Vulnerability Deterrence System Using Web Application Firewalls- Deep Learning & Reinforcement Learning

Angadh Ramnath[1]
[1]Student, Dept. Of Computer Science & Engineering,Mangalam College of Engineering, India

Devika Nandakumar[3]
[3]Student, Dept. Of Computer Science & Engineering, Mangalam College of Engineering, India

Dacena C Martin[2]
[2]Student, Dept. Of Computer Science & Engineering, Mangalam College of Engineering, India

Elson P Aniyan[4]
[4]Student, Dept. Of Computer Science & Engineering, Mangalam College of Engineering, India

Jinu P Sainudeen[5]
[5]Assistant Professor, Dept. Of Computer Science & Engineering,
Mangalam College of Engineering, India

***Abstract*— Web Application Firewalls (WAFs) play a crucial role in securing numerous web applications from increasing sophisticated web attacks. However, as web attacks become more advanced, regular testing and updates are required to ensure the WAFs can withstand the attacks. Brute-force attacks are impractical due to the vast array of attack patterns. Therefore, black-box testing techniques have been proposed, but they are not yet mature and suffer from low efficiency and effectiveness. This paper proposes a method for vulnerability detection in web applications using a web application firewall (WAF) integrated with deep learning and reinforcement learning techniques, specifically convolutional neural networks (CNNs) and artificial neural networks (ANNs). The method aims to improve accuracy and reduce false positives by training the CNN and ANN models using a dataset of known vulnerabilities and employing reinforcement learning techniques. Experimental results show that the proposed method outperforms traditional vulnerability detection methods in terms of accuracy and reduced false positives. The proposed method has the potential to be an effective tool for enhancing the security of web applications.**

***Keywords*— Vulnerability detection, SQL injection, XSS attack, Web Application Firewall, Deep Learning, Reinforcement Learning, CNN, ANN.**

I. INTRODUCTION

In recent years, there has been a significant shift towards online businesses, such as e-commerce, online banking, and social media. This transformation has led to the accumulation of large amounts of private data belonging to both individuals and organizations in web application databases. Unfortunately, this data has made web applications an attractive target for attackers. Recent reports indicate that web applications may experience up to 26 attacks per minute, and 76% of websites are vulnerable to several attacks, according to Symantec's security report [7][8]. This situation highlights the need for robust security measures to protect against these attacks and safeguard sensitive data.

Vulnerability detection using a web application firewall (WAF) is a proactive approach for identifying and mitigating security vulnerabilities in web applications. A WAF is a type of security solution that sits between a web application and the internet, analyzing incoming traffic and blocking any malicious requests [1]. One of the primary benefits of using a WAF for vulnerability detection is that it can provide continuous monitoring and protection against a wide range of attacks, including SQL injection and cross-site scripting (XSS). By analyzing incoming traffic in real-time, a WAF can quickly detect and block any requests that appear to be malicious or anomalous, providing an additional layer of defense against potential vulnerabilities.

In addition to providing real-time protection, many WAFs also offer reporting and analytics features that can help organizations identify trends and patterns in their web traffic, as well as detect any potential vulnerabilities that may need to be addressed. Overall, vulnerability detection using a WAF is an effective way to proactively identify and mitigate security vulnerabilities in web applications, helping to prevent cyber-attacks and protect sensitive data.

WAFs are a recommended solution for protecting web applications against attacks. WAFs analyze HTTP(S) traffic to prevent malicious requests from reaching the web applications, intercepting bi-directional traffic and making decisions based on whether the traffic is malicious or benign, according to the

Open Web Application Security Project's definition [1]. WAF can be deployed as a hardware appliance, virtual machine, or cloud-based service and can be configured to block, allow, or modify traffic based on predefined security rules. WAFs also provide additional security features, such as encryption, authentication, and access control, to protect web applications against unauthorized access and data breaches. However, WAFs are not a standalone solution and should be used in conjunction with other security technologies and best practices.

