90%	10 days	5/15/2024	5/25/2024
19	Final thesis 100%	30 days	6/1/2024	6/30/2024

Figure 10: Time Plan Chart

7 Supportive Documents

7.1 Dataset Description

The project's primary dataset is the same one utilized for all of our research publications. **The utilized dataset used is: HTTP dataset CSIC 2010** Dataset [11] [5] [3], This dataset was produced by the Spanish Research National Council's "Information Security Institute" (CSIC). Over 25,000 fraudulent or hazardous requests and 36,000 normal or safe requests make up the automatically generated dataset. The dataset contains numerous attacks, including server-side include, parameter manipulation, CRLF injection, SQL injection, buffer overflow, information collecting, file disclosure, and XSS. Three types of harmful or unexpected requests were taken into consideration in this dataset:

1. **Static attacks** [Such attacks attempt to obtain resources that are hidden or nonexistent, such as configuration files, default files, session ID in URL rewrites, outdated files, etc.]

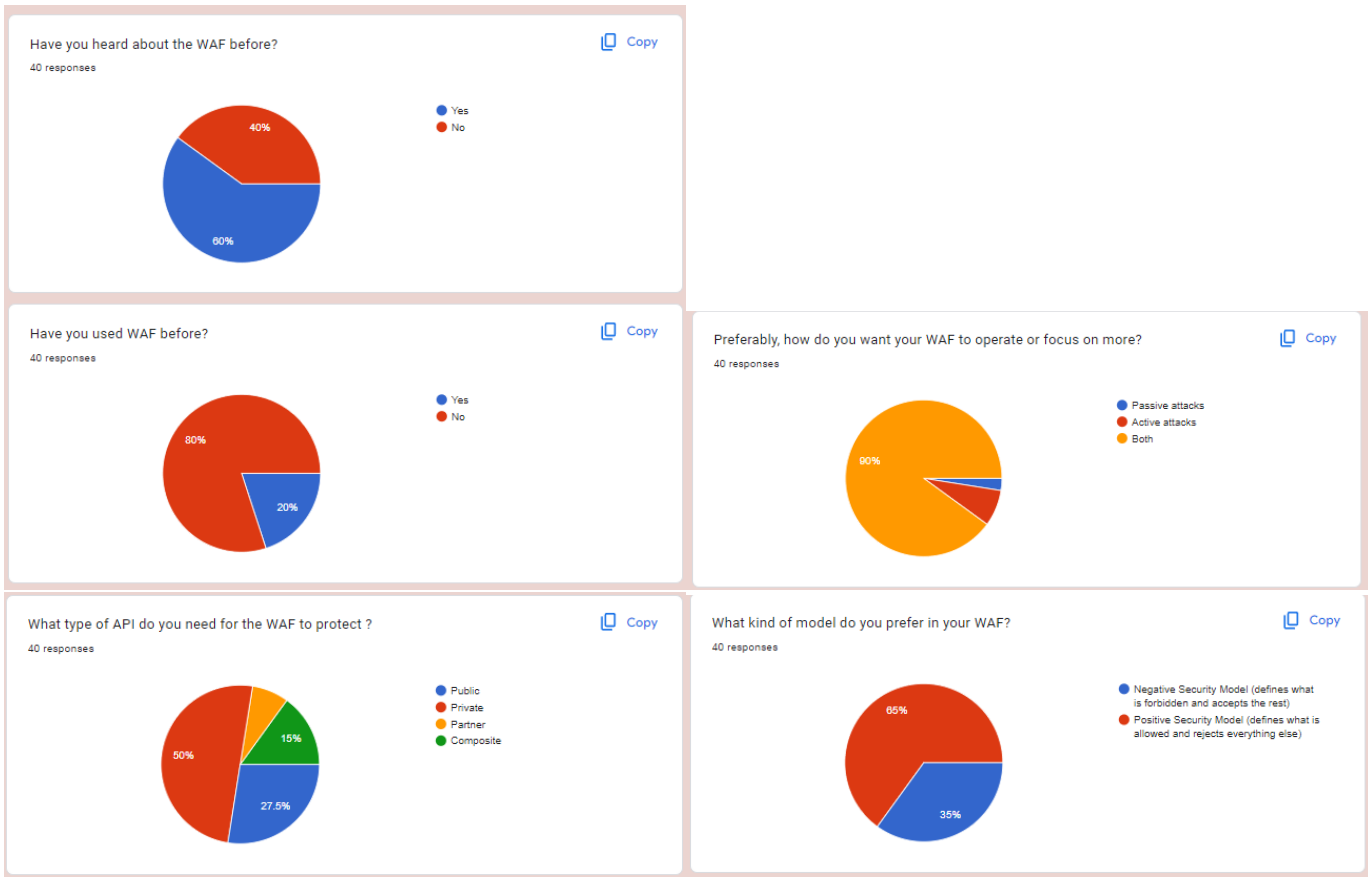
2. **Dynamic attacks** [These types of attacks alter legitimate request parameters, such as buffer overflows, cross-site scripting, SQL injection, and CRLF injection.]

