2022 International Conference on Computers, Information Processing and Advanced Education (CIPAE) | 978-1-6654-6812-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/CIPAE55637.2022.00101

Design of Web Application Firewall System through Convolutional Neural Network and Deep Learning

Shihao Wang*, Ruiheng Liu, Xu Guo, Gaoda Wei
School of Software, Zhengzhou University, Zhengzhou, China
*Corresponding author: gypsoilpha@stu.zzu.edu.cn

***Abstract*—Attacks against web applications emerge in endlessly, and protecting the security of web applications is an important part of the current cyberspace security. This paper constructs one cloud web application firewall system based on deep learning. The system architecture adopts a multi-server distributed architecture. Taking into account the real-time nature of web requests and the user's sense of user experience, through the use of a high-performance server framework Openers and high-efficiency volumes. The convolutional neural network improves the operating efficiency of the system while ensuring the recognition rate. At the same time, the system implemented in this paper also saves server request information in the database to further provide targeted data sets for artificial intelligence training.**

Keywords*—*Web Application Firewall, Cyber Security, Deep learning

I. Introduction

In our daily life, we can find web applications all around us. Shopping, payment, travel and many other aspects are inseparable from Web applications, which are directly related to people's quality of life. Meanwhile, with the widespread use of web applications, there are more and more malicious attacks against web applications.

Web application developers may not notice some logic or code details during the development process, resulting in vulnerabilities in the program that can be exploited by hackers. Common attack behaviors include malicious behaviors such as SQL injection, cross-site scripting attacks, and session hijacking [1]. Many of these attacks are based on URL construction and at the same time carry out intrusions against the database.

In order to improve the security of web applications and ensure the normal use of users, developers design and develop web application firewalls to protect web applications. Since the web request is implemented based on the HTTP and HTTPS protocols, the web application firewall also analyzes the characteristics of the protocol request packet to determine whether it is a malicious request and respond in time [2]. The web application firewall extracts the characteristic data in the HTTP data stream, and recognizes it through rule matching, machine learning algorithms and other methods [3] to realize the protection of web applications, but these traditional methods are insufficient in identifying new types of attacks. The deep learning method has a certain degree of self-learning ability, and has been applied to a variety of classification problems, and its superiority in identifying web application attacks has also been proven [4].

This paper designs and implements a set of Web application firewalls system based on convolutional neural networks. At the same time, the web application firewall is built on the cloud to improve the scalability and convenience of the overall system.

II. Material and Methods

A. System structure

In terms of system architecture, we deploy the entire system sub-modules on different servers. Through this method, the flexibility of the system can be further improved, and it is convenient for the system to be updated and expanded. We divide the system into WAF request processing server, deep learning processing server, Web front-end server and core database server. The specific server framework is shown in the Figure 1.

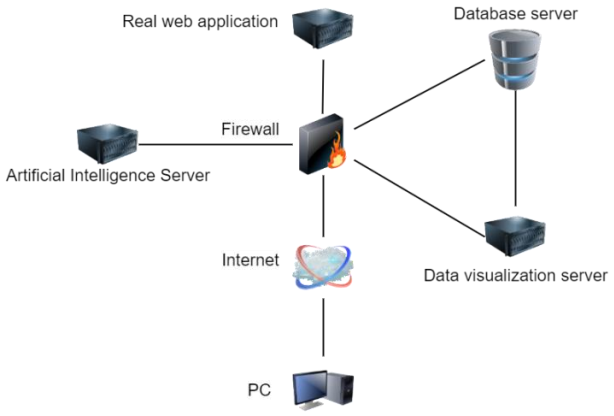

Figure 1. System architecture diagram.

978-1-6654-6812-1/22/$31.00 ©2022 IEEE
DOI 10.1109/CIPAE55637.2022.00101

The protected customer site will resolve the domain name to our WAF main server by changing the DNS. The WAF main server will make preliminary judgments on the received user requests through regular matching. This will be the first line of defense for our system. The malicious request is initially intercepted, and then the WAF main server forwards the user request to the artificial intelligence server. The artificial intelligence server calls a local script to encode the sent data and uses artificial intelligence algorithms to judge its safety, and returns the judgment result to the WAF main server, The WAF main server decides whether to release the traffic. If the artificial intelligence server returns the result as a normal request, the user traffic will be forwarded, and the response result of the back-end real web application server will be received and returned to the visitor; if it is judged to be a malicious request, The visitor request is intercepted, and the WAF main server returns a customized warning page.

B. System configuration

The WAF main server uses OpenResty, a high-performance platform based on Lua and nginx. OpenResty's integrated Nginx has unique advantages for processing high concurrent requests. For WAF platforms with huge network throughput, Nginx load balancing configuration is also very convenient and efficient. At the same time, the Lua language, known for its fast speed, can also provide reliable services while providing reliable services.

