IJE TRANSACTIONS A: Basics Vol. 37 No. 04, (April 2024) 635-645

International Journal of Engineering

Journal Homepage: www.ije.ir

Mitigation from SQL Injection Attacks on Web Server using Open Web Application Security Project Framework

A. Fadlil[a], I. Riadi[b], M. A. Mu'min*[b]

[a] *Department of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia*
[b] *Department of Information System, Universitas Ahmad Dahlan, Yogyakarta, Indonesia*

P A P E R I N F O

Paper history:
Received 20 September 2023
Received in revised form 03 November 2023
Accepted 04 November 2023

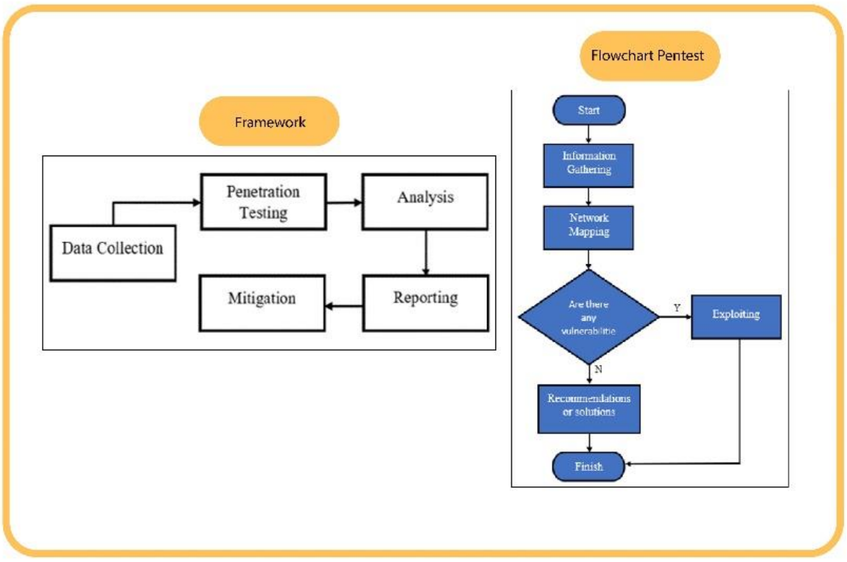
Keywords:
Mitigation
Open Web Application Security Project
Database
Data security
Web server

A B S T R A C T

SQL injection (SQLi) is one of the most common attacks against database servers and has the potential to threaten server services by utilizing SQL commands to change, delete, or falsify data. In this study, researchers tested SQLi attacks against websites using a number of tools, including Whois, SSL Scan, Nmap, Open Web Application Security Project (OWASP) Zap, and SQL Map. Then, researchers identified SQLi vulnerabilities on the tested web server. Next, researchers developed and implemented mitigation measures to protect the website from SQLi attacks. Test results using OWASP Zap identified 14 vulnerabilities, with five of them at a medium level of 35%, seven at a low level of 50%, and two at an informational level of 14%. Meanwhile, testing using SQL Map succeeded in gaining access to the database and username on the web server. The next step in this research is to provide recommendations for installing a firewall on the website as a mitigation measure to reduce the risk of SQLi attacks. The main contribution of this research is the development of a structured methodology to identify and address SQLi vulnerabilities in web servers, which play an important role in maintaining data security and integrity in a rapidly evolving online environment.

doi*: 10.5829/ije.2024.37.04a.06*

Graphical Abstract



*Corresponding Author Email: mumin2008048038@webmail.uad.ac.id (M. A. Mu'min)

1. INTRODUCTION

The rapid development of technology and information has entered all walks of life for various purposes and interests (1, 2). Therefore in recent years, hacking of web servers has dramatically increased (3). This causes more and more companies or organizations to require security in maintaining web servers (4, 5). However, with this development and convenience, it becomes the biggest threat if the security aspect is not given special attention (6, 7). One of the toughest problems faced by web owners is ensuring that the web server is safe from attacks and misuse (8, 9). According to Open Web Application Security Project (OWASP) top 10 (10) SQL injection (SQLi) is one of the most numerous and common attacks that attacks database servers (11) and compromises server services such as: confidentiality, authentication, authorization and integrity (12, 13). This technique is commonly used to exploit web-based applications (14, 15). Injection attacks allow attackers to fake identities, change and delete existing data in the server database by inserting SQL commands into the query string (16, 17). Attackers exploit website application vulnerabilities by entering or injecting special commands into websites that have security holes (18). Improper validation of user input can lead to SQLi attacks (19). Thus, it can cause important user data to be deleted or changed from the web application (20, 21).