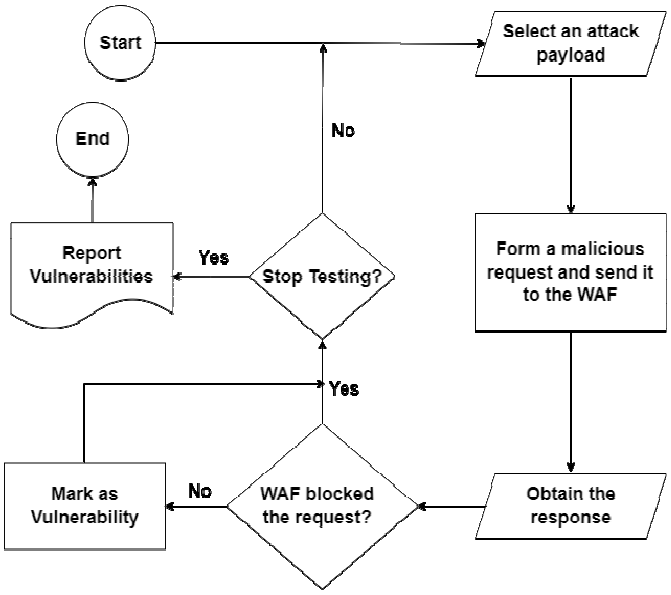

Fig. 1: The procedure for testing a WAF to discover vulnerabilities

II. BACKGROUND

This section is intended to provide a brief description of code injection attacks, such as SQL Injection and Cross-Site Scripting.

A. SQL Injection

The management and modification of relational databases are commonly performed using SQL, a programming language that enjoys widespread usage among its users. SQL is popular in web development, data analytics, and business intelligence applications. It allows the creation of relationships between tables to manage complex data structures.

SQL injection is a type of cyber-attack that exploits vulnerabilities in web applications to inject malicious SQL statements into a database. This can allow an attacker to gain unauthorized access, modify or delete data, and even take control of the entire database. Attackers typically manipulate input fields on a web application, such as username or password fields, to include SQL code that tricks the application into executing its code instead of the intended query.

Effective prevention of SQL injection attacks in web applications requires web application developers to implement security measures such as input validation and the use of parameterized queries. Input validation ensures that user input is properly formatted and sanitized to prevent malicious code from being included. Parameterized queries use placeholders in SQL statements to prevent malicious code from being executed.

B. Cross-Site Scripting (XSS)

XSS (Cross-Site Scripting) is a type of cyber-attack in which an attacker injects malicious code into web pages that are viewed by other users. This is done by exploiting vulnerabilities in a web application that allows the attacker to inject code into the page, which can then be executed by unsuspecting users who view the page.

The most common type of XSS attack is known as "reflected XSS," in which an attacker includes malicious code in a URL or form input, which is then reflected to the user in the page's response. When the user views the page, the malicious code is executed, allowing the attacker to steal sensitive information or take control of the user's session.

To prevent XSS attacks, it is important to implement proper security measures in web applications, such as input validation and output encoding. Input validation involves ensuring that user input is properly formatted and sanitized to prevent malicious code from being included. Output encoding involves converting special characters into their HTML or URL-encoded equivalents, which prevents malicious code from being executed when the page is rendered.

III. RELATED WORK

There has been significant research on vulnerability detection using web application firewalls (WAFs). Some of the relevant works in this area are:

1) Detecting attacks on web applications using an autoencoder is a method which provides high accuracy. Autoencoders can be highly accurate in detecting anomalies or outliers, which makes them effective in detecting attacks. Although it is limited to only known attacks. Autoencoders may not be able to detect novel or previously unseen attacks that have not been observed during the training phase [2].

2) Another testing technique is the Search-based security testing of Web Applications which provides wide coverage of test cases. Search-based security testing can potentially cover a large space of test cases, allowing for more comprehensive test coverage of the web application. Search-based security testing can be computationally expensive and time-consuming, especially when searching for vulnerabilities in complex applications [3].

