JPPIPA 10(12) (2024)

Jurnal Penelitian Pendidikan IPA

Journal of Research in Science Education

http://jppipa.unram.ac.id/index.php/jppipa/index

Design of Realtime Web Application Firewall on Deep Learning-Based to Improve Web Application Security

Rofif Zainul Muttaqin[1*], Dodi Sudiana[1]

[1] Department of Electrical Engineering, University of Indonesia.

Received: July 02, 2024
Revised: October 09, 2024
Accepted: December 25, 2024
Published: December 31, 2024

Corresponding Author:
Rofif Zainul Muttaqin
rofif.zainul@ui.ac.id

DOI: 10.29303/jppipa.v10i12.8346

Abstract: Web applications are widely used nowadays, but comprises several vulnerabilities that are often used by attacker to exploit the system. There is web application firewall (WAF) that could mitigate these problem. WAF generally works based on pre-established rules. However, the weakness of this system is the evolving nature of attacks, and configuring rules on WAF requires in-depth knowledge related to existing applications. Artificial intelligence technology, both machine learning (ML) and deep learning (DL), shows good potential in recognizing types of attacks. In this research, a Real-time DL-based WAF was built to enhance security in web applications. Various ML and DL models were tested to perform the task of web attack detection, including Support Vector Machine (SVM), Random Forest (RF), Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM). Based on the test results, the CNN-LSTM model achieved the highest performance, namely an accuracy of 98.61%, precision of 99%, recall of 98.08%, and f1-score of 98.54%. From the testing results with a web vulnerability scanner, the performance of the DL-based WAF is not inferior to ModSecurity WAF, which is used as a comparison. From the analysis results, it can be concluded that the implementation of DL-based WAF can improve the security of web applications.

Keywords: Cybersecurity; Deep learning; Web application firewall (WAF); Web attack detection; Web vulnerabilities

Introduction

Web applications are widely used in various sectors, ranging from use for social media, e-commerce, news portals, multimedia streaming platforms, and others (Altulaihan et al., 2023; Aydos et al., 2022). This web application is generally an easy target to attack as it can be accessed via the internet network and has various vulnerabilities (Eunaicy et al., 2022). Attackers usually exploit existing vulnerabilities in web systems to carry out attacks (Altulaihan et al., 2023). When a user visits a web page, the web server delivers the requested content. Web servers are responsible for tasks like storing, delivering, and rendering web pages for clients. HTTP serves as the communication protocol for this interaction (Kresna et al., 2018).

Attackers, typically exploit web systems via HTTP protocol (Mac et al., 2018). As with many protocol, HTTP and HTTPS protocol also have vulnerabilities. Attacker exploit these vulnerabilities and perform attacks such as Man in the Middle (MITM), Brute Force, Distributed Denial of Service (DDoS), SQL injection (SQLi), and Cross-site Scripting (XSS) (Luxemburk et al., 2021).Increasing the security of a web application is very necessary to overcome potential attacks. This problem can be resolved using various methods, including the use of Web Application Firewall (WAF). WAF has ability to filter, block and logging the HTTP traffic (Muzaki et al., 2020). WAF would filter incoming requests, and reject requests that are suspicious or that are classified as attacks (Applebaum et al., 2021; Dawadi et al., 2023; Khamdamov et al., 2019).

How to Cite:
Muttaqin, R. Z., & Sudiana, D. (2024). Design of Realtime Web Application Firewall on Deep Learning-Based to Improve Web Application Security. *Jurnal Penelitian Pendidikan IPA*, *10*(12), 11121–11129. https://doi.org/10.29303/jppipa.v10i12.8346

WAF works based on predetermined rules. Its configuration requires someone who has technical knowledge of the web application in it. Generally, WAF has several web applications in it, so the configuration applied must be adjusted to the function and needs of each application in it (Clincy et al., 2018). Configuring rules in WAF sometimes takes longer time and carries the risk of errors because it is done manually for each web application. This technique is quite successful in dealing with existing attacks but faces difficulties when attacks become more complex and sophisticated. The attacks continue to develop and evolve.

Machine Learning (ML) is a tool that has the potential to be used in the field of cyber security, as in this case, machine learning can be used to detect and prevent attacks on web applications (Paleyes et al., 2022; Rolnick et al., 2023; Roscher et al., 2020; Topuz et al., 2023). Machine learning can be used to recognize patterns, detect anomalies, identify suspicious activity, and predict potential attacks. The use of machine learning allows the detection process to be carried out in real-time and adaptively. The machine learning algorithm can be classify into supervised learning, unsupervised learning, reinforcement learning, and semi-supervised learning. Supervised learning works by classifying data based on the data labels that have been provided (Verbraeken et al., 2020). The data is first given a label/class, which is then used for the learning process. This algorithm is widely used for classification tasks, such as classifying attacks on IoT (Internet of Things) networks in research (Ioannou et al., 2019, 2020; Krishnan et al., 2021; Rani et al., 2020).

