Neural Network-Based Wed Security Firewall

Prof. Sonali Benke[1], Khule Abhijit[2], Pawase Sanket[3], More Vaibhav[4], Thorat Siddhant[5]

[1]Assistant Professor, Department of Computer Engineering, Sir Visvesvaraya Institute of Technology, Nashik, Maharashtra, India.

[2,3,4,5]Department of Computer Engineering, Sir Visvesvaraya Institute of Technology, Nashik, Maharashtra, India.

---***---

Abstract
In today's digital environment, securing web applications is paramount. As cyberattacks become more sophisticated, traditional defense mechanisms struggle to keep pace. This project proposes a novel Web Application Firewall (WAF) empowered by Neural Network (NN) technology.

Our objective is to significantly bolster web application security by proactively detecting and blocking malicious attacks. The NN-based WAF leverages deep learning to analyze incoming HTTP requests in real-time, effectively differentiating between legitimate traffic and potentially harmful attempts. Through meticulous data gathering, feature engineering, and rigorous model training, the WAF achieves robust and adaptable behavior.

This project signifies a significant leap forward in web application security. By capitalizing on the power of neural networks, it delivers real-time protection against a vast array of cyber threats. The NN-based WAF's ability to adapt to the ever-changing threat landscape makes it an invaluable tool for safeguarding web applications in the face of evolving and persistent security challenges.
.

Keywords: Web Application Firewall, Neural Network, Cybersecurity, Real-time Protection, The NN-based WAF.

1.INTRODUCTION

Web applications in today's digital world face a constant barrage of cyber threats. Distributed Denial-of-Service (DDoS) attacks, for instance, can cripple websites by overwhelming them with a flood of traffic, rendering them inaccessible to legitimate users. These attacks disrupt essential web services and can cause significant financial losses and downtime for organizations.

OS command injection exploits vulnerabilities in web applications to execute unauthorized system commands, potentially compromising the server's security and allowing attackers to gain control over the system.

SQL injection attacks manipulate SQL queries sent to a web application's database, enabling attackers to extract, modify, or delete sensitive data and execute unauthorized actions, posing a significant threat to data integrity and application security.

XSS (Cross-Site Scripting) attacks inject malicious scripts into web pages viewed by other users, allowing attackers to steal information, hijack sessions, or deface websites, posing a serious threat to user security and data integrity.

File Inclusion attacks exploit vulnerabilities in web applications to include remote files, potentially granting attackers unauthorized access to sensitive data or allowing them to execute malicious code on the server.
Current Web Application Firewalls (WAFs) often struggle to effectively identify and mitigate these threats, especially zero-day attacks. This leaves web applications vulnerable to data breaches and service disruptions, posing risks to both organizations and users. To address this pressing need, a more robust and adaptive WAF solution is crucial.

This project tackles this challenge by developing a next-generation Neural Network (NN)-based WAF. Our objective is to significantly bolster web application security, minimize vulnerabilities, and ensure uninterrupted service availability. This will ultimately safeguard user data and organizational assets.

Why We're Doing This?

We're undertaking this project to enhance web security by developing robust defenses against various cyber threats, safeguarding sensitive data, maintaining service availability, and protecting user privacy. By leveraging advanced technologies and methodologies, we aim to mitigate risks associated with attacks such as SQL injection, XSS, DDoS, OS command injection, and file inclusion, contributing to a safer online environment for individuals and organizations worldwide.

How It Works?

Our project utilizes Deep learning algorithms to analyze web traffic patterns and detect anomalies indicative of cyber-attacks. By training models on diverse datasets containing normal and malicious traffic, we enable our system to differentiate between legitimate requests and potential threats in real-time. This proactive approach enhances web security by identifying and mitigating attacks such as SQL injection, XSS, DDoS, OS command injection, and file inclusion, thereby safeguarding web applications and their users from harm.

2. LITERATURE SURVEY

WAFs are crucial security tools that safeguard web applications against various attacks like SQL injection, cross-site scripting (XSS), DDoS, OS command injection, and file inclusion. They rely on rule sets to analyze incoming traffic and block malicious requests. Traditional WAFs depend on signature-based detection, which struggles to identify new and unknown attacks. However, machine learning algorithms, such as neural networks, offer a promising approach to enhance WAF accuracy in detecting these novel threats [2].

