# An enhanced Web Application Firewall Security for real-time DDOS detection using Deep Learning and integrating LSTM models into Modsecurity Firewall

Haithem Ali Alghazzawi[1]
The Libyan Academy
School of Applied Sciences and Engineering -Tripoli, Libya
210100071@academy.edu.ly

Mohamed Elbeshti[2]
Information Technology School
University of Zawia Department of Information Technology
m.elbeshti@zu.edu.ly

Abdalrahman Alfagi[3]
Department of Information Technology, IT faculty, University of Zawia
Abd.alfagi@zu.edu.ly

***Abstract-*** ***Web applications are the backbone of most digital online services, whereby clients communicate with the server, and enormous amounts of sensitive data are processed in real-time. Consequently, sensitive information may be compromised or become vulnerable to several threats, or it may disrupt information availability, which directly threatens Confidentiality, Integrity, and Availability, causing financial loss, legal consequences, erosion of user trust, or business discontinuity. Based on recent statistics, DDoS attacks represent 61% of web attacks, while traditional web application firewall capabilities haven't met modern environment needs to effectively and successfully protect web-based applications from DDoS attacks. Therefore, an enhanced WAF is proposed to protect an organization's web applications and their resources from DDoS attacks. A deep learning approach using an LSTM model (DDoS detection layer) positioned beside the WAF to enhance its capabilities for real-Time DDoS detection. This layer analyzes all incoming traffic to identify the abnormal requests and then feeds its output to the WAF, helping the WAF respond only to legitimate traffic. Thus, both the DDoS layer and the WAF work together to detect and block DDOS attack. The integrated system was tested using evaluation metrics such as accuracy and loss. The results show the ability of the system to process 500–800 requests per second, had a 2–5 ms average response time, and consumed a low amount of computing resources (50–80 MB memory, 10–15% CPU). The drop in the throughput around 20% (to 400–600 req/s), an increase in latency up to 20–30 ms, a rise in memory usage up to 200–300 MB, and a CPU consumption of 25–40%. The tradeoff was very much worth it because the integration resulted in a significant improvement in detection and security overall efficiency. The experimental results provide evidence that the proposed LSTM models led to a better enhancement in the detection capabilities of the WAF systems over the traditional methods. This approach represents a highly important and practical method of relying upon deep learning for traditional WAF systems.***

***Keywords- Web Application Firewall, Deep Learning, Attack Detection, Firewall, and DDoS.***

## 1. INTRODUCTION

Web applications have a significant impact on our daily life; they are essential tools in modern life due to their flexibility and accessibility. Web applications provide several indispensable services that streamline communication, E-commerce, education, entertainment, finance, e-banking and countless online services across personal, professional, and societal domains that could be accessed by any smart device with internet access [3]. In fact, web applications became the backbone of the digital transformation, dominating personal, individual or governmental organizations to operate smarter, faster, and more flexibly. By the nature of web applications, they are software programs that are accessed through clients' browsers and hosted on remote servers. Basically, web applications process data in real-time when the clients communicate with the server [4].

Despite the effectiveness of web applications in facilitating daily modern life, dependence on these applications certainly makes them potential targets for cyberattacks. Therefore, securing sensitive information of personal, individual or governmental organizations is a very significant concern and critical issue that most researchers and those interested in information security focus on since web applications handle enormous amounts of sensitive information at real-time. Consequently, information may be compromised, vulnerable to several threats or may disrupt information availability which cause a financial loss, legal consequences, erosion of user trust or business discontinuity. With the use of networks, the problems of preserving the heart of the information security which are the Confidentiality, Integrity, and Availability are compounded. These three concepts are often referred as the CIA triad [5]. For instance, DDoS (Distributed Denial of Service) attack is a common threat where attackers' goal is to affect user trust and business continuity by making web applications, server, or network resources unavailable to legitimate entities. In this type of attack, hackers intentionally exploit weaknesses in the network by repeatedly sending initial connection request packets. Attackers use several DDoS methods such as SYN floods or Ping of Death, targeting specific apps or services, HTTP floods or flooding bandwidth with massive traffic, UDP floods, DNS amplification. For defending against DDOS attacks, traditional Web applications Firewall (WAF) is designed as a security mechanism to secure web applications from a variety of threats, including DDoS. WAFs often struggle to eliminate modern threats such as zero-day attacks, polymorphic malware, flooding bandwidth with massive traffic, sophisticated injection techniques and excessive API calls [2].

According to recent statistics [6], DDoS recorded a higher rate with a widespread spectrum of targets it causing a significant effect for personal, individual, or governmental organizations, and online gaming sites. Consequently, DDoS caused a financial loss, legal consequences, erosion of user trust, or business discontinuity. Therefore, WAF is an active area where most researchers' goal is to enhance the robustness of the WAF to make it able to defend against

DOI:10.5281/zenodo.18108194

modern DDoS attacks. In this article, the authors proposed an enhanced WAF model for real-time DDoS detection using state-of-the-art technologies such as machine learning (ML), Long Short-Term Memory (LSTM) technique and artificial intelligence (AI).

