ASTES

Advances in Science, Technology and Engineering Systems Journal Vol. 6, No. 2, 405-411 (2021)

www.astesj.com

ASTESJ

ISSN: 2415-6698

SEA WAF: The Prevention of SQL Injection Attacks on Web Applications

Jeklin Harefa*, Gredion Prajena, Alexander, Abdillah Muhamad, Edmundus Valin Setia Dewa, Sena Yuliandry

Computer Science Department, School of Computer Science, Bina Nusantara University, Jakarta, Indonesia 11480

ARTICLE INFO

Article history:
Received: 23 December, 2020
Accepted: 23 February, 2021
Online: 17 March, 2021

Keywords:
SQL Injection
Web Application Firewall
WAF

ABSTRACT

The security of website application has become important in the last decades. According to the Open Web Application Security Project (OWASP), the SQL Injection is classified as one of the major vulnerabilities found in web application security. This research is focused on improving website security in dealing with SQL Injection attacks by stopping, monitoring, and dividing types of SQL Injection attacks using the features provided by the proposed Web Application Firewall (WAF). The architecture is designed to detect and prevent some types of SQL Injection attacks, including Tautologies, Logically Incorrect Queries, Union Queries, Piggy Backed Queries, Stored Procedures. For the testing scenario, this experiment uses an application that has become an industry standard in identifying and validating security holes on a website. The result of this research is that the proposed system is able to increase the website security from SQL Injection.

1. Introduction

The growth of technology has been evolved tremendously in many aspect. One of the examples is the World Wide Web evolve from 1.0 to 6.0 [1]. It shows that this evolution also needs to improve the web security. Nowadays, cyber criminals have exploited the COVID-19 pandemic situation to launch a highly sophisticated cyberattacks in every industry. In the first quarter of 2020 (Q1 2020), even 8.4 billion records have been revealed [2]. Data breaches frequently occur at various companies, ranging from small-scale companies to large-scale companies such as eBay, TJX Companies, Inc., Uber, JP Morgan Chase, Sony's PlayStation Network, Home Depot, Adobe, and many others [3]. Those companies use the web applications and web services that can be accessed online, and their application are also exposed to the public. Most web attack target the vulnerabilities of web applications [4].

OWASP (Open Web Application Security Project), a non-profit foundation that works to improve software security, has announced the top 10 Web Application security risks. Based on these data. It is further stated that the highest level of security risk in web application is injection [5]. Those injections can occur when some suspicious data is sent by a command or query. Injection consist of SQL (Structured Query Language), NoSQL, OS (Operating System), and LDAP injection. As one of the common injections, SQL Injection is to cheat the server to execute malicious SQL commands like CRUD (Creating, Reading, Updating, Deleting) by inserting SQL commands into the query string of Web form submission or input domain name or page request [6], [7].

There are several types of SQL Injection including Tautologies, Logically Incorrect Queries, Union Queries, Piggy Backed Queries, Stored Procedures [8], [9]. Tautologies is an attack to produce TRUE condition by using '=' (equal) to the query. This type of SQL injection can bypass the authentication page and get the extracted data. The next type of SQL Injection is Logically Incorrect Queries. This attack tries to gather information about the database by injects query with type mismatch, syntax error or logical errors. This attack will generate an error message with some useful debugging information. While for Union Queries is an attack to combine two or more queries by using UNION syntax. They used the 'union' keyword to combine the original query and injected query and get some information from database. Piggy Backed Queries is an attack to inject additional queries to the original query. For this injection, the attacker tries not to modify the query, but they execute multiple queries which may lead to Database exploitation. The last type of SQL Injection is Stored Procedure. Stored Procedures is an attack execute or create stored procedure in database. This will result in remote commands and a denial of service.

This paper contributes to detecting 5 types of SQL injection attacks, including: Tautologies, Logically Incorrect Queries, Union Queries, Piggy Backed Queries, Stored Procedures. This paper also contributes to the development of a web-based-

*Corresponding Author: Jeklin Harefa, Email: jharefa@binus.edu

https://dx.doi.org/10.25046/aj060247

application that can block the attacker's IP either manually or automatically, and an additional map feature that can show the number of attacks from various countries based on their IP addresses. In addition to increasing security, this could benefit the SecOps (security operations) in analyzing SQL Injection attacks.

2. Recent Work

Several works have been done to prevent the attacks on SQL injections. In 2020, Hlaing and Khaing presents an approach which detects a query token with reserved words-based lexicon to detect Structured Query Language Injection Attacks (SQLIA). The proposed system is done by using two approaches, which are: creates lexicon and tokenize the input query statement and each string token was detected to predefined words lexicon to prevent SQLIA. Based on the experiment conducted, the proposed system is able to provide a successful prevention from various malicious query for injections [10].

Another research comes from [11] the authors introduce the principle of SQL injection attack: The main form of SQL injection attack, types of injection attack and how to prevent SQL injection. In general, there are several types of SQL injection attacks: the escape character is not filtered correctly, incorrect type handling, vulnerabilities in database servers, blind SQL injection attack, conditional errors, conditional response, and time delay. They also proposed some codes to prevent SQL injection.