Traditional machine learning techniques, including decision trees, support vector machines, and clustering algorithms, have demonstrated promising results in detecting known patterns (Ullah et al., 2020). However, these approaches often face challenges in addressing the growing complexity and sophistication of cyber security threats. Additionally, they are less effective when dealing with large-scale, high-dimensional, and imbalanced datasets, which are prevalent in cyber security scenarios (Li et al., 2022). Traditional ML models require human intervention to obtain optimal results. This models can work well on small datasets, while deep learning models show quite good performance with fairly large datasets (Feng et al., 2019; Purushotham et al., 2018). However, if the data is insufficient or not well distributed, bias will arise in the model. Therefore, for better performance, data samples for training need to be distributed well and sufficiently (Shaukat et al., 2020).

Deep learning (DL) is a part of machine learning. This is inspired by the workings of the human brain which has the ability to think logically and analytically. Deep learning techniques, renowned for their exceptional performance in fields like image recognition, natural language processing, and speech recognition, present promising solutions for enhancing cybersecurity (Yin et al., 2021). Methods such as CNNs, RNNs, and transformer models can autonomously extract complex patterns and representations from raw data (Ullah et al., 2022). This ability allows deep learning models to identify novel and advanced attacks that might bypass traditional machine learning approaches (Vaswani et al., 2017). Deep learning techniques have potential to significantly improve the detection and prevention of web attacks. Research by Salam et al, shows that deep learning approach such as CNN, RNN and transformer model can effectively detect web-based attack with an overall high performance across all models (Salam et al., 2023).

Several studies have been conducted regarding the use of machine learning and deep learning for web attack detection (Alaoui et al., 2022) and get quite good results based on testing with the dataset that has been prepared. In the problem of web attack detection, the data used is text data where there are several approaches taken to perform feature selection starting from character embedding (Demirel et al., 2023), statistical feature (Althubiti et al., 2018), and word-embedding (Jemal et al., 2021). On research conducted by (Jemal et al., 2021) word embedding gets better results compared to when using character embedding. Several algorithms that are widely used for detecting attacks on the web include SVM (Support Vector Machine), RF (Random Forest), CNN (Convolutional Neural Network), and LSTM (Long Short-Term Memory) (Alaoui et al., 2022).

However, several studies have not discussed much about implementation and testing of systems directly, only limited to testing datasets, both general/public datasets and private datasets. In research by (Aswal et al. (2021), testing was also carried out by integrating a machine learning model as a classifier and providing triggers to the rule-based WAF so that it can update rules if a request is detected as an attack based on the model that has been constructed. However, the research is still limited to SQL injection detection and the process of updating rules on the firewall still not effective enough because it requires time and a restart process on the firewall.

Based on these several problems, this research carried out the design of real-time web application firewall based on deep learning. The deep learning model is used to detect whether the request is malicious or not, and triggers the web application firewall to block or allow the request. The model is constructed using the latest http attack dataset that contains several type of attack based on OWASP Top 10 2021 (Ramezany et al., 2022), FWAF dataset that contains malicious and normal http request (Seyyar et al., 2022), and http attack data

that collected from internal webserver. Some of the algorithms used in this research are SVM, RF, CNN, and LSTM. In the research by Seyyar et al., the use of hybrid approach shows good performance for classifying normal and malicious request (Seyyar et al., 2022). Seyyar et al., use BERT (Bidirectional Encoder Representations from Transformers) model and Multi Layer Perceptron (MLP). BERT is used to obtain the feature and MLP to classify the input. In the research by Kim et al, CNN-LSTM model shows good performance in detection http payload attacks by using combination of CNN and LSTM model. The model could distinguished between normal traffic and abnormal traffic that could not be detected by signature-based network intrusion detection system (NIDS) (Kim et al., 2020). Inspired by these research, a hybrid model approach also be carried out in this research by combining one algorithm with other algorithms, such as CNN-SVM, CNN-RF, and CNN-LSTM to explore more about the impact to the model performance. CNN is used to extract existing features which will then be used by the classifier to determine whether the data is classified as a normal payload or a malicious payload.

The best performance model then be integrated with the WAF that also being constructed in this research. In this research, testing and evaluation are conducted by evaluating the model using a testing dataset and comparing the performance of the deep learning-based web application firewall (DL-WAF) with the existing rule-based WAF in handling various web attacks. The web attacks were carried out by simulating an attack with a web vulnerability scanner tool on vulnerable web application to evaluate the performance of WAF. The application to be tested is DVWA (Damn Vulnerable Web Application) which was protected with a deep learning-based web application firewall (DL-WAF). DVWA is an application to have various vulnerabilities in it. DVWA contains top OWASP vulnerabilities such as SQL injection, XSS (cross-site scripting), file inclusion, CSRF and etc (Priyanka et al., 2020). Based on the description of this background, this research aims to create real-time web application firewall design by utilizing deep learning models to improve web application security.

Method

The research method includes the methodology used in the study and the design of the system being developed. The processing steps in http anomalies detection could be grouped into two categories: preprocessing/feature generation and detection/classification (Díaz-Verdejo et al., 2023). Preprocessing comprises following operations: extract, parse, parametrization, and preprocessing features. Classification group comprises operations such as model evaluation and classification/detection. Figure 1 illustrates the model development process, which involves several steps such as dataset collection and setup, dataset preprocessing, tokenization, word embedding, model training and model evaluation/testing.