## 2. Background

This section provides an overview and discusses web applications' common attacks, with more focus on DDoS. It also introduces the general structure of traditional WAFs as well as explore the Long Short-Term Memory.

### *2.1 Common web application attacks*

Internet usage is being widely spread in different domains, facilitating daily modern life. This expansion led to several attacks due to uncontrolled vulnerabilities in web application-based systems. As stated by data breach investigations report [6], every website is vulnerable to cyber-attacks that could threated one of the CIA attributes. Whereas the authors in [7], confirms denial of service (DoS) and DDoS are usually recorded as the most frequently occurring attacks. In addition, Neustar Cyber Threats and Trends Report states that the DDoS increased by 200% in 2019, while this percentage doubled by the end of 2023. Furthermore, the report predicted an increase of 151% in the number of attacks they stated every year. Recently, in 2025 Verizon [6] report that DDoS affected 61% of web applications and their resources availability as well as the report ordered the DDoS at the second top attacks based on incident classification patterns. Therefore, the reports provide strong evidence for the need of deeply explore the DDoS methods to understand how attackers exploited these forms to conduct a successful attack. In addition, the percentage reported by Neustar Cyber Threats and Trends Report, and Data Breach Investigations are a high indicator to develop a new solution to protect organization web applications and their resources from DDoS effectively and successfully or enhance the existing protection model. Due to these evidences the authors of this article aimed to propose an enhanced Web Application Firewall Security for real-time DDOS detection.

The DoS and DDoS have several forms and methods that allow the attackers to conduct a severe attack to achieve their goals. The authors in [7] [2] [8] mentioned some of the DDoS forms, such as User Datagram Protocol (UDP) floods, Flood SYN, TCP, ICMP, and HTTP. However, the authors in this article categorize the DDoS attacks based on the attacker's aim and DDoS method as shown in the **table1**: Categorization of DDoS attacks.

Table 1: Categorization of DDoS attacks

| DDoS Category | Attackers aim | Attack method/ forms |
|---|---|---|
| Volumetric Attacks | Saturate the bandwidth of the target | UDP flood, ICMP flood, DNS amplification |
| state-exhaustion attacks, also named Protocol DDoS attacks | Consume server resources | SYN flood, ACK flood, Smurf attack |
| Application Layer Attacks | Target the application layer to exhaust server resources | HTTP, or HTTPS Flood, including GET or POST |
| Reflection Attacks, also known as an amplification attack | Exhausting bandwidth to cause unreachable services | DNS, NTP, and other UDP-based services |
| Multi-vector Attacks, also known as a mixed DDoS attack | Availability of the services and resources | Combine more than one of the above (Volumetric, Protocol DDoS, or application layer attack) |

### *2.2 Web Application Firewall*

WAF is an important security mechanism implemented in most mobile, APIs, and web application, as the first security defense. It is deployed in the front of web applications in order to detect anomalous traffic for the goals of filtering, monitoring in coming or outgoing packets aiming to mitigate several attacks that are most common within web traffic for protecting CIA attributes [8]. Figure 1 depicts the general structure of the WAF. The WAF operates based on a set of predefined rules mostly known as polices. These pre-trained rules are responsible for predict new incoming requests.

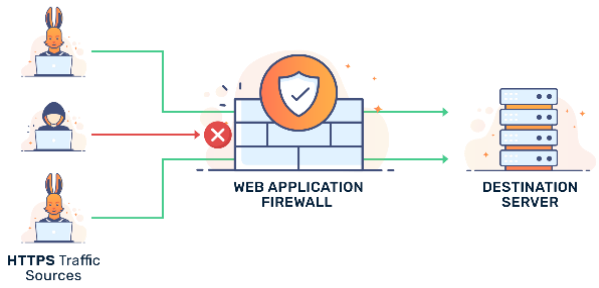



Figure: 1 Web Application Firewall General Structure [2]

The capabilities of the WAF in protecting the web applications mostly depends on the type of firewall which are packet filtering, stateful inspection, application level, and circuit level firewall. Generally, all of firewall types uses four defensive techniques which are service, direction, user and behavior control to dominate access and to enforce the sites' security policy. Regardless, the firewall types, the defensive techniques, or how the firewall operates, weather it operates as a positive filter in which only the packets that meet predefined policies are allowed or as a negative filter where it rejects any packets that do not meet the specified criteria. The authors in [9-12] stated many limitations of the firewall include that the firewalls are unable to protect applications against attacks that bypass the firewall as well as firewall fails in protecting against internal threats or the transfer of virus-infected programs. In addition, the authors in [12] argue that



DOI:10.5281/zenodo.18108194

the firewall may not guarantee full protection threats like malware propagation can still occur which clearly mean firewalls alone aren't sufficient. Even though, the WAF intercepts all traffic that incoming or outgoing between the web application and the internet, still traditional WAFs haven't met the modern environment needs to effectively and successfully protect web-based applications against sophisticated attacks since attackers became smarter to find or use advanced techniques and tactics to exploit vulnerabilities [2]. These means the firewall is important and necessary but not sufficient alone they must be a part of in-depth defense. Therefore, an enhanced Web Application Firewall Security is an active field that need a robust WAF to protect organization web applications and their resources from threats and attacks. From this prospective, the authors of this article use deep learning, and artificial intelligence (AI) to enhance WAF capabilities for Real-Time DDoS Detection.

