SECURING THE SECURITY OF NETWORK PROTOCOLS AND ARCHITECTURE: FIREWALL WEB APPLICATION

Achmad Syafii ie^1, Akira Muhammad Azra[2]

[a]*Computer Science Department, School of Computer Science, Bina Nusantara University, Jakarta, Indonesia 11480*

ABSTRACT

The implementation of security systems on the web is necessary considering that the web itself can be accessed through a public network. This research aims to enhance the security of web applications by implementing a Web Application Firewall (WAF) based on ModSecurity and the Core Rules Set from Owasp. An experimental method is employed by deploying WAF as a web-based protection system, followed by analysis and testing to obtain accurate recommendations for firewall implementation. The research results indicate that the use of ModSecurity as a WAF successfully prevents attacks from attackers using methods such as Cross Site Scripting (XSS) and SQL Injection, providing an effective layer of protection against existing security threats. The implementation of WAF has the potential to be an effective solution in enhancing the security of web applications in open network environments, such as integrated campus management information systems.

CHAPTER 1 - INTRODUCTION

Background

Security in a web application is an essential aspect that must be addressed. Securing a web application can be achieved by installing firewalls, antivirus software, or similar software on computers or routers directly connected to or within the same network as the web application server. Security in web applications can be enhanced by using a web application firewall (WAF) deployed on the web server service. According to Roji (2010), a firewall is a mechanism applied to hardware, software, or systems themselves with the aim of protecting, filtering, restricting, or even rejecting connections/activities from one segment of a private network to an external network that is not within its scope. A web application firewall (WAF) is the primary front-end protection mechanism for Internet-based infrastructure that is constantly under attack. WAF analyzes incoming network traffic to the web application and eliminates unwanted or harmful traffic before it reaches the application. WAF can be used as a system to prevent and detect SQL injection, Cross-Site Scripting (XSS), and Local File Inclusion (LFI) attacks by using pre-established detection rules to block access for attackers into the website.

Problem Statement

1. How effective is the implementation and management of web application firewall in ensuring the security of network protocols and architecture to protect systems and data from potentially damaging cyber attacks?
2. What techniques and strategies can be applied to enhance the security of network protocols and architecture using web application firewall to prevent potentially damaging cyber attacks?

Scope

1. In-depth analysis of network protocol and architecture security.
2. Implementation and management of web application firewall.
3. Review of latest techniques in detecting and preventing cyber attacks.
4. Risk management related to network security.
5. Compliance with regulations in network security protection.

Objectives

1. Investigate the effectiveness of implementing and managing web application firewall to ensure the security of network protocols and architecture, focusing on preventing cyber attacks such as SQL injection, Cross-Site Scripting (XSS), and Local File Inclusion (LFI).
2. Enhance understanding of the role and function of web application firewall in securing network infrastructure.
3. Evaluate latest techniques in detecting and responding to cyber attacks on web applications.
4. Provide practical guidance for organizations in effectively managing web application firewall, including configuration, maintenance, and network security monitoring.
5. Make a significant contribution to enhancing network security and protecting web applications from various cyber threats, while providing practical guidance for organizations to improve their security strategies.

CHAPTER II - LITERATURE REVIEW

Web server applications often come under attack from various irresponsible parties commonly referred to as hackers or attackers. Hackers seek vulnerabilities in web servers for the purpose of obtaining information from organizations and companies for interests that may cause losses to others. A web application firewall (WAF) is a specialized firewall designed to filter and control HTTP traffic between web clients and application servers over the internet. A Web Application Firewall (WAF) is a crucial component for strong application security. It plays an important role in filtering, monitoring, and blocking HTTP traffic to and from web applications. Unlike conventional firewalls, WAF has the ability to filter specific web application content, while traditional firewalls typically act as gateways between servers. By inspecting HTTP traffic, WAF serves as the primary defense against attacks originating from security vulnerabilities in web applications, such as SQL Injection, Cross-site Scripting (XSS), file inclusion, and security configuration errors. The main advantage of using WAF is comprehensive and proactive protection for web applications at the application level, without requiring modifications to the applications themselves. Additionally, WAF provides security mechanisms such as URL encryption and enforcement of site usage to minimize attack surfaces as efficiently as possible, ultimately enhancing web application security against external threats.