First, the http attacks data collected from internal webserver using security incident and event management (SIEM). These data then being integrated with the http payload datasets from public dataset that contains normal request and malicious request (Ramezany et al., 2022; Seyyar et al., 2022).

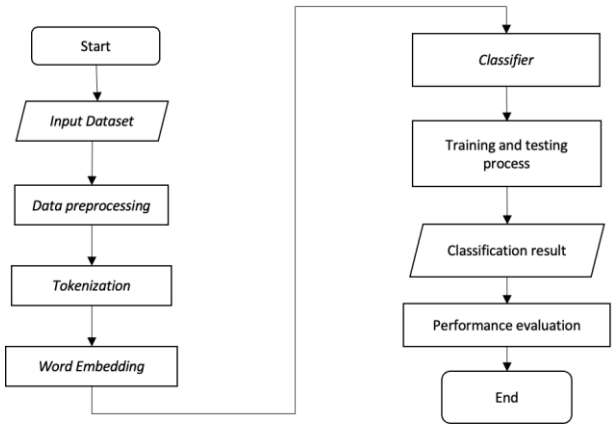

Figure 1. Deep learning model development process

The next process was loading a dataset consisting of http requests along with a label indicating whether the request is categorized as a normal request or a malicious request. The data then passed through the preprocessing stage, namely the process of removing duplicate data, the data cleaning and splitting process based on special characters, the tokenization process, as well as the formation of an embedding matrix. In this research, several features were obtained through the word embedding process, where feature information was extracted from the data by analyzing the relationships between words. The embeddings, which contain semantic relationships and contextual information, were leveraged for tasks such as sentiment analysis and text classification. The FastText library was used to construct the word embeddings. FastText can represent information from a sub-word and overcome the out of vocabulary problem (Hashmi et al., 2024).

At the preprocessing stage, the http payload data passed through a filtering process to form a word embedding, namely the special characters in the data that were removed first. Then, the process of forming a word dictionary was carried out based on the words in the dataset. Each word was encoded into an index, where this process named a tokenization process. After

each word becomes an index, a word embedding was formed as a weight matrix for each word. The data were then shared for training and testing purposes. The proportion of training and testing used show 80% for training and 20% for testing. The next process was the training process using the proposed model and continued with the testing process. Then, several indicators were measured to serve as indicators of model performance. These indicators included accuracy, recall, precision, and f1-score. The experiment was conducted repeatedly with several configurations to achieve the best results. In this research, in addition to developing a model to detect web attacks, a Web Application Firewall (WAF) was created using a deep learning model designed to classify HTTP requests and detect indications of an attack. The DL-based WAF that was constructed is a web application that has a reverse proxy function where this application can forward incoming requests to the target server or web application. This application was constructed with Flask, as a framework for the Python programming language that can be used to build web applications.

DL-based WAF (DL-WAF) works by filtering requests from outside (the internet) to the target application server. When there is a request from outside to the target web application server, the request will go through the DL-based WAF first. The DL-based WAF will detect whether the request is normal or malicious with the help of a deep learning model. If the request is normal, the request will be forwarded to the target application server, whereas if the request is indicated as an attack, the DL-based WAF will not forward the request to the target server. DL-based WAF will block the request. Furthermore, the developed DL-based WAF is also equipped with a notification feature to notify web application server administrator if there are indications of an attack. Then, notifications are sent to Telegram Bots. Besides evaluation using testing dataset, the performance evaluation also carried out using simulated attacks with web vulnerability scanner.

Figure 2. Performance evaluation using web vulnerabilities scanner

Figure 2 shows the performance evaluation scenario using web vulnerabilities scanner. The performance of the DL-based WAF would be compared with the existing WAF, which is ModSecurity. ModSecurity is opensource web application firewall (WAF) module. ModSecurity works as a security module in web server that has several rules that can be configured to detect and prevent attacks on web applications (Muzaki et al., 2020). Web vulnerability scanner is a tools that often used by software developer and cybersecurity expert to do testing on web application to seeks the application vulnerabilities (Alazmi et al., 2022). Acunetix is being used in this research to simulate several web attack. Acunetix is web vulnerability scanner. The type of the attack that being simulated could be configured based on scanning profile in Acunetix (*Configuring Scan Profiles | Acunetix*, n.d.). In this research, the critical/high risk profile was chosen because it would checked the vulnerabilities that has the highest impact on web security.

Result and Discussion

The total of data before going through the preprocessing process were 53074 normal requests, while 87429 for malicious requests. The data was then divided into 80% for training and 20% for testing. The data then goes through the preprocessing stage first, namely the process of removing duplicate data, the text cleaning, and splitting process. Data that originally contained special characters needs to be processed first so that it can be processed to form word embedding. Special characters and number are removed. After that, the data will become a collection of words that can be processed to form a word embedding matrix. To create word embeddings, we use the FastText library to build a word embedding matrix. The final amount of data after deleting duplicate data is 44920 for normal requests and 40543 for malicious requests.