### *2.3 Long Short-Term Memory (LSTM)*

The Long Short-Term Memory is a special type of Recurrent Neural Network (RNN) used in deep learning to process sequential data. It can pick up long-term dependencies while the module is being trained. The general structure of four interacting repeating module of LSTM is shown in figure 2. The LSTM processes the input from sequential input data and output from the previous cell [1, 13-15].

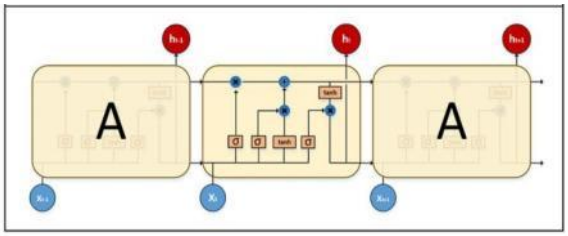



Figure: 2 The four interacting repeating module of LSTM RNN *[1]*

The LSTM model is capable to remember long-term dependencies using its internal cell state and gating mechanisms during the training of the model. The LSTM model consists of four layers integrated together namely; forget gate, input gate and candidate memory, cell state update, and output gate and hidden state [2, 13]. These layers are summarized in table 2, which it illustrates the layer sequence, the gate names, the functions of each layer, the formula that achieves the intended functions, and explains the symbols used in each equation.

Table 2: LSTM Layers summary

| Layer | Gate name | Functions | Equation | Symbols |
|---|---|---|---|---|
| First layer | Forget | Decides which information to keep or forget based on the sigmoid (σ) value; 1 to keep where 0 to forget | $f_t=\sigma(w_f . [h_{t-1}, x_t]+b_f)$ ➪① | $h_{t-1}$ = previous cell state<br>$x_t$ = input vector at time $t$<br>σ = sigmoid function<br>$w_f$ = weight function<br>$b_f$ = biases function<br>$i_t$ = input gate<br>$C_t$ = candidate memory (updated cell)<br>$f_t$ = vector multiplication<br>$h_t$ = hidden state<br>$o_t$ = output gate |
| Second layer | Input | Decides what new information to store | $i_t=\sigma(W_i . [h_{t-1}, x_t]+b_i)$ ➪② | |
| | Candidate | Generates the new potential value to be added | $C_t=\tanh(w_c . [h_{t-1}, x_t]+b_c)$ ➪③ | |
| Third layer | Cell state | Updates the cell state combining forget and input decisions | $C_t=f_t . C_{t-1} + i_t . C_t$ ➪④ | |
| Fourth layer | Output Gate | Decides what to output at $t$ time step | $o_t=\sigma(w_o . [h_{t-1}, x_t]+b_o)$ ➪⑤ | |
| | Hidden state | Hidden state is passed to the next time step and/or output layer. | $h_t=o_t . \tanh(C_t)$ ➪⑥ | |

The first layer which is the forget gate layer known as the decision-making layer. The LSTM looks at the information from the previous cell state ($h_{t-1}$) and the input at time $t$ ($x_t$) then decides which to keep or forget based on the sigmoid function (σ). If the sigmoid is 0 means forget signals and if the sigmoid is 1 means memorize the signals. While both the weight ($w_f$) and the biases ($b_f$) remain the same for the iterations. The second layer in LSTM is represented in the input gate ($i_t$), and candidate memory ($C_t$) as equations ② and ③ represent them respectively. The ($i_t$) decides what new information to store while the ($C_t$) generates the new potential value to be added. The third layer in LSTM model is the cell state update ($C_t$) where it retains important long-term memory and updates with new cell state combining the forget and input decisions. Equation ④ achieves that by vector multiplication of ($f_t$) and the old cell state ($C_{t-1}$). The last layer named the output gate ($o_t$) and hidden state ($h_t$) decides what to output at the time ($t$) step as in equation 5 and what hidden stat is passed to the next time step and or output layer.