In [12], the authors analysed some methods of detecting and preventing the SQL injection attacks such as AMNESIA, SQLCHECK Approach, CANDID, Auto-mated Approach, WASP, Swaddler, Tautology Checker, WebSSari and Ardilla. Based on the research conducted, it can be implied that the structure of developing web application must be considered carefully to avoid various types of SQL injection attacks.

In another study, [13] the author proposed a secure coding approach that can be used by the developer for the prevention of SQL injection attacks to secure their application. They focus the research for prevention of SQL injection attack on: Input and URL validation, data sanitization, prepared statement (or PDO) for query execution, Query, and session tokenization. The result of the proposed research proved to be very effective and efficient in preventing different types of SQL injection attacks.

In 2019, the authors explained, detected, and described the various risk prevention mechanism Injection from the top ten vulnerability risk of OWASP, including, SQL Injection, Broken Authentication, Sensitive Data Exposure, XML External Entities, Broken Access Control, Security Misconfiguration, Cross-site Scripting, Insecure Deserialization, Using Components with Known Vulnerabilities, and Insufficient Logging and Monitoring [14]. They also explained how the attacker identify the vulnerability code and describe some injection risk prevention methods in the web application.

3. Proposed Method

The proposed web application firewall aims to detect and prevent the SQL injection attacks. Several types of SQL injection attacks that can be detected by the proposed system are tautologies, piggy-backed, stored procedures, union query and logically incorrect query. The architecture designed of proposed web application firewall uses Server Resident (or Embedded WAF), where the firewall installed on the host executing the web server [15]. This architecture places the WAF on the server, as it can be seen at Figure 1.

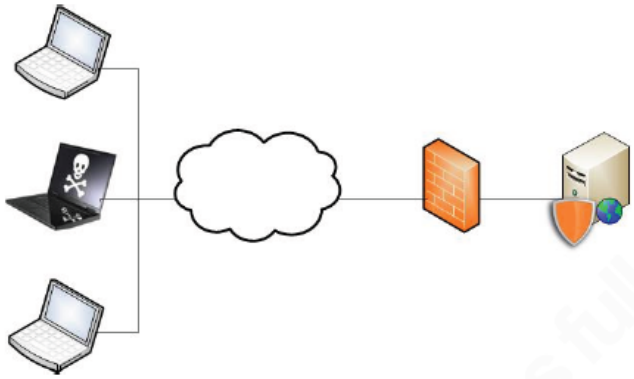

Figure 1: Server Resident Architecture

Based on the Figure 1, the WAF represented by the shield, installed on the protected server. By applying this topology, WAF can protect clients that runs the web service from SQL injection attacks. Figure 2 represents the server resident architecture applies on the proposed system (SEA WAF).