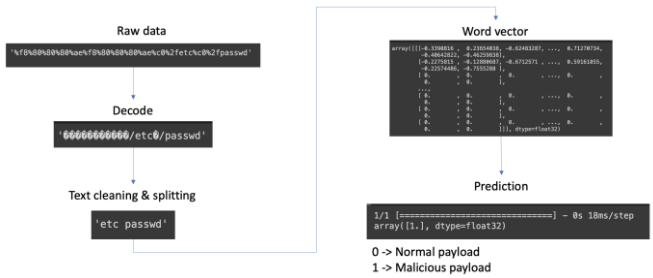

Figure 3. Web attack classification process

Figure 3 shows the classification process being done in these research. First, the http request data was decoded and going into text cleaning and splitting process. After that, each word was encoded into an index during tokenization process and word vector is being form using word embedding method to extracted the information from the data. Once the data is in the form of an embedding matrix, the data will then be

processed by deep learning models to classifiy whether the request is normal or malicious.

Test Results Using Testing Dataset

Testing was carried out using several commonly used algorithms, namely SVM, RF, CNN, and LSTM. In this research, a hybrid approach was also tested by combining two existing algorithms, using CNN as a feature extractor and another algorithm as a classifier. Several indicators used to assess the performance of the model include accuracy, precision, recall, and f1-score. Table 1 shows the hyperparameter configuration used for each model in this research.

Table 1. Hyperparameter Configuration Used in the Research

Model	Hyperparameter
SVM	C=1000, kernel=rbf
RF	n_estimators=1000,max_features=auto
CNN	filter=128, kernel_size=2,pool_size=2
LSTM	units=64
CNN-SVM	CNN (filter=128, kernel_size=2,pool_size=2), SVM (C=1000, kernel=rbf)
CNN-RF	CNN (filter=128, kernel_size=2,pool_size=2), RF (n_estimators=1000,max_features=auto)
CNN-LSTM	CNN (filter=128, kernel_size=2,pool_size=2),LSTM(units=64)

Following are the test results with the testing dataset:

Table 2. Test Results Using the Testing Dataset

Methods	Accuracy	Precision	Recall	F1 score
SVM	0.9717	0.9798	0.9608	0.9702
RF	0.9638	0.9760	0.9477	0.9616
CNN	0.9803	0.9837	0.9749	0.9793
LSTM	0.9849	0.9891	0.9792	0.9841
CNN-SVM	0.9799	0.9891	0.9792	0.9841
CNN-RF	0.9808	0.9831	0.9748	0.9789
CNN-LSTM	0.9861	0.9900	0.9808	0.9854

Table 2 shows the simulation results using the testing dataset with several models. The best results are achieved by using the CNN-LSTM model.The CNN-LSTM model achieved an accuracy of 98.61%, precision of 99%, recall of 98.08% and f1 score of 98.54%. Figure 4 shows that CNN-LSTM model has been trained successfully and has achieved high accuracy on both training and validation. In the early epochs, both training and validation accuracy increase rapidly, indicating that the model is learning effectively from the data. The CNN-LSTM reached the convergence phase after 10 epochs. The accuracy for both training and validation datasets remains stable and high for the remaining epochs. This is a sign that the model is not overfitting, as there is no significant gap between training and validation accuracy.

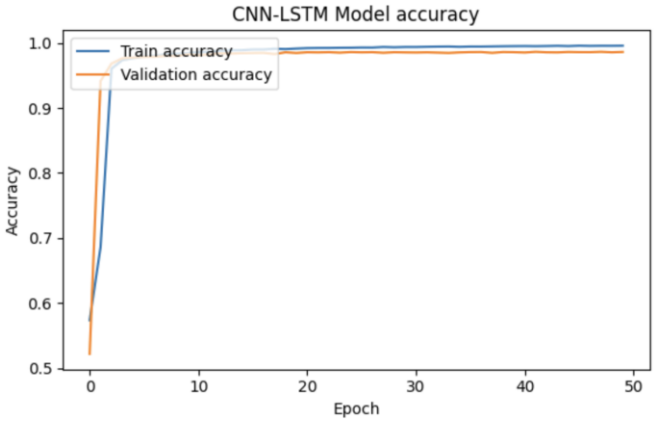

Figure 4. CNN-LSTM model accuracy graph

Figure 5 shows the confusion matrix of the CNN-LSTM classification result on the testing dataset. It shows that the model could correctly predict 8833 normal payloads and 8022 malicious payloads. The number of false positives from the prediction that was made by the model is 81 and the number of false negatives is 157. It is relatively low compared to the number of true positives and true negatives, indicating that model could distinguish between a normal and malicious payload. The results found that the hybrid approach using CNN-LSTM successfully improved the model's performance and could effectively classify the HTTP request as a normal or malicious request.

