2018 42nd IEEE International Conference on Computer Software & Applications

Web Application Firewall: Network Security Models and Configuration

Victor Clincy[1], Hossain Shahriar[2]
[1]Department of Computer Science, Kennesaw State University
[2]Department of Information Technology, Kennesaw State University
{vclincy, hshahria}@kennesaw.edu

Abstract- Web Application Firewalls (WAFs) are deployed to protect web applications and they offer in depth security as long as they are configured correctly. A problem arises when there is over-reliance on these tools. A false sense of security can be obtained with the implementation of a WAF. In this paper, we provide an overview of traffic filtering models and some suggestions to avail the benefit of web app firewall.

Keywords: Web application, Firewall, Network Security, STEM Career.

I. INTRODUCTION

Reliance on web applications is essential in our daily activities such as online banking, ticket booking, and sharing of data with others. Web applications are developed to serve HTML pages based on server side implementation that can be done in various languages such as PHP and ASP. They normally have a backend database as well such as MySQL. Applications, however, contain bugs that may be exploited by attackers for profit and fun [1].

Common bugs include SQL Injection, cross site request forgery, email injection, session hijacking, cookie stealing, cross site scripting and many more. Application developers have the burden of ensuring input data get validated or sanitized, and code is checked for vulnerabilities. Even if a developer implements programs accurately, some vulnerabilities still may surface due to the usage of a default server configuration [3].

Further, legacy applications may be deployed for years without being updated which can leave them open to security vulnerabilities. Thus, an extra layer of protection is needed.

Web Application Firewall (WAF) performs a deep packet inspection of network traffic that occurs between the client and the server sides. By analyzing the data transferred between the client and the server, WAF can identify possible attacks even if the implementation may be missing such a detection.

A number of WAFs are already available in the market. These include Barracuda, Bee Ware, Breach Security, Citrix, F5, Fortinet, and Imperva. One of the most common WAFs is Apache's Mod Security. (F5) Mod Security is popular among LAMP environments as it is open source and free and works with Apache server. It allows to perform simple filtering, regular expression based filtering, URL encoding validation, Unicode encoding validation, auditing, null bye attack detection, upload memory limit violations and servicer identity masking, just to name a few [3].

II. FIREWALL RULE MODEL

WAF can have two types of security models based on the policy type: positive or negative. A positive security model only allows traffic to pass that matches with the policies. All other traffic is blocked. A negative security model allows all traffic to pass and attempts to block only the traffic represented by malicious rules. Generally, most firewalls use either positive or negative rules, rarely both [4].

In a positive security model, packets are analyzed to ensure that the information is allowed. For example, an input field is supposed to be numeric only. If the incoming characters are not numbers, then the defined rule would not allow the input to be routed to the destination. Some other examples of positive security model usage include filtering messages that supposed to contain an email address but include non-email addresses such as a zip code or etc, or filtering an input field with a length of 80 characters or less, but the input exceeded the length. Several of the WAFs available in the market first go through a training period when web applications are profiled during test usage. This helps the firewall to learn what type of input is normally expected from applications. After learning the profiles, the firewall begins to automatically configure the security rules to validate input data.

Imperva's web application firewall works in this manner, it requires a training period or to be manually configured [5]. The downside of manually configuring a WAF is that the security professional responsible for the configuration must know all possible valid and invalid input and output to be processed by applications. As an example, to prevent an SQL injection attack, a WAF should be configured in a whitelisting mode (positive rules) where all legitimate requests to the application are allowed and everything else is to be blocked by default. Blacklisting policies, however, can be used for the deployment mode for filters [6]. Whitelist requires extensive planning and a thorough knowledge of the web application and acceptable, approved input values. Fine

0730-3157/18/$31.00 ©2018 IEEE
DOI 10.1109/COMPSAC.2018.00144

tuning a WAF can take a lot of time and resources and requires extensive knowledge of the system. An example of a whitelisting rule in Mod Security is shown below:

```
<Location /var/www/html/myapp/index.php>
  SecRule &ARGS "!@eq 1"
  SecRule ARG_NAMES "!^search$"
  SecRule ARGS:search "!^\d{a-z}$"
</Location>
```

The above example applies to the index.php file's processing. It defines what an acceptable value is for an *input argument* (e.g., value not equal to 1) and *name* (e.g., name should not contain $ characters) would be for the parameter named search.