Furthermore, WAF has clear advantages over conventional firewalls because it can provide greater visibility into sensitive application data communicated through the HTTP application layer. This allows WAF to prevent attacks at the application layer that can typically bypass network firewalls. Some attacks that can be prevented by WAF include Cross-site Scripting (XSS), SQL Injection, Web Session Hacking, Distributed Denial of Service (DDoS), DoS Layer 7, Buffer Overflow, and Cookie Poisoning. Since various types of DDoS attacks exploit the HTTP protocol, with most using methods at lower levels, WAF becomes a crucial defense mechanism against DDoS attacks at the application/layer 7 (HTTP/FTP) level, thus protecting web servers from excessive application activity.

WAF, operating as a web-based application, functions as a filter, monitor, and traffic blocker on the Hyper Text Transfer Protocol (HTTP). Unlike conventional firewalls, WAF has the ability to filter attacks from various content aimed at web applications. This capability allows WAF to

promptly address and prevent attacks such as SQL Injection, Cross-site Scripting (XSS), file inclusion, and configuration errors by detecting HTTP traffic. The operational workflow of WAF depicts the condition when the Web Application Firewall (WAF) is inactive in the system, allowing potentially harmful requests exploiting security vulnerabilities to directly access the database server and web server. As an integral part of the system, nginx acts as the web server, while naxsi acts as the web application firewall. Naxsi is responsible for filtering content that passes through and blocking content deemed harmful according to predefined rules.

CHAPTER III – METHODOLOGY

At this stage of the research the author describes the steps that will be carried out in this research. The following steps are carried out as follows:

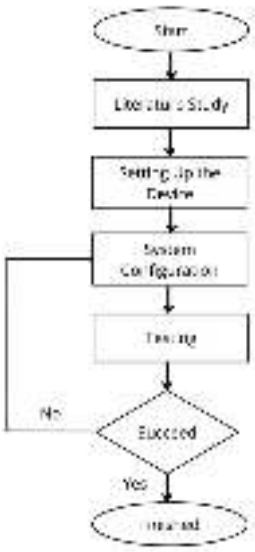

Figure 1. Research Stages

Study of literature

The initial stage before the research is carried out involves identifying the problems that arise. One of the problems that arises is related to the increase in the number of internet users in Indonesia. With the increasing number of internet users, there is an increasing trend of cyber attacks aimed at websites owned by companies or organizations in Indonesia. The large number of websites used by companies or organizations to carry out transactions and provide services to clients increases the risk of cyber attacks. This attack can cause the leak of personal data from clients or companies stored on the server.

WAF System Requirements Analysis

This stage is based on the results of observations made on the web server used in the integrated campus management information system. This research uses software requirements specifications as shown in table 1.

Table 1. Software Requirements

No	Software	Spesification
1	Operating System	Debian
2	Web Server (WS)	Apache Sv 2.5
3	Database Server (DS)	MariaDB
4	Programming language	PHP 7.3

WAF Testing

This study compares various models in implementing Web Application Firewall (WAF) to prevent SQL injection attacks on websites. These models are evaluated based on several key parameters such as effectiveness in preventing attacks, ease of configuration, website performance, and implementation complexity. Each model has its own advantages and disadvantages that are important to understand in order to choose the most appropriate model for a particular situation. The Web Application Firewall (WAF) testing phase uses 2 test scenarios. The first test is carried out with ModSecurity status inactive with Cross Site Scripting (XSS) and SQL Injection attacks on the integrated campus management information system web application. In the second scenario, the test is carried out with ModSecurity status active with the same