## 3. Related work

In 2022, the authors in [3] adopted the LSTM model for detecting anomalies in Web application attacks, such as SQL injection and cross-site scripting. Research was worked to develop a WAF system based on anomaly detection using deep learning methods based a semi-supervised approach. The model was trained using a combination of normal and attack payloads. The study included a real-world dataset from the "Payload All the Things" repository and normal payloads from the HTTP Dataset CSIC 2010. The input data was preprocessed by converting the character sequences into a lower-dimensional vector representation using an embedding layer. In this study, the authors ignore the HTTP response data in the analysis, which could provide additional information for improving the detection accuracy. In addition, the researchers recommended more exploration by using their trained model in analysing the HTTP response data and incorporating it into the models, which could be a potential direction for future work. Similarly, Román-Gallego and others [16] focused on improving the detection capabilities of Web Application Firewalls ModSecurity, particularly in reducing false positives associated with complex SQL injection attacks. They proposed a novel approach to dissect HTTP requests to enhance security rule management by using various supervised machine learning techniques, including Naïve Bayes (NB), K-nearest neighbours (KNN), Support Vector Machines (SVM), and Linear Regression (LR), to classify malicious and valid HTTP requests. They also discuss techniques for data preprocessing, such as vectorization and obfuscation, to create a synthetic dataset for training and testing the models. However, the improved WAF was proposed for detecting SQL injection attacks.

Another study by [17] focused on the HTTP protocol, where the n-gram feature extraction model was used to extract features for model development. The authors use three different machine learning models with the CSIC2010 and ECML/PKDD2007 datasets, and compared the performance of these models to verify which had better performance as a web application firewall in the detection of anomalies.



DOI:10.5281/zenodo.18108194

Furthermore, WAF enhancement using machine learning is still an active area where the authors in [18] worked to create a web application firewall is able to identify frequent online threats by utilizing feature engineering and machine learning approaches. The primary goal was to propose a WAF model that detects frequent online threats using features to address the limitations of previous works by extracting more comprehensive features from the entire HTTP request, including URL, payload, headers, and files. Input length, alphanumeric character ratio, special character ratio, and attack weight are the last four characteristics they extract from the HTTP request. These attributes are calculated depending on the basic elements collected from the request, such as HTTP method, URL, payload, headers, and files. The authors evaluate their proposed model using four classification algorithms namely naïve bayes, logistic regression, decision tree, and support vector machine. Aref and the other authors use four datasets (CSIC 2010, HTTP Params2015, a hybrid dataset, and a custom dataset of a compromised web server).

On the other hand, Tariqul Islam and others [19] aimed at the conceiving a method for identifying DDoS attacks by artificial intelligence with high accuracy. They worked with a dataset on DDoS attacks. The authors use two approaches: firstly, trying unsupervised attack detection, that is, the computer tries to find some pattern in the data on its own. Secondly, supervised learning, where the computer is trained on labelled data (with traffic labels known beforehand). For the supervised method, a deep learning model was used, and for the validation of deep learning models, stratified K-fold cross-validation was adopted to mitigate the risk of having poor testing due to an imbalanced distribution of attack classes in the dataset. In their results, the unsupervised methods showed that clustering attack types is possible to some extent. Similarly, the authors in [4] worked on WAF to detect DDoS, where they developed a real-time method for detecting DDoS attacks using machine learning algorithms within an SDN environment. The study included an analysis of various features of packet flow, such as UDP flood attacks, ICMP ping flood attacks, TCP SYN flood attacks, and land attacks, to determine whether the given traffic was a normal or DDoS attack. The authors then compared the various ml algorithms, such as K-nearest neighbour, decision tree, random forest, and naive Bayes. However, the proposed WAF in [4] considers only a conventional or specific network environment, which restricts the planning and advancement of a response for overcoming a DDoS attack in real-time. In addition, in their WAF to detect DDoS attacks, it is required to address all features properly.

As the review study indicates, the DDoS affected 61% of web applications, and their resource availability. Only a few studies have been conducted to examine DDoS attacks based on different protocols, including HTTP requests, TCP, UDP, SYN, and NTP flood types. In addition, the majority of studies focus on well-known machine learning algorithms, lacking integration of the generated models in a web application firewall.

## 4. Methodology

The section illustrates the methodology and the enhanced WAF architecture based on LSTM neural networks for the aim of improving the WAF capability in detecting DDoS attacks in real-time. The idea is based on adding an extra layer named the DDoS detection layer. The goal is to combine the strengths of artificial intelligence with WAF for anomaly detection. The methodology consists of five phases, as shown in Figure 3, which illustrates the workflow of the proposed models, showing the sequence of steps from data acquisition to decision making. In the proposed method, a training dataset was used to train the enhanced WAF to identify malicious or legal new incoming HTTP traffic. The module parses a new HTTP request and extracts the parameters for prediction. It then uses the pre-trained module to make a prediction. If the traffic coming from HTTP is predicted to be normal traffic, the HTTP session is passed to the web server; otherwise, the HTTP session will be discarded/dropped in the WAF itself.