In a negative security model, the packets are inspected; if something is identified as malicious, then the packet is blocked. This model alone may not be sufficient as hackers are always identifying new ways to bypass security policies. For example, blacklisting input characters may not be enough as hackers may obfuscate characters to bypass the validation of input to perform email injection.

If a negative model alone is used in a WAF, hackers may be able to bypass the inspection. For instance, a WAF may blacklist the text ' or 1=1— ;however, ' or 2=2—may go undetected. Thus, it is recommended that a combination of both the negative and positive model would result in better protection.

III. CONFIGURATION OF FIREWALL

Configuration of Web Application Firewalls takes someone with technical knowledge of the web application to know what information should be allowed and what shouldn't be allowed. Additionally, web servers that host web applications normally do not just contain one web application: there may be multiple applications running on the same server. The configurations set on that server must fit for each of the web applications running. The needed policies may be conflicting with each other. For example, one application might need to allow for file uploading whereas another application could not need file uploading. In such a case, if the system administrators simply disabled all the rules on the WAF, it could lead to security holes in the deployed environment.

Misconfiguration in a web application firewall can be just as devastating as not having any protection. The additional responsibility falls on the developer to ensure that the code is secure. Many developers, however, are not well taught to develop secure code. Those that have been taught how to develop secure code are forced to weaken their security by providing functionalities that are user friendly focused as opposed to security focused. One approach to overcome this issue is to test applications to ensure server configuration rules are not posing security threats, such as by providing a large input of 5000000 character in a text field to cause a buffer overflow in the server.

Another approach is a hybrid method in detecting attacks by using character distribution and session anomaly [2]. In this method, a statistical approach is taken to analyze the character frequency and exceptional inputs are generated.

IV. CONCLUSION AND FUTURE WORK

In this paper, we discuss the necessity of a Web Application Firewall (WAF), and provided the strengths and weaknesses of positive and negative policy-based attack detection models. Using the default configuration of a web server may lead to vulnerabilities despite having a firewall; this should be addressed through security testing. The future research work includes comparison of various web application firewalls and default security configurations, followed by appropriate mitigation strategies. We plan to explore the relation and defense-in-depth by connecting web app firewall with traditional network firewall and intrusion detection systems.

REFERENCES

[1] L. Desmet, F. Peissen, W. Joosen, and P. Verbaeten, "Bridging the Gap Between Web Application Firewalls and Web Applications," *Proceedings of the fourth ACM workshop on Formal methods in Security*, pp. 67-77, Alexandria, Virginia, USA, November 2006.

[2] A. Tekerek, C. Gemsy, O. Bay, "Development of a Hybrid Web Application Firewall to Prevent Web Based Attacks," *Proc. of 8th IEEE International Conference on Application of Information and Communication Technologies (AICT)*, Oct 2014.

[3] A. Razzaq, A. Hoor, S. Shahbaz, M. Masood, H. Ahmad, "Critical Analysis on Web Application Firewall Solutions," Proc. of *IEEE Eleventh International Symposium on Decentralized Systems (ISADS)*, March 2013.

[4] J. Beechey, Web Application Firewalls: Defense in Depth for Your Web Infrastructure, 2009, Accessed from https://www.sans.edu/student-files/projects/200904_01.doc.

[5] M. Gendron, Imperva Introduces First Network Adaptive Web Application Firewall. 2006, http://investors.imperva.com/phoenix.zhtml?c=247116&p=irol-newsArticle&ID=1596618

[6] M. Heiderich. E. Nava, G. Heyes, D. Lindsay, Web application obfuscation, Elsevier, 2011, accessed from https://doc.lagout.org/security/Web%20Application%20Obfuscation/Web%20Application%20Obfuscation.pdf