Research has explored incorporating machine learning and feature engineering into WAFs for better attack detection. One study proposed a model that analyzes incoming requests, extracts specific features describing the HTTP request (URL, payload, and headers), and classifies them as normal or anomalous. This model achieved impressive classification accuracy on research datasets (99.6%) and real-world web server data (98.8%) [4].

Another study investigated using convolutional neural networks (CNNs) in a "smart WAF" design. The CNN effectively classified HTTP messages as normal or malicious requests, demonstrating a high attack detection rate [3].

In conclusion, Machine learning, particularly neural networks, holds significant potential for improving WAF detection capabilities, especially for novel and unknown attacks. The models presented in these studies offer a promising foundation for further research and development in this area.

3.METHODOLOGY

In this paper, building a model for attack detection involves several steps, including data collection, preprocessing, feature engineering, model selection, training, and evaluation. Here's a methodology tailored to specific requirements for detecting SQL injection, XSS (Cross-Site Scripting), OS command injection, DDoS (Distributed Denial of Service) attacks, and file inclusion attacks:

I. Data Collection:

We gather a diverse dataset containing examples of normal web traffic and instances of each type of attack (SQL injection, XSS, OS command injection, DDoS, file inclusion). We used five datasets to develop our neural network-based model for different attacks on web applications. For SQL injections and XSS attacks we collected datasetes from kaggle.com a community for users to explore, analyze, and share datasets.

Foe DDoS attack collecte dataset from "DDoS Evaluation Dataset (CIC-DDoS2019)." DDoS 2019 | Datasets | Research | Canadian Insitute for Cybersecurity. Over 200k+ columns for using neural network to differentiate between DDoS and benign data.

For OS Command injection and File inclusion attack, we collected various attack scrips of these two attacks from online resources such as Github, OWSAP and other web site. Adding some normal HTTP traffic request dataset make dataset of OS Command injection and File inclusion attack dataset.

II. Data Preprocessing:

In preparing HTTP traffic data for analysis, the preprocessing stage involves several key steps to ensure the dataset's quality and suitability for model training.

Initially, noisy data, such as corrupted packets or incomplete requests, is removed to maintain data integrity. Irrelevant features, such as timestamps or source IP addresses unrelated to the classification task, are filtered out to focus on relevant information. Categorical data, like HTTP methods or status codes, are transformed into numerical representations for analysis purposes. Numerical features, such as request sizes or response times, are normalized to a standard scale to aid model convergence.

The dataset is then divided into training, validation, and test sets, ensuring stratified sampling to maintain class distribution proportions across all sets and prevent bias. Randomization is applied within each set to minimize ordering effects and ensure model generalization. Finally, techniques like tokenization may be employed to process textual data within HTTP requests or responses for further feature extraction and analysis. This comprehensive preprocessing pipeline ensures that the dataset is clean, balanced, and appropriately split for training and evaluating models to detect normal and malicious HTTP traffic effectively.

III. Feature Engineering:
In the feature engineering stage, relevant characteristics indicative of each attack type is extracted from the dataset.

These features encompass various aspects of HTTP traffic, including request parameters, headers, URL structure, payload content, and request frequency. Techniques such as tokenization, encoding, and feature scaling are then applied to prepare the data for model training. Tokenization involves breaking down textual data, such as HTTP request parameters or payload content, into individual tokens or words for further analysis. Encoding transforms categorical variables, such as HTTP methods or status codes, into numerical representations suitable for machine learning algorithms. Feature scaling ensures that numerical features are standardized to a consistent scale, preventing any feature from datasets.

3. MODELING AND ANALYSIS

I. Model Selection:

In the case of web application attacks like SQL injection, cross-site scripting (XSS), OS command injection, and file inclusion attacks, analyzing the structure and content of HTTP requests is crucial. Convolutional Neural Networks (CNNs) are adept at processing grid-like data, such as images or sequential data with spatial relationships. By utilizing CNNs, features can be extracted from structured components of HTTP requests, such as URLs, headers, and parameters. CNNs are capable of learning hierarchical representations, facilitating the detection of complex attack patterns within HTTP traffic.