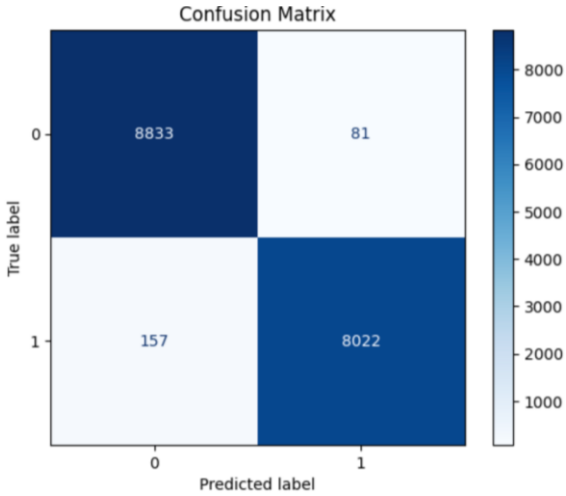

Figure 5. Confusion matrix for CNN-LSTM model

DL-*based WAF Test Results with Web Vulnerability Scanner*

In this research, testing was also carried out using model implementation scenarios on the server for web-attack detection. A web application was constructed that acts as a reverse proxy for the web application that is the target of the attack. This reverse proxy application works like a Web Application Firewall (WAF). The model with the best performance in this research was saved and embedded in the reverse proxy application that was constructed. This model will later be tasked with

classifying requests, including normal requests or malicious requests. The performance of the deep learning-based WAF was tested by comparing it with the commonly used rule-based WAF, namely ModSecurity. The testing was conducted by simulating attacks on an application designed to have various vulnerabilities. DVWA (Damn Vulnerable Web Application) was used to ensure the application had many vulnerabilities. The attack simulation was performed using web vulnerability scanner tools, which were employed to simulate various types of attacks on the web.

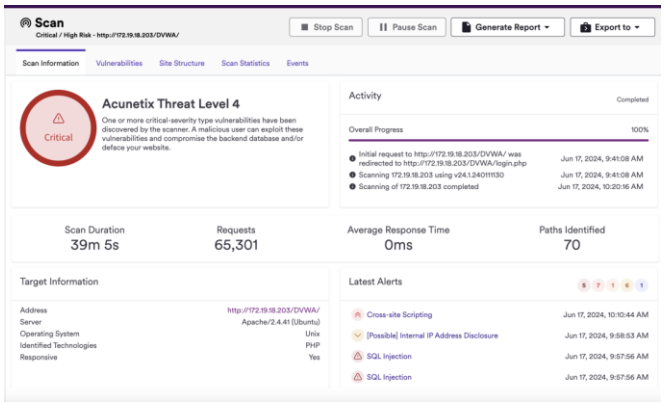

Figure 6. DVWA test results

First, the test was carried out by scanning the DVWA application without WAF protection. Figure 6 shows the scanning results for DVWA application without WAF protection. It is found that there are five vulnerabilities with a critical level, seven vulnerabilities with a high level, one vulnerability with a medium level, and six vulnerabilities with a low level. Vulnerabilities with a critical level include vulnerabilities to local file inclusion (LFI) attacks and SQL injection, while vulnerabilities with a high level are vulnerabilities to XSS. Then the next test was to carry out scanning process on the DVWA application by activating rule-based WAF using ModSecurity.

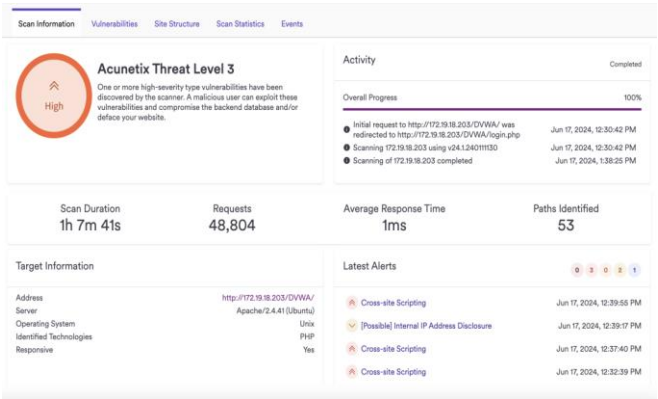

Figure 7. DVWA test results with ModSecurity WAF protection

Figure 7 shows the scanning result on DVWA application with ModSecurity WAF protection. It is found that there are three vulnerabilities with a high level and two with a low level. The highest level of vulnerability found is XSS. From these results, it can be seen that we can improve security in web applications by implementing WAF. Next, tests were conducted on the DL-based WAF to compare it with ModSecurity WAF.

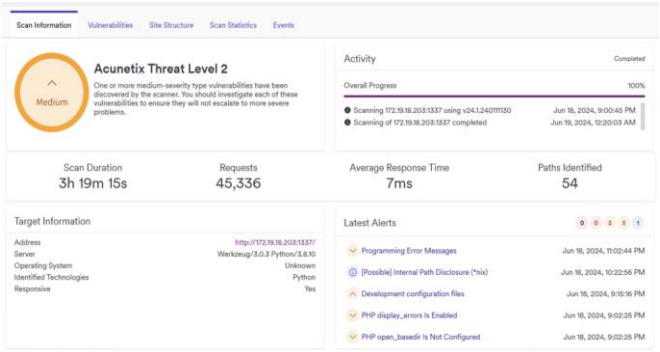

Figure 8. DVWA test results with ModSecurity WAF protection

Figure 8 shows the test result for DVWA application with DL-based WAF protection. From the test results, it is found that there are two vulnerabilities with a medium level and five with a low level. From the test results on DL-based WAF, the DVWA application does not have critical or high level vulnerabilities. The only vulnerabilities found were vulnerabilities at the medium and low levels. If checked in more detail, the medium level vulnerabilities that exist include the detection of the phpinfo page and development file configuration.