In terms of load balancing, if the polling strategy is used, then multiple visits of a user may be sent to each server. This strategy is very inefficient due to the problem of multiple establishments of http links. Therefore, we consider using the Hash processing method for load balancing, that is, the ip_hash strategy provided by nginx. Hash the user's IP address and map it to different servers. In this way, each user's request to the same server can be satisfied, and the load balance between different users can be satisfied.

At the same time, in order to ensure the security of the system, we adopt a port restriction strategy. That is, the entire system is placed in the internal network, and only the WAF main server and the data visualization server are connected to the external network. The connections of other servers are bound to specific ports, and the server is configured to adopt a whitelist method to allow only specific intranet hosts to access specific port services.

C. Collection and processing of malicious URL data sets

In terms of selecting the URL data set, we consulted relevant information on the Internet and collected a number of Github warehouses of well-known malicious URLs. The final data set used the XSS, SQL injection and other attack payloads in Github's well-known malicious URL database Foospidy, and combined for other well-known warehouse payloads on Github, for the normal request data set, the normal log request of the website in the Security Repo was used, and finally 50,000 malicious request data and 100,000 normal request data were sorted out. As shown in Figure 2.

```
/id;1627282494;fp;2;fpid;1/
/<img src="javascript:alert(cross_site_scripting.nasl);">.do
/bmeun223.exe?<meta http-equiv=set-cookie content="testhhwu=7044">
/htbin/windmail.exe
/<script>document.cookie="testrluj=1420;"</script>
/javascript/.passwd.jpg
/opensiteadmin/scripts/classes/databasemanager.php?path=http://192.168.202.118:8080/ghl9il?\x00
/examples/jsp/jsp2/el/search=<script>alert('xss')</script>
/javascript/signer.exe
/help.php?q="&del\x0bq26193259&rem\x0b
/c'hoario\xc3\xb9/
/main.php?logout="&del\x0cq31768299&rem\x0c
/\xd0\x97\xd0\xb4\xd0\xbe\xd1\x80\xd0\xbe\xd0\xb2'\xd1\x8f/
/themes/modern/user_style.php?user_colors[bg_color]="</style><script>alert(411136083423)</script>
/nyjgaorz.mscgi?<img src="javascript:alert(cross_site_scripting.nasl);">
/cgi-bin/index.php?op=default&date=200607' union select 1,501184215,1,1,1,1,1,1,1,1--&blogid=1
/scripts/cfooter.php3
/en-us/dda2qr7j.fts?<script>cross_site_scripting.nasl</script>
/?<meta http-equiv=set-cookie content="testpokn=7494">
/169okeyj.jspa?<meta http-equiv=set-cookie content="testxeoi=3573">
/j734qobz.aspx?<img src="javascript:alert(cross_site_scripting.nasl);">
```

Figure 2. Display of part of the data set.

D. Data processing

The obtained data set is the string after removing the domain name containing the host address information. The URL string is not as complicated as natural language. The characters that can appear in the URL string are limited, mainly including uppercase and lowercase letters and numbers. And some special characters. So, you can count all the characters that can appear in the URL, and then perform One-Hot Encoding on all the characters. In this way, a URL string can be converted into a large 0/1 matrix, which can complete the initial digital representation of the URL, and finally transform the entire data set into a numpy vector of shape (90000, 50, 101).

E. Neural network model

Convolutional neural network is very fast. The basic convolutional neural network consists of three structures: convolutional layer, pooling layer, and fully connected layer. The convolutional layer is responsible for feature extraction of the input data. The convolutional layer contains multiple convolution kernels, each element of which represents a weight coefficient and deviation, which is similar to the

neuron of the feedforward neural network. Through the movement and calculation of the convolution kernel, the convolution layer realizes the extraction of data features. The pooling layer extracts the statistical information of the feature map through the preset pooling function, and realizes the selection and filtering of the feature information extracted by the convolutional layer. The fully connected layer of the convolutional neural network is equivalent to the hidden layer in the traditional feedforward neural network, and the purpose is too non-linearly combine and output the extracted features. As shown in Table I.

TABLE I PARAMETERS OF THE NEURAL NETWORK MODEL

optimizer	Adam
loss	binary_crossentropy
epochs	50
batch size	256
activation	Sigmoid

III. RESULTS

A. Neural network performance

After 50 rounds of training, the training accuracy of the neural network can reach 99.14%. From the Figure 3, it can be seen that the accuracy of the training process quickly reaches a high level, and gradually improved afterwards. The overall training loss (Figure 4) showed a downward trend. After 2 rounds of training, it dropped to a lower level, and finally decreased to 0.0286 at 50 rounds.