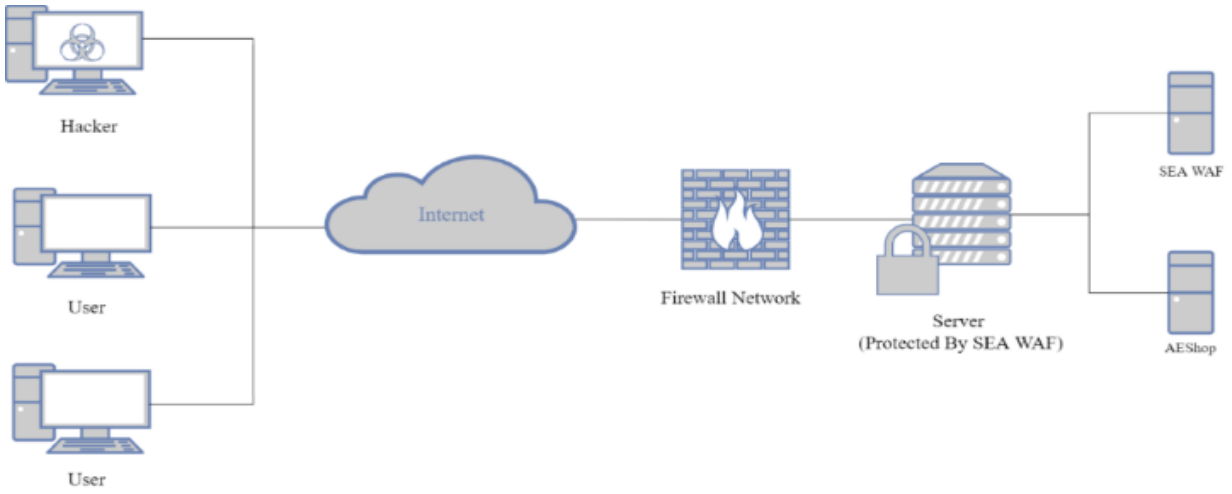

Figure 2: SEA WAF Architecture

The architecture of the proposed Web Application Firewall can be seen at Figure 2. Based on the architecture, firstly, the user/hacker uses the internet connection to access the website the goes through the network firewall and after that, the server will receive the requests from the user/hacker. When a request comes into the server, the request will be filtered first by the proposed firewall (SEA WAF). Furthermore, the SEA WAF will filter the 5 different types of SQL Injection attacks, which are: tautologies, piggy-backed, stored procedures, union query and logically incorrect query. If the request is marked as safe from the 5 types of SQL injection attacks that have been mentioned before, then the request will be continued to the client web service. However, if the request is marked as dangerous, then the request will be dropped, and a blocked message will be appeared. Some of the features that available in the SEA WAF application, including view a list of visitor requested on the website, remove the attacker from the blocked list, check the attack state based on the attacker's IP address, manually block the attacker, change the safe request to unsafe request according to the user, automatically block the attacker based on the number of attacks carried out, set up the activation of SEA WAF and manual setting for blocking the attacker.

To find out if the proposed web application firewall can prevent the SQL injection attack, this research use an e-commerce website that build with PHP Laravel Framework called AEShop as targeted website. AEShop is an online website that uses a concept such as a department store with a one stop fashion theme. This online shop offers various types of clothing in both of male and female categories. For the test scenario, the SEA WAF is installed on the AEShop website using the burp suite application [16] to check if the proposed system is capable of handling the SQL Injection attacks. To provide a brief overview of the existing WAF applications, this paper also conducted a feature comparison that handles SQL Injection attack problems, with the application comparison including CloudFlare and Barikode WAF.

4. Experimental Result

In this section we compare two Web Application Firewall. CloudFlare (https://cloudflare.com) and Barikode (https://ethic.ninja). CloudFlare (Figure 3) is a Content Delivery Service that use as Web Application Firewall as well. CloudFlare sits in the middle between domain and web hosting. CloudFlare uses a set of rules to filter and monitor HTTP traffic between web applications and the Internet. As a third-party intermediary for the domain and web hosting, CloudFlare also plays a role in minimizing the threat such as Brute Force Attack, Cross-site Scripting, and SQL Injection as well [17]. In this research we will focusing on SQL Injection threat. CloudFlare only prevents certain SQL Injection, Tautologies, Logical incorrect and Piggy backed queries.