For detecting DDoS attacks, which typically involve high-volume traffic from multiple sources, traditional methods may fall short. Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN), excel at capturing sequential patterns and detecting anomalies in time series data. Leveraging LSTM networks allows for the analysis of temporal traffic patterns, enabling the identification of sudden spikes or irregularities indicative of DDoS attacks.

II. Model Training:
In the model training phase, the selected model(s) are trained using the prepared dataset, specifically the training set. Hyperparameters are fine-tuned through techniques like grid search or randomized search to optimize the model's performance, ensuring it achieves the best possible results on unseen data.
These techniques help prevent the model from being biased towards the majority class and ensure it can effectively learn from the minority class examples as well. By addressing class imbalance, the model becomes more adept at accurately detecting both normal and malicious instances, thereby enhancing its overall performance and reliability in real-world scenarios.

III. Model Evaluation:
In the model evaluation phase, the trained model(s) are assessed using the validation set to gauge their generalization performance on unseen data. Various performance metrics are calculated, including accuracy, precision, recall and F1-score. These metrics provide insights into different aspects of the model's performance, such as its ability to correctly classify instances, balance between true positives and false positives, and overall discriminative power.

Following are table of Models Classification Report:

Model	Accuracy		Precision	Recall	F1score
DDoS	0.99	0	1.00	1.00	1.00
		1	1.00	1.00	1.00
SQL injection	0.99	0	1.00	1.00	1.00
		1	0.98	1.00	1.00
XXS	0.99	0	1.00	1.00	1.00
		1	1.00	1.00	1.00
File inclusion	0.98	0	0.97	0.99	0.98
		1	1.00	0.99	0.99
OS Command	0.98	0	0.98	1.00	0.99
		1	0.99	0.95	0.97

Table 3.1 Model Classification Report

IV. Model Workflow Analysis:
It is a flowchart of a Web Application Firewall (WAF) working. A WAF is a type of firewall that specifically protects web applications from attacks.

Here is a detailed explanation of the flowchart of how a WAF works:

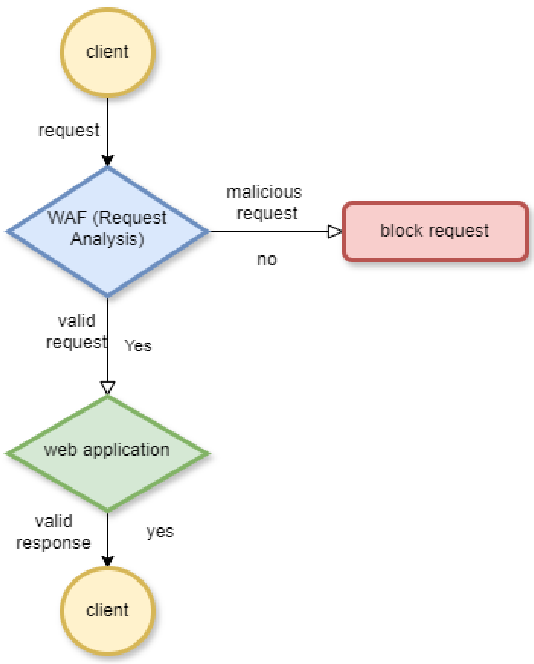
1. **Client Sends Request:** The process begins with a client, such as a web browser, sending a request to a web application.

2. **WAF Analyzes Request:** The request is then routed through the WAF. The WAF analyzes the request to determine if it is malicious. In WAF contain trained model of attacks are SQL injection, XXS, OS Command injection, File inclusion and DDoS attacks.

3. **Valid Request:** If the WAF determines that the request is valid, it allows the request to pass through to the web application.

4. **Malicious Request:** If the WAF determines that the request is malicious, it blocks the request

and prevents it from reaching the web application.



3.2 Flow chart

5. **Web Application Response:** If the request is valid, the web application processes the request and sends a response back to the client.