attack. This test is carried out to determine the load time of the website, there are two stages of this test, namely when the website is before and after implementing Web Application Firewall in it. From the test results, it can be seen that the load time of the website increased from before using the WAF security mod and after using the WAF security mod. ModSecurity is one of the most commonly used WAFs. Its advantages include the ability to detect and block SQL injection attacks in real-time. ModSecurity has flexibility in terms of configuration and can be integrated with Apache and Nginx servers. Additionally, it offers customizable rules, allowing users to tailor protection to their specific website needs. The main downside of ModSecurity is the increased website load time due to the additional security checks performed on each request. While effective in preventing attacks, ModSecurity does have some drawbacks. Increased website load time can affect user experience, especially on high-traffic websites. ModSecurity configuration and maintenance also require a fair amount of technical knowledge, which can be challenging for users without a strong IT background. Additionally, while it can prevent many common attacks, ModSecurity may require periodic updates and adjustments to address new threats.

OWASP WAF is another model that is often used in web security implementations. Its advantages include a constantly updated rules database that can protect websites from a variety of attacks, including SQL injection. OWASP WAF is also known for its active community and good documentation, making it easier to implement and maintain compared to some of the alternatives. OWASP WAF also has analytics and reporting tools that can help in continuous monitoring of website security. However, like ModSecurity, OWASP WAF also has its drawbacks. Website performance can be affected by the additional security layer, although the impact may be smaller compared to ModSecurity. In addition, OWASP WAF can be too complex for initial configuration, requiring considerable time and effort to ensure that all rules are implemented correctly. Because OWASP WAF focuses on security, it is possible that some website functionality features may be disrupted if the rules are not configured properly.

Research conducted by Robinson et all showed that the load time of using OWASP WAF was obtained an average of around 5,028Ms using 100 simulated users for 1 minute. Testing conducted by Robinson et al. provides additional insight into the evaluation of Web Application Firewall (WAF) in web application security. In this study, testing was conducted with a focus on the effectiveness of OWASP ModSecurity in reducing SQL Injection attacks, SQLmap exploitation, and XSSer exploitation. Basically, SQL Injection is a way to exploit security holes that appear at the level or "layer" of the database and its applications. The security hole is shown in the form of a response from a request for incorrect data to the server that was deliberately sent by the attacker. There is also a similar test conducted by Yulianingsih, namely testing against Cross Site Scripting (XSS) attacks using the Sanitize Values Passed To Other Systems method with the Metacharacters technique. The test steps consist of two stages, namely the First Test without security and the Second Test with security. From the results of observations during the test, it can be concluded that the metacharacter method has also proven to be able to close the entry of Cross Site Scripting attacks on a site.

BAB IV – RESULT AND DISCUSSION

The results of the study show that the implementation of Web Application Firewall (WAF) using ModSecurity is effective in

preventing SQL Injection attacks. In the first stage, when the website is not yet protected by WAF, both SQLMap and JSQLInjection successfully exploited the website. SQLMap uses a time-based blind injection technique with a specific payload to execute the attack, while JSQLInjection utilizes various payloads to retrieve important information from the target database. Both tools successfully gain access and sensitive data from the website.

However, in the second stage after WAF implementation, both tools failed to carry out the attack. ModSecurity effectively blocked SQL Injection exploitation attempts carried out by SQLMap and JSQLInjection. SQLMap detected protection from WAF/IPS, which stopped further exploitation, while JSQLInjection could not find any injection gaps that could be exploited. This shows that WAF successfully closed all previously existing vulnerabilities and significantly increased the level of website security.

Performance testing showed an increase in website load time after WAF implementation. Before WAF implementation, the average load time was 78.9 ms, while after WAF implementation it increased to 115.9 ms. This increase is due to the additional checking process performed by WAF on each incoming request. Although there is an increase in load time, this difference is relatively small and acceptable, comparable to the additional level of security provided by WAF. This shows that WAF can improve security without significantly sacrificing user experience.

This study is in line with the findings of Robinson et al. (2020) and Risma et al. (2020). Robinson et al. found that OWASP WAF is able to prevent various types of SQL injection attacks with a high success rate. In their study, OWASP WAF successfully detected and prevented the attack attempts, indicating that this tool is effective in dealing with increasingly sophisticated threats. Risma et al. observed that Naxsi WAF for Linux Mint successfully identified and mitigated attacks carried out on web applications. Both studies support the results of this study, confirming that WAF implementation is an important step in improving the security of web applications without significantly sacrificing performance.