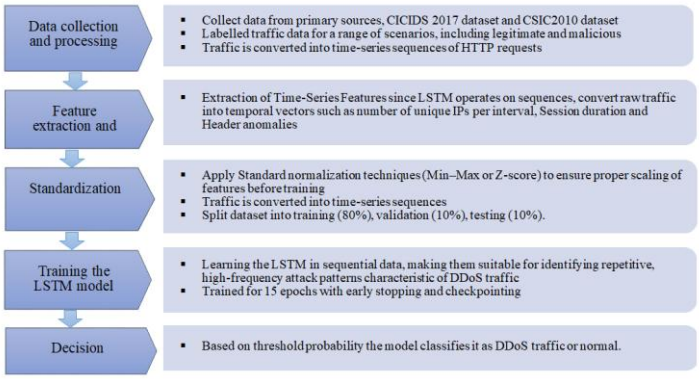



Figure: 3 Methodology

## 5. THE PROPOSED MODELS

As the traditional WAF mostly exists between the client and the web server for parsing and examining request and response units. Once the proposed LSTM model is trained then it is positioned within the WAF architecture to support its functions, as shown in Figure 4. The layer is designed to operate as an intelligent detection model for analyzing traffic (incoming) for the aim of identifying the potential DDoS patterns earlier, before the requests reach the web application. Integrating the DDoS detection layer into the WAF will enhance the WAF capability to recognize abnormal traffic behavior and provide a robust defensive mechanism against DDoS attacks in real-time. As shown in the figure, the traditional WAF is integrated with an additional AI-based detection layer. Under normal operations, the traditional WAF operates to block common web attacks. The additional DDoS detection layer is developed to learn the presence of both normal and DDoS attack traffic. As mentioned in the methodology, the dataset is divided into 80% training, 10 % validation and 10 % testing subsets. During training, the LSTM model adjusted its internal weights to minimize classification error, while the final model is selected based on its accuracy and ability to identify DDoS attack types. The DDoS layer evaluates the traffic and classifies the outputs into DDoS attacks or normal traffic based on the trained LSTM



DOI:10.5281/zenodo.18108194

model to simplify the process in the enhanced WAF, the incoming traffic (requests) enters the DDoS detection layer, then the traffic is analyzed to identify the abnormal request patterns. Simultaneously, the DDoS layer generates a real-time evaluation of potential DDoS activity in the requests. After that, the output of the DDoS layer is fed to the WAF. If the weight classifies the request as normal the WAF allow to response. If the DDoS fed the output is fed to the WAF as suspicious or high-risk the response to that particular request will be forbidden since it classified as DDoS attack. Accordingly, the DDoS layer classifies all incoming traffics helping WAF to response only to legitimate traffic. Thus, both DDoS layer and WAF work together for detection and blocking the DDoS attacks.

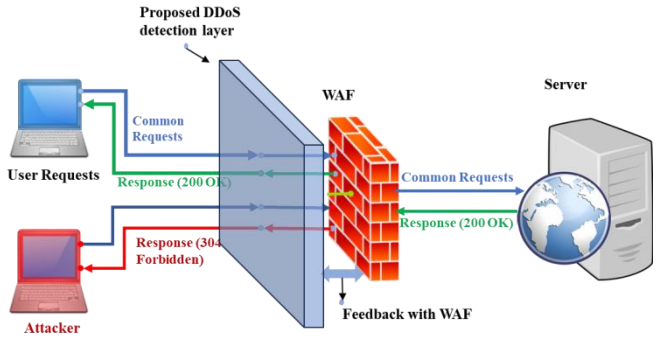



Figure: 3 The enhanced Web Application Firewall Security for real-time DDOS detection

### *5.1 Deployment Environment*

The testing and deployment environment is designed to ensure that the proposed DDoS layer operates as intended to be efficient and integrates seamlessly with the Web Application. The deployment environment consists of the hardware platform, software components, network positioning, scalability considerations, and security requirements. The LSTM model is tested on a virtual machine positioned alongside the WAF infrastructure. The hardware requirements are typically: CPU capable of handling concurrent traffic, minimum RAM is 8 to support sequential data processing and buffering and a GPU with high traffic rates for speeding up LSTM inferences. For the deployment, a Linux OS with Ubuntu server 24.4.02 LTS version was used because of its stability, strong security features, and great supportability for web servers as well as AI frameworks. The deployment process went through the following steps:

*Step 1: server setup*

During this step, a primary Apache web server is hosted on the platform, then the installation and enabling of the Modsecurity as a web application integrated with Apache. Finally, the PHP was enabled to support running the designed website.

*Step 2: Web interface design*

Python and Streamlet were used to create an intuitive online application that allows users to manually enter data or upload CSV files. The app preprocesses inputs, runs inference through the trained models, and displays results with tables and visual charts. The web interfaces consist of four pages, namely, the LSTM network traffic model tester, batch testing via CSV, the CIC2010 manual input and the prediction counts page. These pages are illustrated in Figure 5. The interface uses tab-based navigation to separate the pages. The LSTM network traffic model tester loads its respective LSTM model, and stored in .h5 format, and provides real-time classification results.

*Step 3: AI environment setup*

Since the proposed LSTM model is AI-based, Python 3.x and the required libraries such as Tensor Flow, Keras, and NumPy were installed and the trained models (ddosattack.h5) were placed in the /opt/ml/ directory. In addition, the integration script in Python (waf_ml_integration.py) was developed to load the models and perform classification.