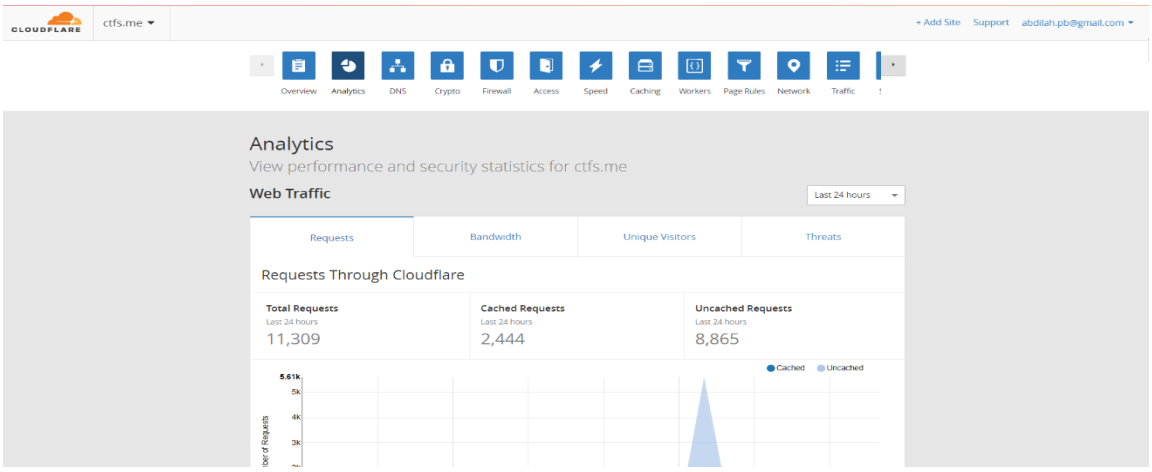

Figure 3: CloudFlare Admin Panel

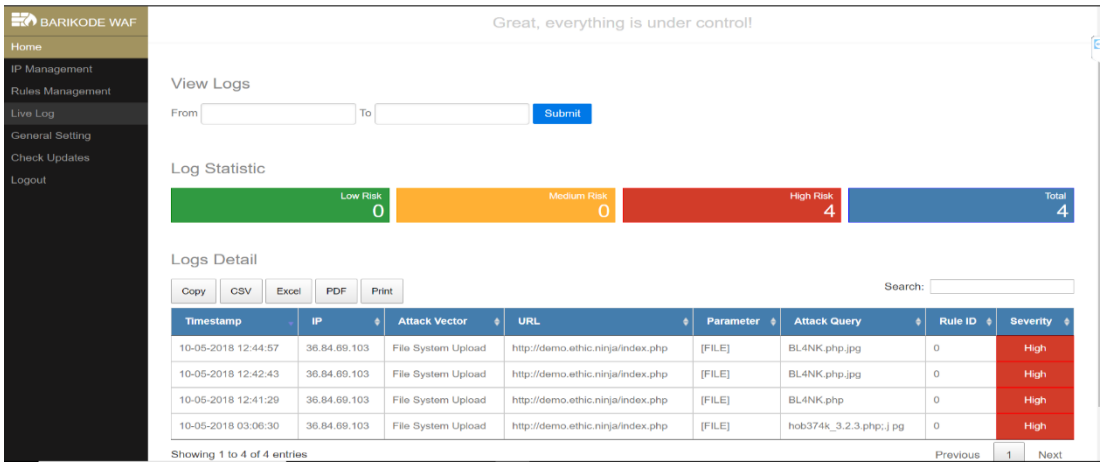

Figure 4: Barikode WAF Admin Panel

The second app Barikode WAF (Figure 4) is a website-based firewall that guard both public and internal websites (ERP, sisfo, etc.). Barikode has the ability to detect attacks by hackers early, monitor live traffic, and detect malicious scripts (shells / backdoors) that have been planted by hackers [18]. Barikode WAF can be install on any CMS or PHP Framework such as Laravel, CodeIgniter, Wordpress, Joomla, etc. In the security area, Barikode WAF can prevent threat like SQL Injection, XSS, local file inclusions, and remote code execution. For the SQL Injection threat Barikode WAF only cover Tautologies and Logical Incorrect Queries.

Table 1: Attack Type Comparison Result

Attack Type	CloudFlare	Barikode WAF
Tautologies	✓	✓
Logically Incorrect Queries	✓	✓
Piggy Backed Queries	✓	
Union Queries		
Stored Procedures		

Based on the Table 1, the CloudFlare and Barikode WAF applications only detect the tautologies, logically incorrect queries and Piggy backed queries Injection (Piggy backed queries only works for CloudFlare application). While for the union queries and stored procedures injection, both CloudFlare and Barikode WAF applications are unable to prevent these two types of attack. Therefore, the SEA WAF application is designed to be able to detect and prevent five types of SQL Injection attacks, which are: Tautologies, Logically Incorrect Queries, Piggy Backed Queries, Union Queries and Stored Procedures.

5. Result and Discussion

The result of this research is a web application firewall called SEA WAF that can prevent SQL Injection. SEA WAF have several web pages to help administrator to manage the data. After user login, SEA WAF will display a dashboard (see Figure 5) containing important things including five types of SQL Injection attacks with a total of attacks from each type of SQL Injection, a list of five IP addresses based on the highest number of accesses, and then a box containing the name of the country and the total attacks received based on sending attacks from the IP address location. Then finally there is a line graph box that contains the number of attacks received by websites that have been accumulated for each month.

As can be seen in Figure 6 regarding IP management page, there are IPs that have attacked the website and were blocked by SEA WAF. There is a function for manual blocking on this page by entering the IP address that is suspected of carrying out the attack and determining the blocking period by pressing the block button. User can also see the country from their IP address and delete IPs that have been blocked so they can re-access the website.