WAF not only serves as a protection mechanism against direct attacks, but also as part of a broader security strategy to reduce the risk of cyber attacks. By adopting ModSecurity-based WAF, organizations can improve the security of web applications without having to make major changes to the application itself. This makes WAF a practical and effective solution in dealing with various cyber security threats.

BAB V – CONCLUSION

1. Increased Load Time: WAF implementation, especially ModSecurity, causes an increase in website load time. This is because the WAF security system checks every incoming access before it is forwarded to the website. Although there is an increase in load time, this difference is relatively small and does not significantly affect the overall performance of the website. 2. Website Performance: Although there is a difference in load time (ms) between using OWASP WAF and ModSecurity WAF, the average HTTP load time does not actually affect web performance significantly. This shows that the additional security provided by WAF does not drastically sacrifice user experience.

3. Effectiveness in Preventing SQL Injection Attacks: WAF implementation has been proven to minimize SQL Injection attacks. ModSecurity works well in limiting attackers from executing SQL Injection, so that attackers are unable to get any information from the website. This security system is effective in reducing exploitable vulnerabilities.

4. Detection and Protection Success:

- OWASP ModSecurity successfully detected and secured web applications from SQL Injection attacks 100% after 15 tests using 3 different operating systems.

- OWASP ModSecurity failed to secure web applications from Stored Cross Site Scripting (XSS) attacks, which allowed attackers to successfully link the target PC. - OWASP ModSecurity successfully detected and secured web applications from exploitation using the SQLmap tool on 3 different operating systems. - OWASP ModSecurity successfully detected and secured web applications from exploitation using the XSSer tool on 3 different operating systems.

5. No Major Impact on Web Performance: The comparison results show that there is no major impact that will affect the performance of web applications after using OWASP ModSecurity. This shows that implementing a WAF can improve security without sacrificing performance significantly.

Overall, this study confirms that using WAFs such as ModSecurity and OWASP is an important step in improving the security of web applications. This additional security comes at the cost of only a slight increase in load time, which is still within acceptable limits and does not significantly impact the user experience. Implementing a WAF is an effective strategy for protecting web applications from a variety of evolving security threats.

REFERENCE

[1] Riska, R., & Alamsyah, H. (2021). Penerapan Sistem Keamanan Web Menggunakan Metode Web Aplication Firewall. *JURNAL AMPLIFIER : JURNAL ILMIAH BIDANG TEKNIK ELEKTRO DAN KOMPUTER*, *11*(1), 37–42. https://doi.org/10.33369/jamplifier.v11i1.16683

[2] Rizal, R., & Sumaryana, Y. (2021). Peningkatan Keamanan Aplikasi Web Menggunakan Web Application Firewall (WAF) Pada Sistem Informasi Manajemen Kampus Terintegrasi. *Jurnal ICT : Information Communication & Technology*, *20*(2), 323–330. https://doi.org/10.36054/jict-ikmi.v20i2.416

[3] Elanda, A., & Buana, R. L. (2020). Analisis Keamanan Sistem Informasi Berbasis Website Dengan Metode Open Web Application Security Project (OWASP) Versi 4: Systematic Review. *CESS (Journal of Computer Engineering, System and Science)*, *5*(2), 185. https://doi.org/10.24114/cess.v5i2.17149

[4] J. Karisma Anggreana, "Simulasi keamanan pada aplikasi web dengan web aplication firewall."

[5] Universitas Komputer Indonesia, 2014. *IMPLEMENTASI FIREWALL APLIKASI WEB UNTUK MENCEGAH*. (n.d.). Retrieved April 21, 2024, from https://digilib.uin-suka.ac.id/id/eprint/14398/2/10650030_BAB-i_iv-atau-v_daftar-pustaka.pdf

[6] Suryayusra, & Muharromin, M. (n.d.). Analisis Performance Web Application Firewall ModSecurity. *393*.