3. **Unintentional illegal requests** [These requests are not harmful, but they lack the web service's usual protocol and fail to have the same format as regular parameter values (a phone number composed of letters, for instance)].

Tools like Paros and W3AF were used to build each attack in this dataset. The dataset is finally divided into three components. For the training phase, there is a single subset that only includes regular or normal traffic. as well as two test subgroups: one for legitimate traffic and the other for spam traffic.

More datasets will be used later on if needed and in case this dataset is insufficient. And regarding the newly discovered attacks (zero-day attacks) have no dataset, and patterns will be used to discover them.

7.2 Survey


Have you heard about the WAF before?

40 responses

Have you used WAF before?

40 responses

Preferably, how do you want your WAF to operate or focus on more?

40 responses

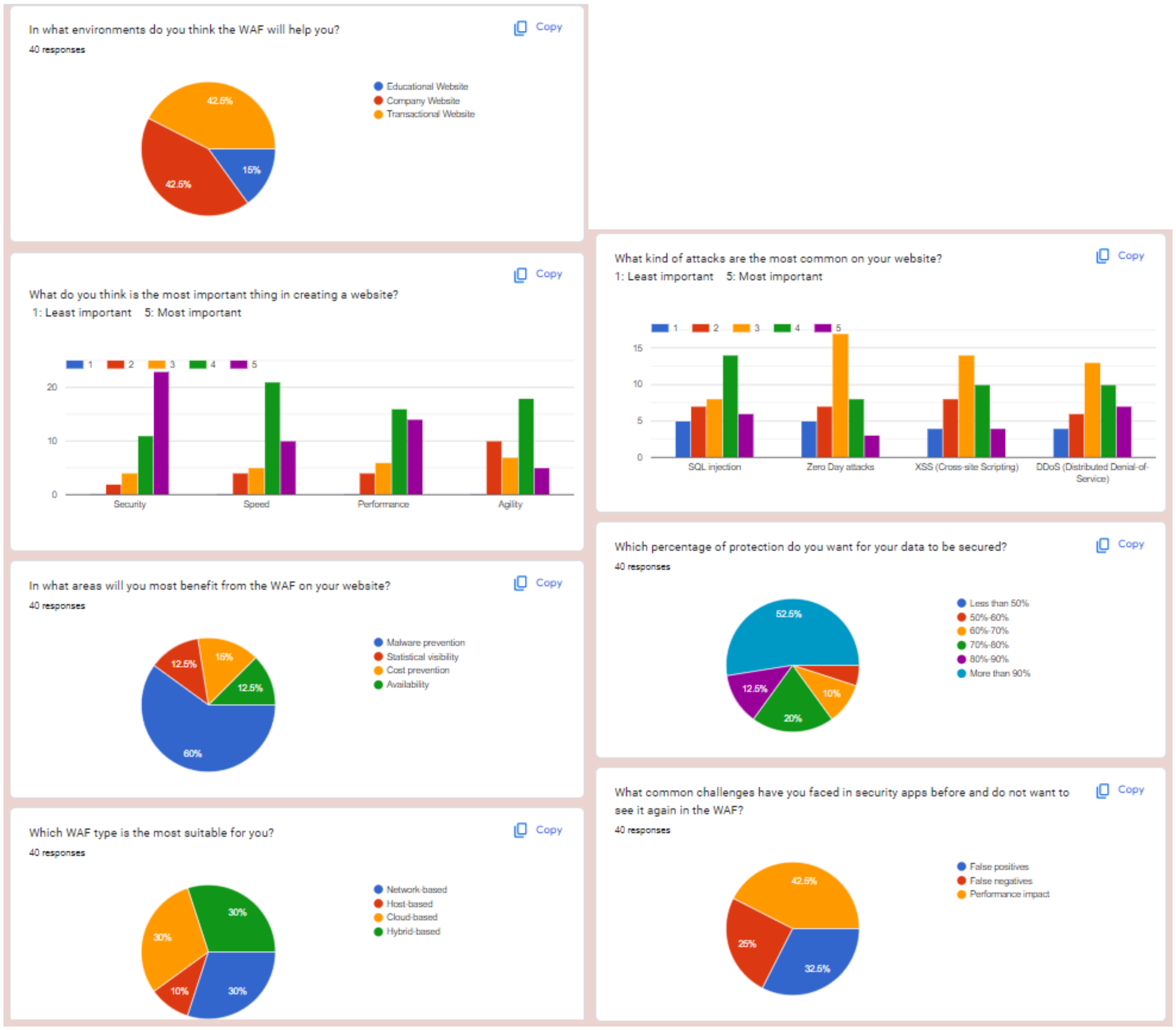
What type of API do you need for the WAF to protect ?