4.RESULTS AND DISCUSSION

The project identified various security threats including XSS, SQL injection, OS command injection, and file inclusion. A Web Application Firewall (WAF) was developed using Python, coupled with a MongoDB database for real-time attack information storage. Feature engineering was conducted, and neural network-based models were employed instead of traditional rule-based systems, enhancing adaptability to evolving threats. Model training utilized Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) techniques. The WAF effectively enhanced web application security by proactively identifying and thwarting malicious attacks, supported by a dashboard for visualizing attack reports. Continuous model training ensured adaptability to evolving threat landscapes, providing real-time protection against a wide spectrum of cyber threats. Additionally, the project implemented a threshold of 0.9 and achieved a train model accuracy of 99%. Performance assessment was conducted using precision, recall, F1-score, and confusion matrices to evaluate effectiveness. Furthermore, the system was tested on real-time attacks to validate its efficacy in identifying and mitigating security threats in practical scenarios.

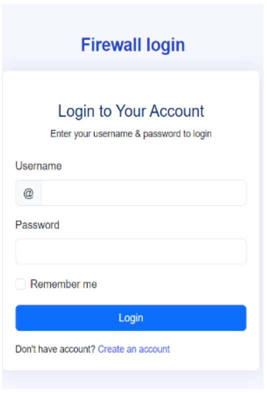
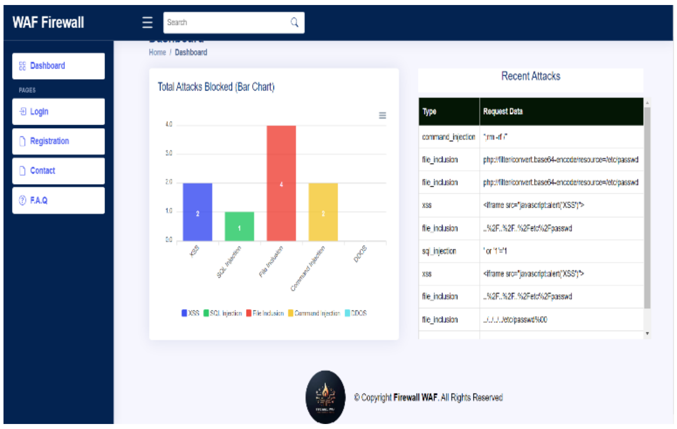

Firewall login
Login to Your Account
Enter your username & password to login
Username
@
Password
Remember me
Login
Don't have account? Create an account

Fig.4.1 Login page



Recent Attacks

Type	Request Data
command_injection	";rm -rf /"
file_inclusion	php://filter/convert.base64-encode/resource=/etc/passwd
file_inclusion	php://filter/convert.base64-encode/resource=/etc/passwd
xss	<iframe src="javascript:alert('XSS')">
file_inclusion	..%2F..%2F..%2Fetc%2Fpasswd
sql_injection	' or '1'='1
xss	<iframe src="javascript:alert('XSS')">
file_inclusion	..%2F..%2F..%2Fetc%2Fpasswd
file_inclusion	../../../../etc/passwd%00

Fig.4.2 Admin Dashboard

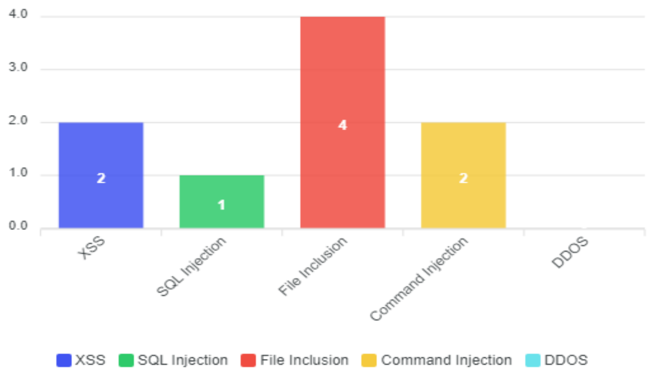
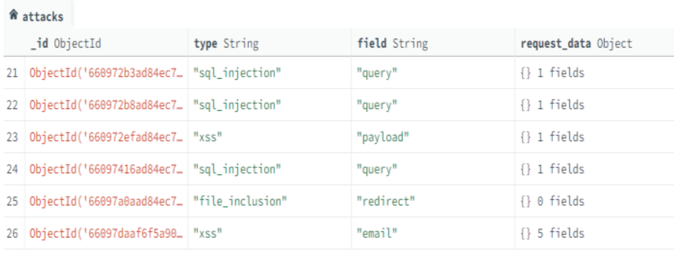