*Step 4: Integration with Modsecurity*

Since the Modsecurity does not natively support AI models, a middleware solution was developed where incoming HTTP requests, intercepted by Modsecurity, are then forwarded to a Python engine via a shell script. The engine loads the trained models, analyses the request, and returns a decision to allow or block. Modsecurity enforces this decision, combining traditional rule-based filtering with AI-driven detection. the authors implement the intermediary script (ml_waf_analyzer.sh) to act as the connector between Modsecurity and the AI engine. They also define custom Modsecurity rules to capture request data, call the analyzer script, and enforce block/allow decisions based on the AI model. Figure 6 shows the proposed architecture for the inte.

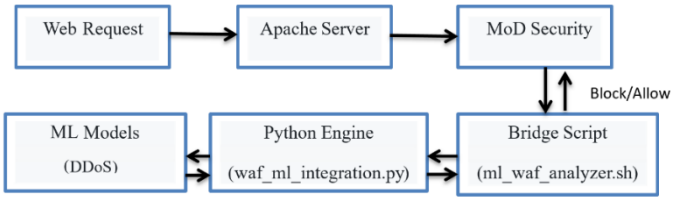



Figure: 4 The proposed architecture for the integration of Modsecurity

## 6.TESTING AND EVALUATION

For testing and verification, the model simulates both benign requests and DDoS attacks to validate the end-to-end functionality of the system. Then the system logs (ml_analyzer.log and ml_performance.log) were monitored to assess the detection accuracy and measure the performance. These were initiated by bringing up a terminal and executing the deployment script, as shown in Figure 6:

•#sudo bash deploy_ubuntu.sh
•#Site: http://localhost:8080/
•#Dashboard: http://localhost:8080/dashboard.php
•#Status: http://localhost:8080/integration_status.php
•#Test API: http://localhost:8080/test_models.php?input=test



DOI:10.5281/zenodo.18108194

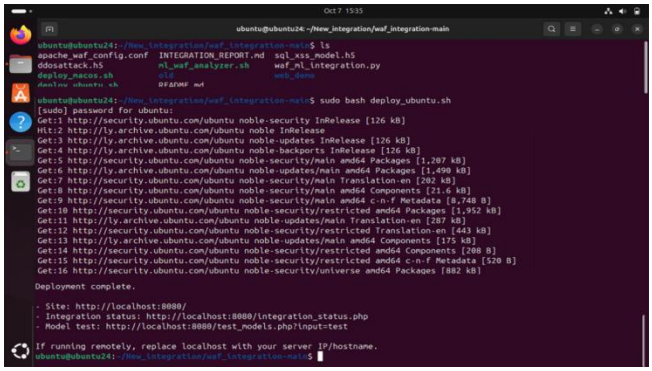



Figure: 5 Executing the deployment script

### *6.1 Evaluation criteria*

Evaluation is an important task in machine learning applications. Model evaluation is the process of measuring the performance of the built machine learning model. There are several measures or metrics for model evaluation, depending on the type of prediction. For evaluating the proposed model, an evaluation matrix and Accuracy and loss are used.

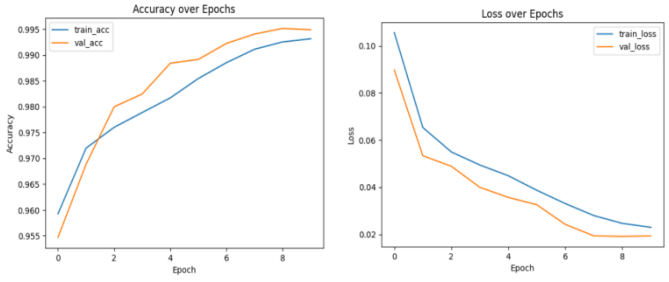
### *6.2 Accuracy and loss*

Model Accuracy: It is the main parameter used to evaluate classification models. The equation given by gives the accuracy of the model.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Number\ of\ Instances} \quad \text{Eq 1 [7]}$$

As the model's loss goes down, accuracy increases, meaning the model is making more correct predictions. Conversely, as loss increases, the accuracy usually decreases.

Figure 7 shows the accuracy and loss graph. the comparative accuracy of the training and validation procedure for the proposed model over 10 epochs, as shown in (a), the accuracy graph illustrates that training accuracy started at around 95% in the first epoch, slowly increased, and ended at about 99.3%. Likewise, validation accuracy climbed from 95.5% to about 99.5%. The validation accuracy usually remained above the training accuracy for most epochs, hinting that the model is well capable of generalizing on unseen data, and no obvious overfitting is visible. Consequently, these very outcomes prove the strength and dependability of the proposed model, affirming that it performs attack detection with very low error rates and thus can be suitably utilized for real-world web application security scenarios. By observing (b) the loss graph, the validation loss has started from a loss number 0.089, but it reduces epoch by epoch, and it is also closer to the training loss. After training the model for 10 epochs, the validation loss reaches to 0.023 at epoch 9.