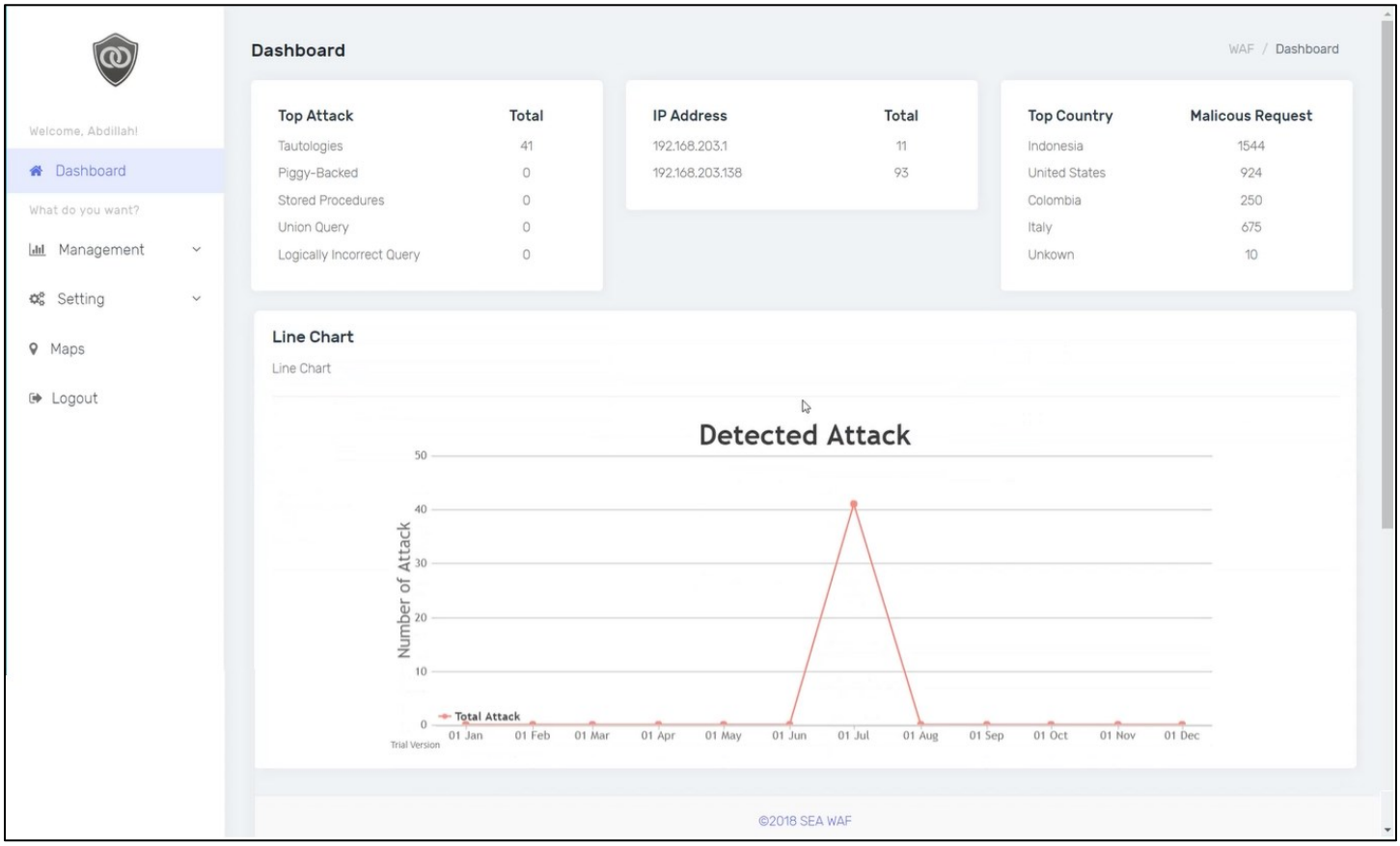

Figure 5: Dashboard

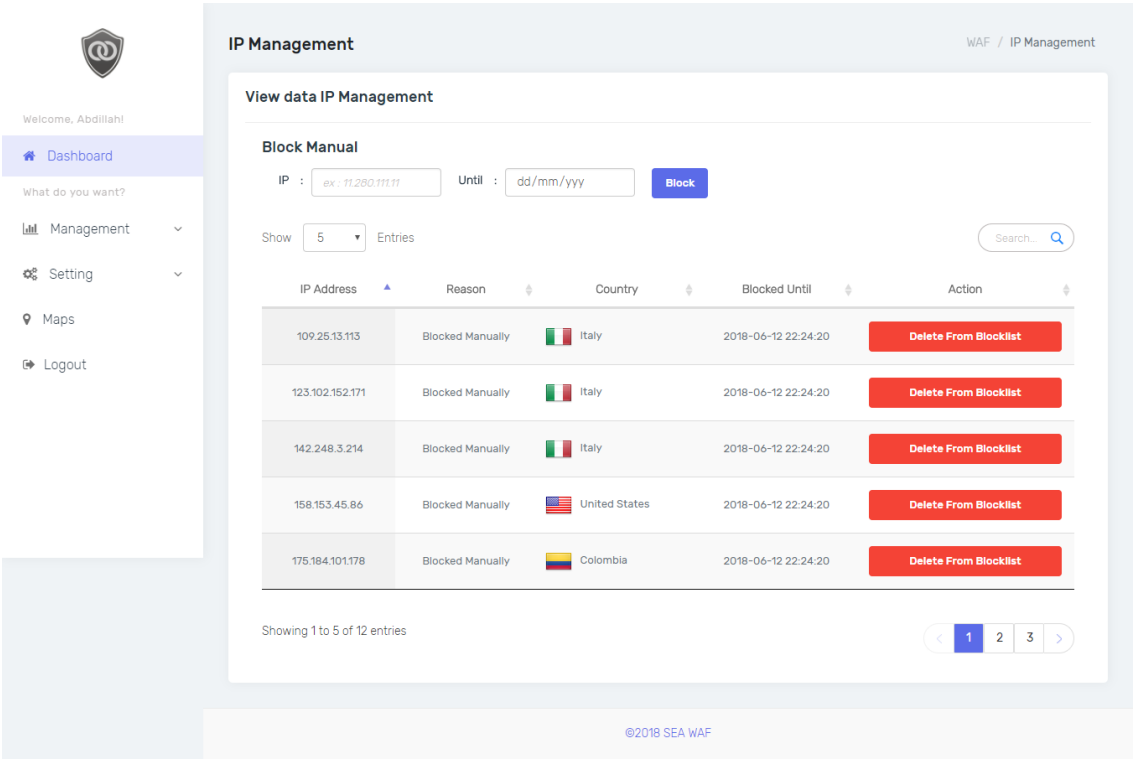

Figure 6: IP Management

Based on the Figure 7 (Log Management page), user can see log of each activity of the targeted web. There are several information about the activity including HTTP method, URL, Query String, IP Address, and Detection. If the activity categorized as one of the SQL Injection attacks, then it will be marked "NOT SAFE: [SQL Injection Type]". User also can mark as safe, mark as malicious, or block the IP address.