40 responses

What kind of model do you prefer in your WAF?

40 responses

Figure 11: Survey


In what environments do you think the WAF will help you?
40 responses

What do you think is the most important thing in creating a website?
1: Least important 5: Most important

In what areas will you most benefit from the WAF on your website?
40 responses

Which WAF type is the most suitable for you?
40 responses

What kind of attacks are the most common on your website?
1: Least important 5: Most important

Which percentage of protection do you want for your data to be secured?
40 responses

What common challenges have you faced in security apps before and do not want to see it again in the WAF?
40 responses

Figure 12: Survey (Rest of the questions)

References

[1] Stefan Prandl, Mihai Lazarescu, and Duc-Son Pham. "A study of web application firewall solutions". In: *Information Systems Security: 11th International Conference, ICISS 2015, Kolkata, India, December 16-20, 2015. Proceedings 11*. Springer. 2015, pp. 501–510.

[2] Nikita Gandotra and Lalit Sen Sharma. "Exploring the use of Iptables as an Application Layer Firewall". In: *Journal of The Institution of Engineers (India): Series B* 101 (2020), pp. 707–715.

[3] Simon Applebaum, Tarek Gaber, and Ali Ahmed. "Signature-based and machine-learning-based web application firewalls: A short survey". In: *Procedia Computer Science* 189 (2021), pp. 359–367.

[4] Se-Joon Park, Yong-Joon Lee, and Won-Hyung Park. "Configuration Method of AWS Security Architecture That Is Applicable to the Cloud Lifecycle for Sustainable Social Network". In: *Security and Communication Networks* 2022 (2022), pp. 1–12.

[5] Yunus Emre Seyyar, Ali Gökhan Yavuz, and Halil Murat Ünver. "An attack detection framework based on BERT and deep learning". In: *IEEE Access* 10 (2022), pp. 68633–68644.

[6] Tomás Sureda Riera et al. "Systematic Approach for Web Protection Runtime Tools' Effectiveness Analysis". In: (2022).

[7] Maha Alghawazi, Daniyal Alghazzawi, and Suaad Alarifi. "Deep Learning Architecture for Detecting SQL Injection Attacks Based on RNN Autoencoder Model". In: *Mathematics* 11.15 (2023), p. 3286.

[8] Fahad M Alotaibi and Vassilios G Vassilakis. "Toward an SDN-Based Web Application Firewall: Defending against SQL Injection Attacks". In: *Future Internet* 15.5 (2023), p. 170.

[9] Stefan Boneder. "Evaluation and comparison of the security offerings of the big three cloud service providers Amazon Web Services, Microsoft Azure and Google Cloud Platform". PhD thesis. Technische Hochschule Ingolstadt, 2023.

[10] Oumaima Chakir et al. "An empirical assessment of ensemble methods and traditional machine learning techniques for web-based attack detection in industry 5.0". In: *Journal of King Saud University-Computer and Information Sciences* 35.3 (2023), pp. 103–119.

[11] Babu R Dawadi, Bibek Adhikari, and Devesh Kumar Srivastava. "Deep Learning Technique-Enabled Web Application Firewall for the Detection of Web Attacks". In: *Sensors* 23.4 (2023), p. 2073.

[12] Yuting Guan et al. "SSQLi: A Black-Box Adversarial Attack Method for SQL Injection Based on Reinforcement Learning". In: *Future Internet* 15.4 (2023), p. 133.

[13] Biagio Montaruli et al. "Adversarial ModSecurity: Countering Adversarial SQL Injections with Robust Machine Learning". In: *arXiv preprint arXiv:2308.04964* (2023).