Fig.4.3 Bar graph showing attack report



attacks

	_id ObjectId	type String	field String	request_data Object
21	ObjectId('660972b3ad84ec7…	"sql_injection"	"query"	{} 1 fields
22	ObjectId('660972b8ad84ec7…	"sql_injection"	"query"	{} 1 fields
23	ObjectId('660972efad84ec7…	"xss"	"payload"	{} 1 fields
24	ObjectId('66097416ad84ec7…	"sql_injection"	"query"	{} 1 fields
25	ObjectId('66097a0aad84ec7…	"file_inclusion"	"redirect"	{} 0 fields
26	ObjectId('66097daaf6f5a90…	"xss"	"email"	{} 5 fields

Fig.4.4 Real time Attack Data storge

Our research investigates the development of a Web Application Firewall (WAF) equipped with an intuitive admin console. This console empowers website owners to interact with the WAF and visualize attack reports. The reports are presented in clear bar charts, categorized by the type of attack detected. To facilitate data storage and retrieval, we leverage a MongoDB database.

6. CONCLUSIONS

In This paper has examined the evolving landscape of web application security and the growing importance of neural network-based web application firewalls (WAFs) in protecting online platforms. As cyberattacks become increasingly intricate and sophisticated, the cybersecurity community requires innovative solutions to effectively mitigate these risks. Neural networks, with their ability to learn complex patterns and detect anomalies, demonstrate remarkable potential for strengthening web application resilience against a broad spectrum of attacks.

Our exploration of various methodologies, including data collection, preprocessing techniques, and evaluation metrics, has provided insights into the practical implementation of neural network-based WAFs. By leveraging the capabilities of neural networks, this system offers real-time defense against a wide variety of cyber threats. Furthermore, the inherent adaptability of neural networks allows the WAF to adjust to the ever-changing threat landscape, making it a vital asset for securing web applications in the face of dynamic and persistent security challenges.

REFERENCES

(1) Dawadi, B. R., Adhikari, B., & Srivastava, D. K. (2023). Deep Learning Technique-Enabled Web Application firewall for the detection of web attacks. Sensors,
(2) Demilie, W. B., & Deriba, F. G. (2022),Detection and prevention of SQLI attacks and developing compressive framework using machine learning and hybrid techniques. Journal of Big Data,
(3) Jemal, Ines, et al. ‘SWAF: A Smart Web Application Firewall Based on Convolutional Neural Network’. 2022 (SIN), IEEE, 2022.
(4) Satyanarayana, D., and Aisha Said Alasmi. ‘Detection and Mitigation of DDOS Based Attacks Using Machine Learning Algorithm’. 2022(ICCR), IEEE.
(5) Joshi, Nikita, et al. ‘A Detailed Evaluation of SQL Injection Attacks, Detection and Prevention Techniques’. 2022(ICAST), IEEE, 2022.
(6) En, Voo Teck, and Vinesha Selvarajah. ‘Cross-Site Scripting (XSS)’. 2022 (ICMNWC), IEEE, 2022.
(7) Zhao, C., Si, S., Tu, T., Shi, Y., & Qin, S. (2022). Deep-Learning based Injection Attacks Detection Method for HTTP. Mathematics,
(8) Shaheed, A., & Kurdy, M. (2022). Web application firewall using machine learning and features engineering. Security and Communication Networks.
(9) Gogoi, Bronjon, et al. ‘Defending against SQL Injection Attacks in Web Applications Using Machine Learning and Natural Language Processing’. 2021, IEEE.
(10) Ito, Michiaki, and Hitoshi Iyatomi. ‘Web Application Firewall Using Character-Level Convolutional Neural Network’. 2018 IEEE (CSPA).
(11) Hu, H. (2017). Research on the technology of detecting the SQL injection attack and non-intrusive prevention in WEB system.
(12) A. Begum; Md. M.Hassan;T. Bhuiyan; Md. Hasan Sharif,(2016) RFI and SQLi based local file inclusion vulnerabilities in web applications of Bangladesh. (2016, December 1). IEEE Conference Publication | IEEE Xplore.
(13) Stephan, J. J., Mohammed, S. D., & Abbas, M. K. (2015). Neural network approach to web application protection. International Journal of Information and Education Technology.