The world map page shows the map with number of attacks based on the location of the IP address (Figure 8). If the user hovers the cursor to the map area, the name of the country and the total number of attacks the website has received from that country will appear. This map categorized attack by level of risk, from No Risk/Clear Risk (Green colour) to High Risk (Red colour).

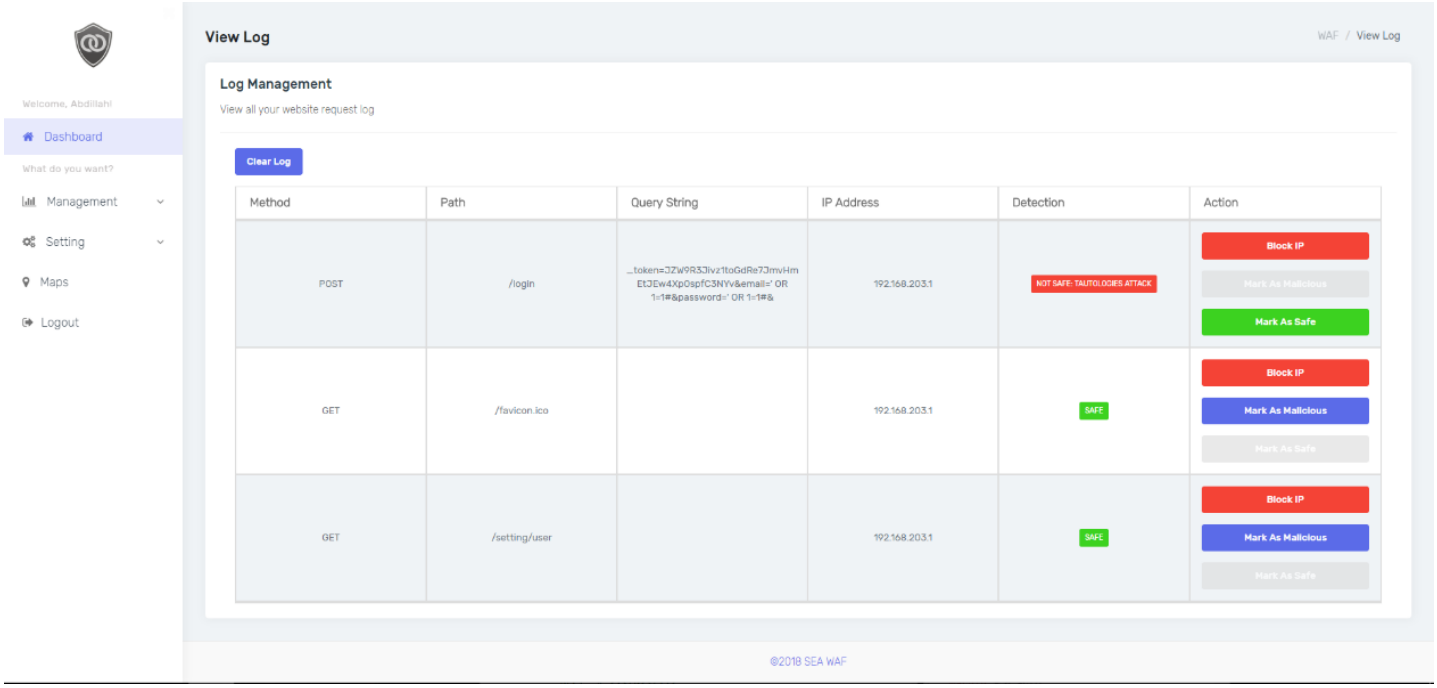

Figure 7: Log Management

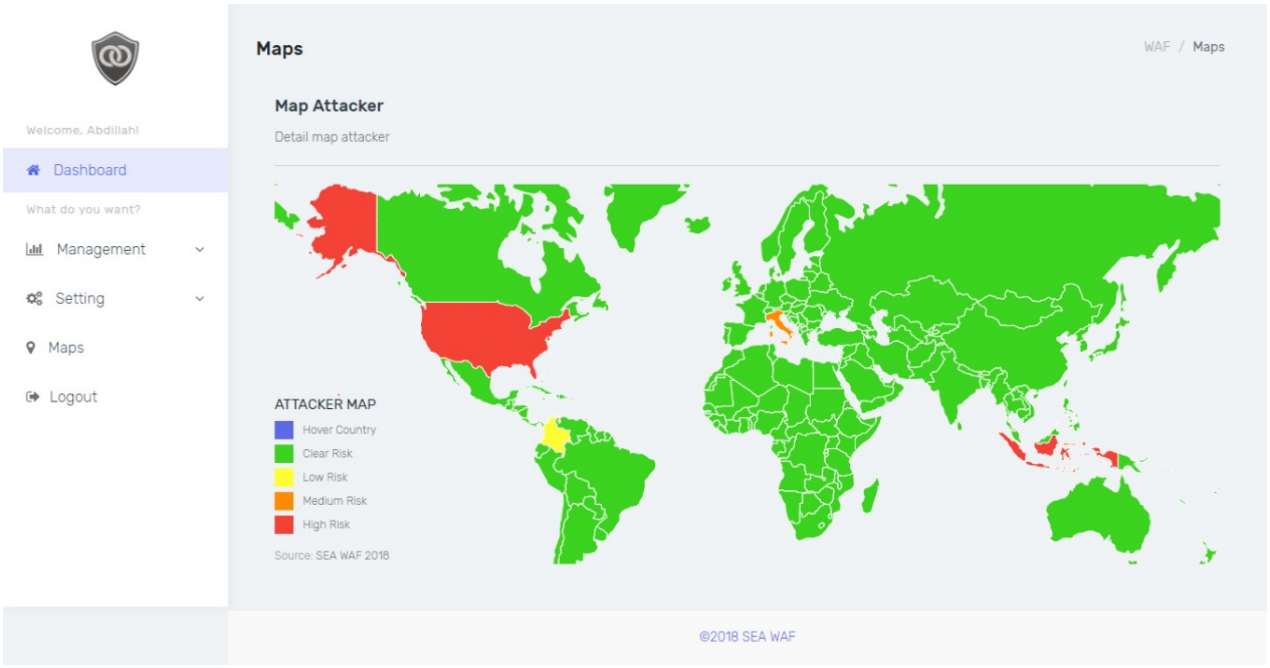

Figure 8: World Map

6. Testing

As part of the research, testing is important activity to see the result of the method. This research uses an e-commerce web called AEShop as targeted web. Based on Figure 9, we can see the web have an URL webdummy.test and it view kind of products.

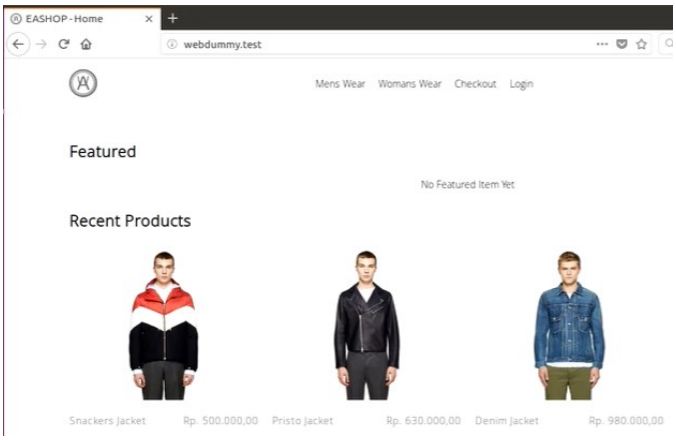

Figure 9: AE Shop

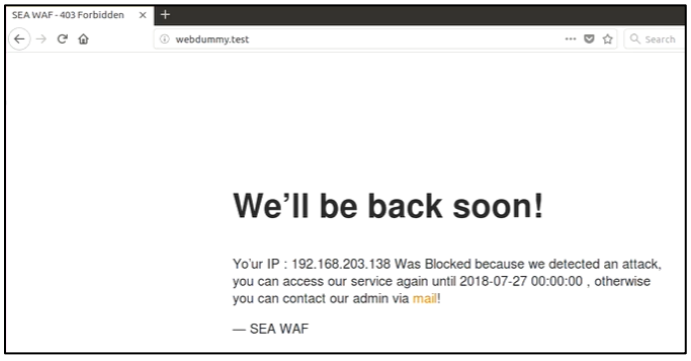

Figure 10: Blocked Message

Then if there is SQL Injection attack from the user, SEA WAF will detect and block it automatically. Then the IP address of the user will be blocked, and user will get the block message (See Figure 10). User cannot access the web until admin remove the block or the block is expired.

By using the burp suit application, the functionality of the SEA WAF application can be seen whether it is able to identify and block SQL injection attacks. The following Table 2 shows the application testing in blocking SQL Injection attacks.

Table 2: SQL Injection attack testing

No	Condition	Result
1	A tautologies-type SQL Injection attack occurred on the AEShop website	The SEA WAF application detect and block the tautologies-type SQL Injection attack
2	A piggy-backed-type SQL Injection attack occurred on the AEShop website	The SEA WAF application detect and block the piggy-backed-type SQL Injection attack
3	A stored procedure-type SQL Injection attack occurred on the AEShop website	The SEA WAF application detect and block the stored procedure-type SQL Injection attack
4	An union-type SQL Injection attack occurred on the AEShop website	The SEA WAF application detect and block the union-type SQL Injection attack
5	A logically incorrect query-type SQL Injection attack occurred on the AEShop website	The SEA WAF application detect and block the logically incorrect query-type SQL Injection attack

Based on the Table 2, the SEA WAF application is able to identify and block five types of SQL Injection attacks, including Tautologies, Logically Incorrect Queries, Piggy Backed Queries, Union Queries dan Stored Procedures. The comparison with the other applications (CloudFlare and Barikode) can be seen in Table 3.