From the test results, it can be seen that the DL-based WAF that was constructed can improve security in web applications. Several critical and high-level vulnerabilities that existed previously were not found in the web applications tested after implementing DL-based WAF. The performance of the DL-based WAF is pretty good and show great potential compared to the rule-based ModSecurity WAF. While high-level vulnerabilities were still detected with the ModSecurity WAF, these vulnerabilities were not present when using the DL-based WAF.

Test Results of Logging and Notification Features on DL-based WAF

The DL-based WAF that was constructed is also equipped with a logging feature to record requests identified as attacks. The log will record the attacker's IP address and also the http payload used. This data can be used for further analysis as well as developing and

increasing model capacity. The figure 9 shows an example of a log from a DL-based WAF:

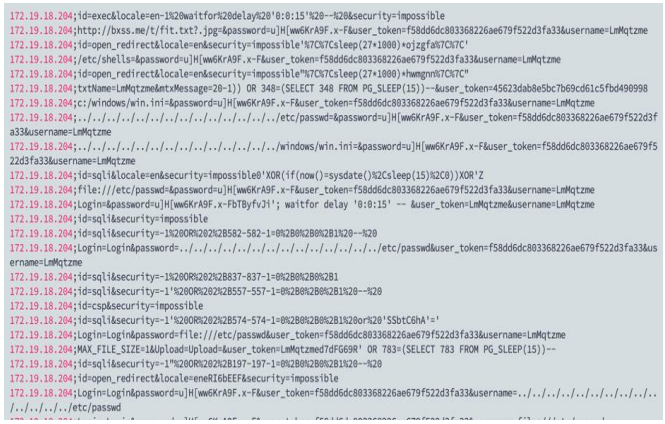

Figure 9. DL-based WAF log

The constructed DL-based WAF is also equipped with a notification feature so that application or system administrator know if there are indications of an attack on their web application or system. In this research, notification was carried out by sending messages to Telegram automatically such as shown in figure 10.

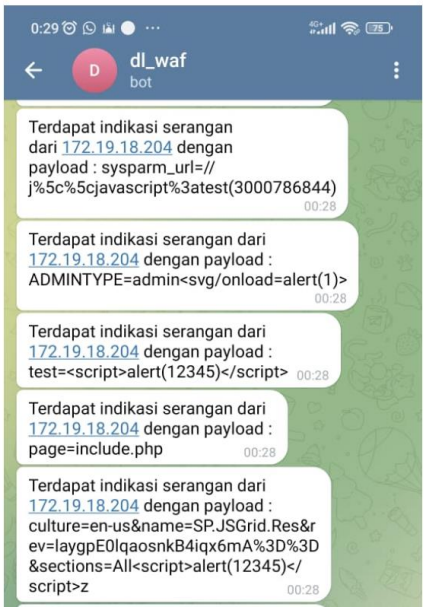

Figure 10. The DL-based WAF notification on telegram

Conclusion

The use of deep learning models to detect web attacks via http payloads shows quite good results, where the CNN-LSTM model achieving the best results compared to several other models in this research. The CNN-LSTM model achieved the best results, namely 98.61% accuracy, 99% precision, 98.08% recall, and f1-score of 98.54%. A hybrid model approach using CNN-LSTM successfully improved the performance of the model. The constructed DL-based WAF is quite effective in dealing with attacks based on test results using a web vulnerability scanner. The performance of the DL-based WAF is comparable to the widely used rule-based WAF, specifically ModSecurity WAF. In this research, the proposed solution is evaluated not only using a testing dataset but also through simulated HTTP attacks to demonstrate real-world incidents and further assess the solution's performance. Based on the research results, implementing a DL-based WAF can enhance web application security by protecting against existing web attacks. The result shows that there is no critical and high vulnerabilities found in the web application that being tested after applying the DL-based WAF. The critical and high vulnerabilities were effectively mitigated by the DL-based WAF.

Acknowledgments

The authors would like to thanks to University for give occasion for this research

Author Contributions

Investigation, R.Z.M and D.S; formal analysis, R.Z.M and D.S; investigation R.Z.M and D.S; resources, R.Z.M and D.S; data curation, R.Z.M and D.S: writing—original draft preparation, R.Z.M and D.S; writing—review and editing, R.Z.M and D.S: visualization, S R.Z.M and D.S; supervision, R.Z.M and D.S; project administration, R.Z.M and D.S; funding acquisition, R.Z.M and D.S. All authors have read and agreed to the published version of the manuscript.

Funding

This research is fully supported by the author's funds without any external funding sources

Conflicts of Interest

We certify that there is no conflict of interest with any financial, personal and other relationships with other peoples or organization related to the material discussed in the manuscript.