3) Adaptive random testing for XSS vulnerability is a technique which provides Automation. Adaptive random testing can be automated, allowing for more efficient testing and faster detection of XSS vulnerabilities. disadvantages: Limited scope: Adaptive random testing is focused on XSS vulnerabilities only, and may not detect other types of security vulnerabilities that may be present in the web application [4].

4) XSS vulnerability detection using model inference-assisted evolutionary fuzzing has the advantage of

Adaptability. The machine learning model used in the technique can be retrained and updated as new vulnerabilities are discovered or as the web application being tested evolves. This helps ensure that the technology remains effective and up-to-date. While this technique can be effective at identifying certain types of XSS vulnerabilities, it is not a foolproof solution. There may be other types of vulnerabilities or attack vectors that the technique is not designed to detect, which can lead to a false sense of security [5].

5) Deep continuous clustering is another technique with the ability to handle large datasets. Deep continuous clustering can handle large and complex datasets with high dimensionality. Deep continuous clustering can produce complex and difficult-to-interpret results [6].

Overall, these works highlight the importance of using WAFs as an effective defense mechanism against web application vulnerabilities and demonstrate the potential of combining rule-based and machine learning-based approaches for improved detection accuracy.

IV. METHODOLOGY

A. Proposed System

The proposed system for vulnerability detection using a web application with the application of Deep Learning and Reinforcement learning along with Convolutional Neural Networks (CNN) and Artificial Neural Networks (ANN) consists of several key components.

- Firstly, the system includes a dataset module which collects and stores data related to software and system vulnerabilities. This dataset serves as the foundation for the system's learning and decision-making capabilities.
- Next, the data preprocessing module is responsible for preparing the data for analysis by cleaning and transforming it into a format that can be used by the feature selection and model-building modules.
- The feature selection module uses various techniques to identify the most relevant features from the dataset that will be used to train the models. This helps to reduce the complexity of the data and improve the accuracy of the models.
- The model-building module uses both CNN and ANN to train the models based on the selected features. CNN is used to analyze and extract features from images related to vulnerabilities, while ANN is used to analyze and extract features from other types of data, such as text or numerical data.
- The system also includes a splitting data module that divides the data into training and testing sets, which are used to evaluate the performance of the models.
- Once the models are trained and tested, the evaluating model module analyzes the performance of the models and provides feedback on how to improve them. This module also uses Reinforcement learning to continuously improve the system's decision-making capabilities over time.
- Finally, the input data module receives data related to vulnerabilities from the user and feeds it into the trained models. The trained models then analyze the input data and provide an output indicating whether the data is associated with a vulnerability or not.

Overall, the proposed system for vulnerability detection using a web application with the application of Deep Learning and Reinforcement learning along with CNN and ANN provides an effective and efficient way to detect vulnerabilities in software and systems, with the potential to adapt and improve over time.

B. Algorithm

a) ANN Algorithm: Training to recognize patterns

The use of Artificial Neural Networks (ANNs) in vulnerability detection with web application firewalls (WAFs) is aimed at improving the accuracy and effectiveness of WAFs in identifying and blocking malicious traffic. ANNs can analyze large datasets of labeled traffic to learn complex patterns and relationships, allowing them to identify potential security threats such as SQL injection and cross-site scripting attacks.

By analyzing traffic data using ANNs, WAFs can automatically block traffic that matches those patterns, which can help to protect web applications against emerging threats. The use of ANNs in vulnerability detection with WAFs can enhance the overall security posture of web applications and improve the ability of organizations to detect and block potential security threats.

b) CNN Algorithm: Feature Extraction

CNNs can also be applied in vulnerability detection using web application firewalls (WAFs) to enhance their ability to detect and block potential security threats. The input data is represented as a sequence of characters or tokens extracted from the HTTP request. The CNN processes this sequence through a series of convolutional and pooling layers to automatically learn relevant features and relationships between tokens, such as common attack patterns or unusual behavior. The output of the CNN is then passed through fully connected layers to produce a final prediction or classification, indicating whether the request is malicious or not.