(a) The accuracy graph (a) The loss graph

Figure: 6 The accuracy and loss graph for the proposed LSTM model

### *6.3 Confusion matrix*

A confusion matrix is created to test the DDOS detection model for the aim of assessing the efficiency. The y-axis represents the actual class, while the predicted is displayed on the x-axis. Looking at the confusion matrices, the testing data attained validation accuracies of 99.5% for the first model (DDOS). Thus, it can differentiate between malicious and normal requests.

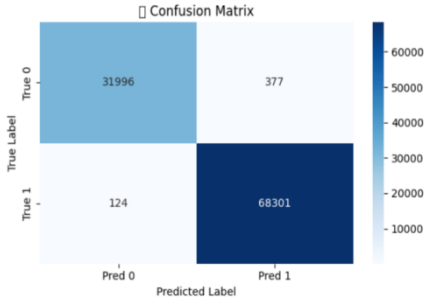



Figure: 7 the confusion matrix for the proposed DDOS Detection model

### *6.4 Evaluation Metrics (Precision, Recall, and F-measure) Analysis per Class*

By using the confusion matrix, it is possible to establish the standards of a given set of statistical metrics, each one tested individually against every class, to measure the efficiency of the model to distinguish between benign and attack sample sets. The DDOS model considered Precision, Recall, and F1-score as common parameters to practically interpret results from a classification model. Precision is the rate at which the samples classified into a given class are truly a true example of that class. Recall implies how many of the truly positive samples are detected by the model, while the F1-score gives an overall balanced measure between precision and recall as its harmonic mean. The well-known measures (Precision, Recall, and F-measure) can be calculated as follows:

$$\text{Precision} = \frac{True\ Positives}{True\ Positive + FalsePositive} \quad \frac{TP}{TP+FP} \quad \text{Eq2 [7]}$$

$$\text{Recall} = \frac{True\ Positives}{True\ Positive\ + FalseNegatives} \quad \frac{TP}{TP+FN} \quad \text{Eq3 [7]}$$

$$\text{F1} = \frac{2\times\ \text{Precision}\times\text{Recall}}{\text{Precision} + \text{Recall}} \quad Eq4\ [7]$$

Where:

**True Positive (TP):** Cases that are predicted as positive and they are actually positive.

**True Negative (TN):** Cases that are predicted as negative and they are actually negative.

**False Positive (FP):** Cases that are predicted as positive but they are negative.

**False Negative (FN):** Cases that are predicted as negative but, they are positive.



DOI:10.5281/zenodo.18108194

The figure 9, illustrates an excellent performance obtained by the proposed DDoS model for the two Classes. More precisely, the precision, recall, and F1-score values of 99.6%, 98.8%, and 99.2%, respectively, for the Benign class indicated a very high ability to identify benign samples with very few misclassifications. The model had also performed well in the Attack Class, showing 99.5% precision, 99.8% recall, and 99.6% F1-score, thus indicating the ability to detect almost all attack samples, with an extremely low rate of missed detections or misclassifications.

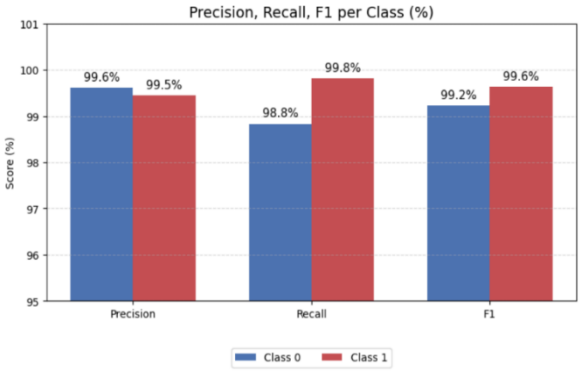



Figure: 8 The Classification metrics for poposed DDoS model

*6.5 Testing integration of the DDoS model*

The proposed model is also tested to evaluate the efficiency of the model when it is integrated with the system. The tests include simulating the HTTP, Slowloris, SQLi, XSS, SYN, UDP attacks. The collect performance data before and after integration are compared based throughput, average latency, CPU, and memory usage. **Figure 10** show the system performance before and after the integration, it clearly shows that the latency impact 200-300% increase due to model loading and prediction computation. While the throughput impact recorded 10-20% decrease. The Memory Overhead ranged from +200 to 250MB due to models and intermediate data. And the CPU Impact +25-40% compared to baseline. The results from the experiment show that the LSTM-based deep learning models are very powerful in recognizing the different types of web application attacks such as DDoS, which made them superior to traditional WAF systems in terms of detection accuracy. The models had a very good performance even under concurrent requests, thus showing their scalability and adaptability to different attack patterns. On the other hand, the deep learning integration had an impact on performance by increasing the detection latency, and usage of CPU and memory. Even so, the security enhancement and vast detection of threats made the overhead acceptable.