[7] Jamain, R. Y., Periyadi, P., & Ismail, S. J. I. (2015). Implementasi Keamanan Aplikasi Web Dengan Web Application Firewall. *EProceedings of Applied Science*, *1*(3). https://openlibrarypublications.telkomuniversity.ac.id/index.php/appliedscience/article/view/4232/3962

[8] Roji, Mukhamad Fatkhur. "SISTEM KEAMANAN INTERNET DENGAN MENGGUNAKAN IPTABLES SEBAGAI FIREWALL." *DINAMIKA DOTCOM* 1.1 (2010).

[9] Bangkit Wiguna, Adi Prabowo, W., & Ananda, R. (2020). Implementasi Web Application Firewall Dalam Mencegah Serangan SQL Injection Pada Website. *Digital Zone: Jurnal Teknologi Informasi Dan Komunikasi*, *11*(2), 245–256. https://doi.org/10.31849/digitalzone.v11i2.4867

[10] Wicaksono , D., & R. Widiasari, I. (2022). Sistem Keamanan Jaringan Menggunakan Firewall Dengan Metode Port Blocking Dan Firewall Filtering. *Jurnal Teknik Informatika Dan Sistem Informasi*, *9*.

[11] Profile, S., George, H., George, A. S., & George, A. (2021). A Brief Study on The Evolution of Next Generation Firewall and Web Application Firewall This work is licensed under a Creative Commons Attribution 4.0 International License A Brief Study on The Evolution of Next Generation Firewall and Web Application Firewall. *International Journal of Advanced Research in Computer and Communication Engineering*, *10*(5).

[12] Alghawazi, M., Alghazzawi, D., & Alarifi, S. (2022). Detection of SQL Injection Attack Using Machine Learning Techniques: A Systematic Literature Review. *Journal of Cybersecurity and Privacy*, *2*(4), 764–777. https://doi.org/10.3390/jcp2040039

[13] Robinson, Akbar, M., & Fadhly Ridha, M. A. (2018). SQL Injection and Cross Site Scripting Prevention using OWASP ModSecurity Web Application Firewall. *JOIV : International Journal on Informatics Visualization*, *2*(4), 286–292. https://doi.org/10.30630/joiv.2.4.107

[14] Dani, R., Permana, R., & Heriyanto, H. (2021, February 1). *ANALISIS JARINGAN KEAMANAN MENGGUNKAN IDS DAN HONEYPOT PADA TRANS STUDIO MINI PEKANBARU*. Repository.upiyptk.ac.id. http://repository.upiyptk.ac.id/5992/

[15] Yulianingsih, Y. (2017). Melindungi Aplikasi dari Serangan Cross Site Scripting dengan Metode Metacharacter. *Jurnal Nasional Teknologi Dan Sistem Informasi*, *3*(1), 83–88. https://teknosi.fti.unand.ac.id/index.php/teknosi/article/view/170/85

[16] Crotts, L. (n.d.). *Exploring Cross-Site Scripting (XSS): Attack Payloads, Prevention, and Mitigation Techniques*.

[17] Made Suartana, I., Wahanani, H., Sandy, A., Teknik Informatika, J., Timur, J., & Surabaya. (n.d.). *SISTEM PENGAMANAN WEB SERVER DENGAN WEB APPLICATION FIREWALL (WAF)*.

[18] Prokhorenko, Victor, Kim-Kwang Raymond Choo, and Helen Ashman. (2016) “Web Application Protection Techniques: A Taxonomy.” Journal of Network and Computer Applications 60: 95-112

[19] Xiaowei Li., Yuan Xue., 2013, A Survey on Server-side Approaches to Securing Web
Applications, ACM Transactions on Computing Surveys.

[20] Marashdih, A. W., Zaaba, Z. F., Suwais, K., & Mohd, N. A. (2019). Web Application Security: An Investigation on Static Analysis with other Algorithms to Detect Cross Site Scripting. *Procedia Computer*

Science, *161*, 1173–1181. https://doi.org/10.1016/j.procs.2019.11.230