By using CNNs, WAFs can automatically and adaptively learn relevant features and patterns in web traffic, without the need for manual feature engineering. This can improve the accuracy and effectiveness of security defenses and help organizations better protect against emerging threats. The application of CNNs in vulnerability detection with WAFs can aid in the identification of potential security threats such as SQL injection and cross-site scripting attacks by analyzing the content of web requests.

Once the CNN has been trained, it can quickly classify new traffic data as either normal or malicious with a high degree of accuracy. This allows the WAF to focus on analyzing only the potentially malicious traffic, rather than wasting resources analyzing normal traffic. CNNs can also be used to detect anomalies in web traffic that may be indicative of new or unknown attacks. By comparing new traffic data to the learned patterns of normal traffic, CNN can identify deviations from normal behavior that may be suspicious.

C. System Architecture

The system architecture aims to improve the accuracy of vulnerability detection and reduce false positives by leveraging the capabilities of CNNs and ANNs to analyze large amounts of data and learn patterns that may be indicative of a vulnerability. The system is trained using a dataset of known vulnerabilities to enable the models to learn to identify patterns that are indicative of vulnerabilities. The system also employs reinforcement learning techniques to further enhance the accuracy of vulnerability detection.

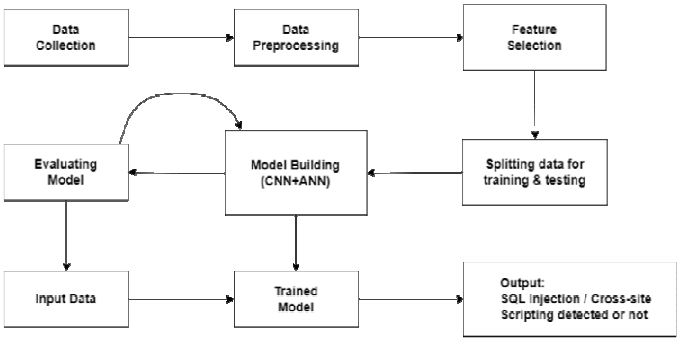

Fig. 2: System Architecture

D. *Data Flow Diagram*

1. *Level 0- DFD*

Fig. 3: Web Application Firewall Data Flow Diagram

2. *Level 1- DFD*

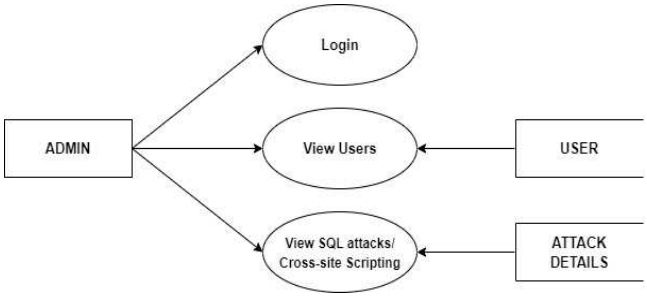

Fig. 4: Admin-login Data Flow Diagram

3. *Level 1.1- DFD*

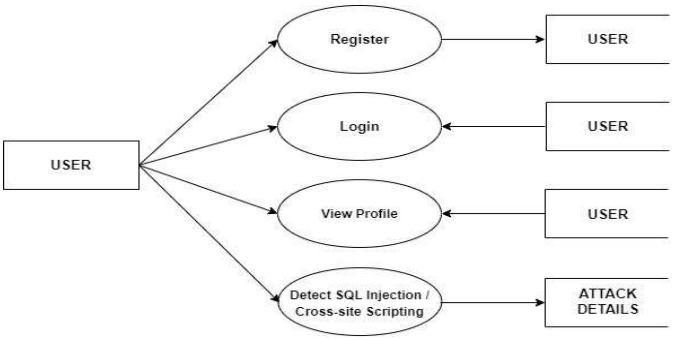

Fig.5 : User-login Data Flow Diagram

V. RESULT

Users can access a website by logging in with their unique user ID and password. New users can also register and create an account by choosing a unique user ID and password. However, if an attacker attempts to gain unauthorized access by injecting SQL queries or using Cross-site Scripting (XSS) attacks, the trained Web Application Firewall detects the vulnerability and prevents the intrusion by displaying a message that the vulnerability is detected. Experimental results demonstrate the effectiveness of the proposed method in detecting vulnerabilities in web applications, making it a valuable tool for enhancing the security of web applications.