References

Alaoui, R. L., & Nfaoui, E. H. (2022). Deep Learning for Vulnerability and Attack Detection on Web Applications: A Systematic Literature Review. *Future Internet*, *14*(4), 118. https://doi.org/10.3390/fi14040118

Alazmi, S., & De Leon, D. C. (2022). A Systematic Literature Review on the Characteristics and Effectiveness of Web Application Vulnerability Scanners. *IEEE Access*, *10*, 33200–33219. https://doi.org/10.1109/ACCESS.2022.3161522

Althubiti, S., Nick, W., Mason, J., Yuan, X., & Esterline, A. (2018). Applying Long Short-Term Memory Recurrent Neural Network for Intrusion Detection. *Conference Proceedings - IEEE SOUTHEASTCON*.

https://doi.org/10.1109/SECON.2018.8478898

Altulaihan, E. A., Alismail, A., & Frikha, M. (2023). A Survey on Web Application Penetration Testing. In *Electronics (Switzerland)* (Vol. 12, Issue 5). https://doi.org/10.3390/electronics12051229

Applebaum, S., Gaber, T., & Ahmed, A. (2021). Signature-based and Machine-Learning-based Web Application Firewalls: A Short Survey. *Procedia CIRP*, *189*. https://doi.org/10.1016/j.procs.2021.05.105

Aswal, K., Rajmohan, A., Mukund, S., Panicker, V. J., & Dhivvya, J. P. (2021). Kavach: A Machine Learning based approach for enhancing the attack detection capability of firewalls. *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 1–5. https://doi.org/10.1109/ICCCNT51525.2021.9579836

Aydos, M., Aldan, Ç., Coşkun, E., & Soydan, A. (2022). Security testing of web applications: A systematic mapping of the literature. In *Journal of King Saud University - Computer and Information Sciences* (Vol. 34, Issue 9). https://doi.org/10.1016/j.jksuci.2021.09.018

Clincy, V., & Shahriar, H. (2018). Web Application Firewall: Network Security Models and Configuration. *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, *01*, 835–836. https://doi.org/10.1109/COMPSAC.2018.00144

Dawadi, B. R., Adhikari, B., & Srivastava, D. K. (2023). Deep Learning Technique-Enabled Web Application Firewall for the Detection of Web Attacks. *Sensors, 23*(4), 2073. https://doi.org/10.3390/S23042073

Demirel, D. Y., & Sandikkaya, M. T. (2023). Web Based Anomaly Detection Using Zero-Shot Learning With CNN. *IEEE Access*, *11*, 91511–91525. https://doi.org/10.1109/ACCESS.2023.3303845

Díaz-Verdejo, J. E., Estepa Alonso, R., Estepa Alonso, A., & Madinabeitia, G. (2023). A critical review of the techniques used for anomaly detection of HTTP-based attacks: taxonomy, limitations and open challenges. *Computers & Security*, *124*, 102997. https://doi.org/10.1016/j.cose.2022.102997

Eunaicy, C., & Suguna, S. (2022). Web attack detection using deep learning models. *Materials Today: Proceedings*, *62*. https://doi.org/10.1016/j.matpr.2022.03.348

Hashmi, E., Yayilgan, S. Y., Yamin, M. M., Ali, S., & Abomhara, M. (2024). Advancing Fake News Detection: Hybrid Deep Learning With FastText and Explainable AI. *IEEE Access*, *12*(March), 44462–44480. https://doi.org/10.1109/ACCESS.2024.3381038

Ioannou, C., & Vassiliou, V. (2019). Classifying Security Attacks in IoT Networks Using Supervised Learning. *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 652–658. https://doi.org/10.1109/DCOSS.2019.00118

Ioannou, C., & Vassiliou, V. (2020). Experimentation with Local Intrusion Detection in IoT Networks Using Supervised Learning. *2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 423–428. https://doi.org/10.1109/DCOSS49796.2020.00073

Jemal, I., Haddar, M. A., Cheikhrouhou, O., & Mahfoudhi, A. (2021). Performance evaluation of Convolutional Neural Network for web security. *Computer Communications*, *175*, 58–67. https://doi.org/10.1016/j.comcom.2021.04.029

Khamdamov, R. K., Kerimov, K. F., & Ibrahimov, J. O. (2019). Method of developing a web-application firewall. *Journal of Automation and Information Sciences*, *51*(6). https://doi.org/10.1615/JAutomatInfScien.v51.i6.60

Kim, A., Park, M., & Lee, D. H. (2020). AI-IDS: Application of Deep Learning to Real-Time Web Intrusion Detection. *IEEE Access*, *8*, 70245–70261. https://doi.org/10.1109/ACCESS.2020.2986882

Kresna, A. I., & Rosmansyah, Y. (2018). Web Server Farm Design Using Personal Computer (PC) Desktop. *International Conferences on Information Technologies and Electrical Engineering*, 106–111. https://doi.org/10.1109/ICITEED.2018.8534920

Krishnan, S., Neyaz, A., & Liu, Q. (2021). IoT Network Attack Detection using Supervised Machine Learning. *International Journal of Artificial Intelligence and Expert Systems (IJAE), 10*, 18–32. Retrieved from https://www.cscjournals.org/manuscript/Journals/IJAE/Volume10/Issue2/IJAE-201.pdf