Table 3: SQL Injection attack testing

Attack Type	CloudFlare	Barikode WAF	Proposed Application (SEA WAF)
Tautologies	✓	✓	✓
Logically Incorrect Queries	✓	✓	✓
Piggy Backed Queries	✓		✓
Union Queries			✓
Stored Procedures			✓

As can be seen in Table 3, the proposed application can detect all five types of SQL Injection attacks, while for the other applications (CloudFlare and Barikode), it can only detect certain types of SQL Injection attacks.

7. Conclusion and Future Work

The growth of digital brings increase in web application, as well as the web attack. The security of the web needs to be improved, especially in SQL Injection attack that is the top risk security. SEA WAF bring the solution to secure the web from 5 type of SQL Injection attack automatically. Other features can help administrators to add more security manually such as block by IP or mark any suspicious traffic by Mark as Malicious. With simple and easy to use interface can help administrator or even the normal user to manage the security easily. For future research, this application needs to add more features like notification system to administrator, reporting, and expand for more varieties of injection attacks, such as NoSQL, OS (Operating System), or LDAP injection.

Conflict of Interest

The authors declare no conflict of interest.

Acknowledgment

This research is partially supported by BINUS University.

References

[1] K. C. A. Khanzode and R. D. Sarode, "EVOLUTION OF THE WORLD WIDE WEB: FROM WEB 1.0 TO 6.0," International Journal of Digital Library Services, 1-11, 2016.

[2] P. Dutta, "5 Biggest Data Breaches of 2020 (So Far)," Kratikal, 1 August 2020. [Online]. Available: https://www.kratikal.com/blog/5-biggest-data-breaches-of-2020-so-far/. [Accessed 10 November 2020].

[3] D. Swinhoe, "The 15 biggest data breaches of the 21st century," CSO, 17 April 2020. [Online]. Available: https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html. [Accessed 10 November 2020].

[4] R. A. Katole, S. S. Sherekar and V. M. Thakare, "Detection of SQL Injection Attacks by Removing the Parameter Values of SQL Query," IEEE Xplore Compliant, 736-741, 2018. doi: 10.1109/ICISC.2018.8398896.

[5] "OWASP Top Ten," OWASP, 2020. [Online]. Available: https://owasp.org/www-project-top-ten/. [Accessed 10 November 2020].

[6] L. Ma, D. Zhao, Y. Gao and C. Zhao, "Research on SQL Injection Attack and Prevention Technology Based on Web," International Conference on Computer Network, Electronic and Automation (ICCNEA), 176-179, 2019. doi: 10.1109/ICCNEA.2019.00042.

[7] P. Yaworski, Web Hacking 101, Leanpub, 2016.

[8] S. Pratik and J. Gheewala, "Detection and Prevention of SQL Injection Attacks," International Journal of Engineering Development and Research, **2**(2), 2660-2666, 2014.

[9] R. Devi.D, R.Venkatesan and Raghuraman.K, "A study on SQL injection techniques," International Journal of Pharmacy and Technology, **8**(4), 22405-22415, 2016.

[10] Z. C. S. S. Hlaing and M. Khaing, "A Detection and Prevention Technique on SQL Injection Attacks," in 2020 IEEE Conference on Computer Applications (ICCA), Yangon, Myanmar, 2020. doi: 10.1109/ICCA49400.2020.9022833.

[11] L. Ma, Y. Gao, D. Zhao and C. Zhao, "Research on SQL Injection Attack and Prevention Technology Based on Web," in 2019 International Conference on Computer Network, Electronic and Automation (ICCNEA), Xi'an, China, 2019. doi: 10.1109/ICCNEA.2019.00042.

[12] M. A. Kausar, M. Nasar and A. Moyaid, "SQL Injection Detection and Prevention Techniques in ASP.NET Web Application," International Journal of Recent Technology and Engineering (IJRTE), **8**(3), 7759 - 7766, 2019. doi: 10.35940/ijrte.c6319.098319.

[13] B. Gautam, J. Tripathi and D. S. Singh, "A Secure Coding Approach For Prevention of SQL Injection Attacks," International Journal of Applied Engineering Research, **13**(11), 9874-9880, 2018.

[14] S. V, A. R. Syed and G. E, "The Solutions of SQL Injection Vulnerability in Web Application Security," International Journal of Engineering and Advanced Technology (IJEAT), **8**(6), 3803 - 3808, 2019.

[15] J. Pubal, "Web Application Firewalls," SANS Technology Institute, 1-27, 2015.

[16] PortSwigger Ltd, [Online]. Available: https://portswigger.net/burp. [Accessed 23 November 2020].

[17] "Web Application Firewall," CloudFlare, [Online]. Available: https://www.cloudflare.com/waf/. [Accessed 20 12 2020].

[18] "BARIKODE WAF," Ethic Ninja, [Online]. Available: https://www.ethic.ninja/barikode-waf-produk-solusi-keamanan-website. [Accessed 20 12 2020].