VI. FUTURE SCOPE

The efficiency of a WAF can be improved by focusing on building comprehensive datasets with a minimal number of samples. Continuously training the dataset can make the model more resistant to false cases.

VII. CONCLUSION

In today's modern society, the consequences of neglecting privacy and security in cyberspace can be devastating. Therefore, Web Application Firewalls (WAFs) play a prominent role in protecting web applications. However, since attacks are increasing in occurrence and sophistication, WAFs need to be regularly tested and updated. Failure to do so may result in attacks circumventing the WAF and posing a threat to the applications under its protection.

In conclusion, we have proposed a novel approach to enhance the security of web applications by integrating Deep Learning and Reinforcement Learning with Web Application Firewalls (WAFs). Our approach uses Convolutional Neural Networks (CNNs) and Artificial Neural Networks (ANNs) to classify and predict malicious traffic, while Reinforcement Learning enables the WAF to learn and adapt to new and evolving threats over time. It has shown promising results in detecting and preventing attacks on web applications. The system architecture and modules, including dataset, data preprocessing, feature selection, evaluating model, model building, splitting data for training and testing, input data, trained model, and output, have been designed to ensure

effectiveness, efficiency, and adaptability in detecting vulnerabilities. With the integration of reinforcement learning, the system can continuously learn and update its policy based on new data and feedback from the environment, providing an additional layer of adaptability and resilience against evolving threats. Overall, the developed web application firewall offers a robust and reliable solution for protecting web applications in today's increasingly complex and dynamic cyber landscape.

VIII. ACKNOWLEDGMENT

The authors express their gratitude to Vinodh P Vijayan, Principal and Neethu Maria John, HOD of the Department of Computer Science and Engineering, for providing invaluable guidance, support, and insightful comments during the proofreading process.

REFERENCES

[1] Mohammad Hossein Amouei, Mohsen Rezvani, Mansoor Fateh, “RAT: Reinforcement-Learning-Driven and Adaptive Testing for Vulnerability Discovery in Web Application Firewalls” in IEEE Transactions on Dependable and Secure Computing, 2021.

[2] H. Mac, D. Truong, L. Nguyen, H. Nguyen, H. A. Tran, and D. Tran, “Detecting attacks on web applications using autoencoder,” in Proceedings of the Ninth International Symposium on Information and Communication Technology. ACM, 2018, pp. 416–421.

[3] J. Thom´e, A. Gorla, and A. Zeller, “Search-based security testing of web applications,” in Proceedings of the 7th International Workshop on Search-Based Software Testing, 2014, pp. 5–14.

[4] C.Lv, L. Zhang, F. Zeng, and J. Zhang, “Adaptive random testing for xss vulnerability,” in 2019 26th Asia-Pacific Software Engineering Conference (APSEC). IEEE, 2019, pp. 63–69.

[5] BF. Duchene, R. Groz, S. Rawat, and J.-L. Richier, “XSS vulnerability detection using model inference assisted evolutionary fuzzing,” in 2012 IEEE Fifth International Conference on Software Testing, Verification and Validation. IEEE, 2012, pp. 815–817.

[6] S. A. Shah and V. Koltun, “Deep continuous clustering,” arXiv preprint arXiv:1803.01449, 2018.

[7] D. E. Simos, B. Garn, J. Zivanovic, and M. Leithner, “Practical combinatorial testing for xss detection using locally optimized attack models,” in 2019 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW). IEEE, 2019, pp. 122– 130.

[8] K. Chandrasekar, G. Cleary, O. Cox, H. Lau, B. Nahorney, B. O. Gorman, D. O’Brien, S. Wallace, P. Wood, and C. Wueest, “ISTR Symantec Volume 22,” Tech. Rep. April, 2017.