Li, Z., Zou, D., Xu, S., Jin, H., Zhu, Y., & Chen, Z. (2022). SySeVR: A Framework for Using Deep Learning to Detect Software Vulnerabilities. *IEEE Transactions on Dependable and Secure Computing*, *19*(4), 2244–2258. https://doi.org/10.1109/TDSC.2021.3051525

Luxemburk, J., Hynek, K., & Cejka, T. (2021). Detection of HTTPS Brute-Force Attacks with Packet-Level Feature Set. *2021 IEEE 11th Annual Computing and Communication Workshop and Conference, CCWC 2021*, 114–122. https://doi.org/10.1109/CCWC51732.2021.9375998

Mac, H., Truong, D., Nguyen, L., Nguyen, H., Tran, H. A., & Tran, D. (2018). Detecting Attacks on Web Applications using Autoencoder. *Symposium on*

Information and Communication Technology, 416–421. https://doi.org/10.1145/3287921.3287946

Muzaki, R. A., Briliyant, O. C., Hasditama, M. A., & Ritchi, H. (2020). Improving Security of Web-Based Application Using ModSecurity and Reverse Proxy in Web Application Firewall. *2020 International Workshop on Big Data and Information Security, IWBIS 2020*, 85–90. https://doi.org/10.1109/IWBIS50925.2020.9255601

Paleyes, A., Urma, R. G., & Lawrence, N. D. (2022). Challenges in Deploying Machine Learning: A Survey of Case Studies. *ACM Computing Surveys*, *55*(6). https://doi.org/10.1145/3533378

Priyanka, A. K., & Smruthi, S. S. (2020). WebApplication Vulnerabilities:Exploitation and Prevention. *Proceedings of the 2nd International Conference on Inventive Research in Computing Applications, ICIRCA 2020*. https://doi.org/10.1109/ICIRCA48905.2020.9182928

Ramezany, S., Setthawong, R., & Tanprasert, T. (2022). A Machine Learning-based Malicious Payload Detection and Classification Framework for New Web Attacks. *19th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON 2022*. https://doi.org/10.1109/ECTI-CON54298.2022.9795455

Rani, D., & Kaushal, N. C. (2020). Supervised Machine Learning Based Network Intrusion Detection System for Internet of Things. *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 1–7. https://doi.org/10.1109/ICCCNT49239.2020.9225340

Rolnick, D., Donti, P. L., Kaack, L. H., Kochanski, K., Lacoste, A., Sankaran, K., Ross, A. S., Milojevic-Dupont, N., Jaques, N., Waldman-Brown, A., Luccioni, A. S., Maharaj, T., Sherwin, E. D., Mukkavilli, S. K., Kording, K. P., Gomes, C. P., Ng, A. Y., Hassabis, D., Platt, J. C., … Bengio, Y. (2023). Tackling Climate Change with Machine Learning. In *ACM Computing Surveys* (Vol. 55, Issue 2). https://doi.org/10.1145/3485128

Roscher, R., Bohn, B., Duarte, M. F., & Garcke, J. (2020). Explainable Machine Learning for Scientific Insights and Discoveries. *IEEE Access*, *8*. https://doi.org/10.1109/ACCESS.2020.2976199

Salam, A., Ullah, F., Amin, F., & Abrar, M. (2023). Deep Learning Techniques for Web-Based Attack Detection in Industry 5.0: A Novel Approach. *Technologies*, *11*(4), 107. https://doi.org/10.3390/technologies11040107

Seyyar, Y. E., Yavuz, A. G., & Ünver, H. M. (2022). An Attack Detection Framework Based on BERT and Deep Learning. *IEEE Access*, *10*, 68633–68644. https://doi.org/10.1109/ACCESS.2022.3185748

Topuz, K., Bajaj, A., & Abdulrashid, I. (2023). Interpretable Machine Learning. *Proceedings of the Annual Hawaii International Conference on System Sciences*, *2023-Janua*. https://doi.org/10.1201/9780367816377-16

Ullah, F., Javaid, Q., Salam, A., Ahmad, M., Sarwar, N., Shah, D., & Abrar, M. (2020). Modified Decision Tree Technique for Ransomware Detection at Runtime through API Calls. *Scientific Programming*, *2020*(1), 8845833. https://doi.org/10.1155/2020/8845833

Ullah, F., Salam, A., Abrar, M., Ahmad, M., Ullah, F., Khan, A., Alharbi, A., & Alosaimi, W. (2022). Machine health surveillance system by using deep learning sparse autoencoder. *Soft Computing*, *26*(16), 7737–7750. https://doi.org/10.1007/S00500-022-06755-Z/METRICS

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, 5999–6009. Retrieved from https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., & Rellermeyer, J. S. (2020). A Survey on Distributed Machine Learning. In *ACM Computing Surveys* (Vol. 53, Issue 2). https://doi.org/10.1145/3377454

Yin, X., Zhu, Y., & Hu, J. (2021). A Comprehensive Survey of Privacy-preserving Federated Learning. *ACM Computing Surveys (CSUR)*, *54*(6). https://doi.org/10.1145/3460427