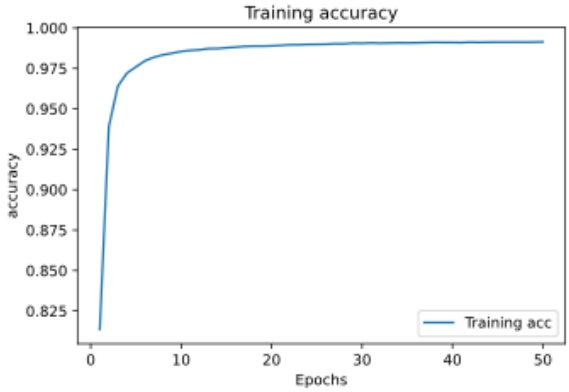

Figure 3. Training accuracy of neural network.

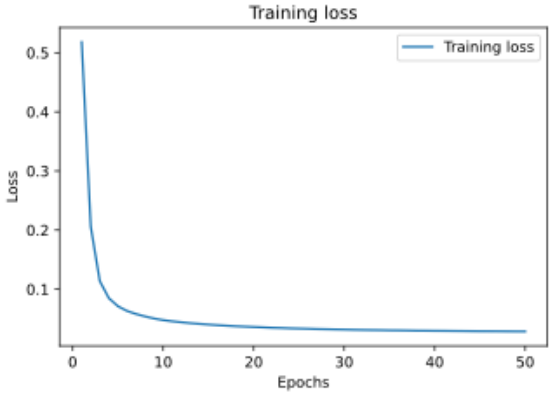

Figure 4. Training loss of neural network.

B. System performance test

We configure the system in the cloud, and use a server with 1 core CPU 1GB; 40GB SSD; 3Mbps peak bandwidth. Members of our team use personal PC computers to run scripts to access the cloud server, and the server response is calculated by the Python script.

After multiple visits to the WAF server for an access speed test, the average access speed is about 153ms.

sqlmap is an open-source penetration testing tool that can automatically detect and exploit SQL injection vulnerabilities and take over the database server. It has a powerful detection engine and many functions at the same time, such as database fingerprint recognition, data acquisition from the database, access to the underlying file system, and in-band connection execution commands on the operating system. We use sqlmap in the kali operating system to perform performance tests on the system. First, through preliminary tests, it can be concluded that sqlmap first determines that there is a firewall on the website and asks whether to conduct further testing. At the same time, the well-known XSS detection tool XSStrike was used to detect the WAF system, and it was found that the interception rate was also at a high level. As shown in Figure 5.

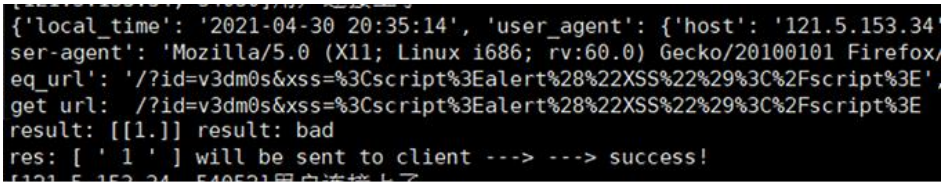

Figure 5. System output.

IV. CONCLUSION

This article proposes and implements a set of Web application firewall system for Web application security. The core technology of the system lies in the use of convolutional neural networks to identify malicious URLs, and the use of regular matching to configure Web application firewalls to achieve protection against DDOS attacks and CC attacks. The test proves that the system has a certain feasibility. Future research work will include further improvement of artificial intelligence algorithms, achieving higher recognition accuracy and faster recognition speed, and improving the security of the cloud web application firewall system itself.

ACKNOWLEDGMENTS

This work was financially supported by Zhengzhou University Student Innovation and Entrepreneurship Training Program fund.

REFERENCES

[1] Clincy, Victor, and H. Shahriar. "Web Application Firewall: Network Security Models and Configuration." IEEE Computer Software & Applications Conference IEEE Computer Society, 2018: 835 - 836.

[2] Makiou, A, Y. Begriche , and A. Serhrouchni. "Improving Web Application Firewalls to detect advanced SQL injection attacks." International Conference on Information Assurance & Security IEEE, 2014.

[3] Betarte, G. , L. Pardo, and R Martínez. "Web Application Attacks Detection Using Machine Learning Techniques." 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA) IEEE, 2019.

[4] Zhang, K. "A Machine Learning Based Approach to Identify SQL Injection Vulnerabilities." 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE) ACM, 2019.