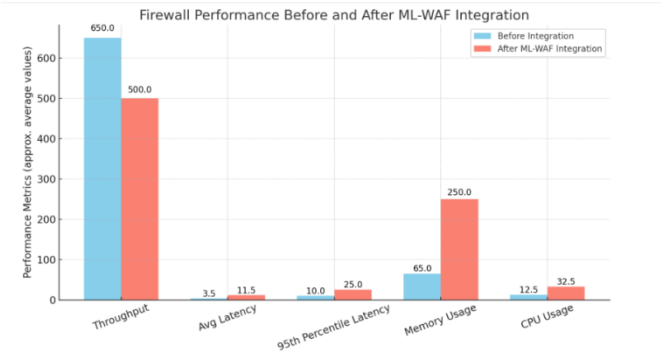



Figure: 9 a comparison of firewall performance before and after integrating the AI model (ML-WAF).

## 7.CONCLUSION

As web applications are essential for providing several online services such as e-banking, and different educational facilities whereby securing personal, individual or governmental information are very significant concern and critical issue due to web application's common attacks like DDoS and other attacks. This study aimed to enhance web applications' firewall based on integrating trained LSTM model. The experimental results provided evidences that the application of LSTM models led to a better enhancement in the detection capabilities of the WAF systems over the traditional methods, by achieving a high rate of success in the detection of DDoS attacks. The enhanced model combining the traditional WAF with an AI-based DDoS detection layer. Combining two systems introduce some computational difficulties with an increase in the response time, however the security benefits achieved by the proposed DDoS model were so great that they outweighed this burden. Additionally, the research offers an unambiguous structure for potential future uses that can aim at working on model efficiency, cutting down on computational overhead, and broadening detection options, all the while presenting a cost-effective solution against pricey commercial firewalls that might be out of reach for individual users and small enterprises. As a future work, it is highly suggested to try out various methods to reduce the computational cost of deep learning models, like model quantization, pruning, or the usage of specific hardware (like GPUs or TPUs), with the primary goal of increasing throughput and decreasing delay.



DOI:10.5281/zenodo.18108194

## 8.REFERENCES


[1] P. K. Bediako, "Long short-term memory recurrent neural network for detecting DDoS flooding attacks within TensorFlow implementation framework," 2017.

[2] B. R. Dawadi, B. Adhikari, and D. K. Srivastava, "Deep learning technique-enabled web application firewall for the detection of web attacks," *Sensors*, vol. 23, no. 4, p. 2073, 2023.

[3] S. Toprak and A. G. Yavuz, "Web application firewall based on anomaly detection using deep learning," *Acta Infologica*, vol. 6, no. 2, pp. 219–244, 2022.

[4] S. Rajesh *et al.*, "Real-time DDoS attack detection based on machine learning algorithms," in *Proc. Yukthi Conf.*, 2021.

[5] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 8th ed. London, UK: Pearson Education Limited, 2023.

[6] Verizon, *2025 Data Breach Investigations Report*. Verizon Enterprise Solutions, 2025.

[7] M. Ramzan *et al.*, "Distributed denial of service attack detection in network traffic using deep learning algorithm," *Sensors*, vol. 23, no. 20, p. 8642, 2023.

[8] E. Leka *et al.*, "Web application firewall for detecting and mitigation of DDoS attacks using machine learning and blockchain," *TEM Journal*, vol. 13, no. 4, 2024.

[9] C. Diekmann, L. Hupel, and G. Carle, "Semantics-preserving simplification of real-world firewall rule sets," in *Proc. Int. Symp. Formal Methods*, 2015, pp. 1–16, Springer.

[10] K. Goss and W. Jiang, "Distributing and obfuscating firewalls via oblivious Bloom filter evaluation," *arXiv preprint arXiv:1810.01571*, 2018.

[11] V. A. Krishna and T. A. A. Victoire, "Analysis of firewall policy rules: A comparative study," *International Journal of Computer Applications*, vol. 6, no. 5, pp. 112–118, 2013.

[12] A.Zhaikhan *et al.*, "Safeguarding the IoT from malware epidemics: A percolation theory approach," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 6039–6052, 2020.

[13] Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] Y. Ming *et al.*, "Understanding hidden memories of recurrent neural networks," in *Proc. IEEE Conf. Visual Analytics Science and Technology (VAST)*, 2017, pp. 13–24.

[16] J. Á. Román-Gallego *et al.*, "Artificial intelligence web application firewall for advanced detection of web injection attacks," *Expert Systems*, vol. 42, no. 1, p. e13505, 2025.

[17] A.Moradi Vartouni, M. Teshnehlab, and S. S. Kashi, "Leveraging deep neural networks for anomaly-based web application firewall," *IET Information Security*, vol. 13, no. 4, pp. 352–361, 2019.

[18] Shaheed and M. B. Kurdy, "Web application firewall using machine learning and features engineering," *Security and Communication Networks*, vol. 2022, p. 5280158, 2022.

[19] T. Islam, M. I. Jabiullah, and D. M. H. Abid, "DDoS attack preventing and detection with the artificial intelligence approach," in *Proc. Int. Symp. Intelligent Computing Systems*, 2022, Springer.




DOI:10.5281/zenodo.18108194