In order to maintain security from SQLi attacks, mitigation needs to be implemented as a very important step taken to reduce the risk and impact of SQLi attacks on an application or system (22). The main goal of SQLi mitigation is to prevent attackers from successfully injecting malicious SQL commands into an application or system, so that sensitive data is not exposed or corrupted (23). One effective way of mitigation is to install a firewall layer on the website (24). These firewalls play a key role in protecting websites by implementing specific rules aimed at blocking and eliminating potentially malicious traffic (25). The rules implemented by this firewall are designed to combat various types of attacks, including cross-site scripting and SQL injection (26). With a firewall, websites are better able to detect and respond quickly to SQLi attack attempts that can compromise the integrity and confidentiality of the data stored therein. Thus, implementing a firewall is one of the key components in a mitigation strategy that can help maintain the security and integrity of a website from various SQLi attacks that can arise (27).

Agreindra et al. (8) investigated the Analysis of Web Security Using Open Web Application Security Project 10, their research analyzes and tests web security along with six subdomains with the aim of knowing and assessing the level of security of a website, whether additional security is needed, and recommendations on the website. The results of ther worl showed that the web has a security level of 80%, the web informatics engineering subdomain 60%, information systems 60%, informatics management 60%, integrated academic systems 80%, student acceptance 80% and e-learning 80%.

Alanda et al. (28) introduced the title Mobile Application Security Penetration Testing Based on OWASP, the aim of their research was to determine vulnerabilities and the techniques used to find these vulnerabilities on the Android operating system and Android applications. In addition, it provides recommendations and prevention of these vulnerabilities. The techniques and methods used are based on research from the OWASP Foundation which consists of the 10 main vulnerabilities in Android applications. Results from testing five applications downloaded from the Play Store. Four application based vulnerabilities in the OWASP Mobile Top Ten documentation. OWASP documentation can provide an overview of the most commonly found vulnerabilities in Android applications from the market.

Priyawati et al. (29) investigated Website Vulnerability Testing and Analysis of Internet Management Information System Using OWASP; they have carried out gray box penetration testing techniques using the OWASP method and the OWASP ZAP tool. The test results showed that the target application website has 12 vulnerabilities with details of 8.3% at a high level of vulnerability or 1 warning, 41.7% at a medium level or 5 warnings, 33.3% at a low level or 4 warnings, and 16.7 at information level or 2 warnings. These vulnerabilities relate to matters related to A01-Broken Access Control, A03-Injection, A05-Security Misconfiguration, and A08-Software and Data Integrity Failures.

Alanda et al. (30) studied Web Application Penetration Testing Using SQL Injection Attack, this research randomly tests several websites such as government, school and other commercial sites with several SQL injection attack techniques. Testing was carried out on ten random websites by looking for loopholes to test security using SQL injection attacks. The test results carried out 80% of the websites tested had weaknesses against SQL injection attacks.

In contrast to previous research, the research only focuses on testing and identifying vulnerabilities without carrying out mitigation. In addition, the tools used in previous research did not involve all the stages contained in the OWASP Framework. Therefore, in this research, tools are involved that cover all the stages defined in the OWASP framework, including Information Collection using the Whois tool, SSL Scanning, Network Mapping using Nmap and OWASP Zap, as well as exploiting using

SQL Map. This research has the aim of testing website security against SQLi attacks using various network forensic tools in accordance with the guidelines set out in the OWASP framework and implementing mitigation measures by recommending that the website install a firewall to reduce attacks on the web server. The results of this research are expected to provide valuable guidance for institutions or entities that use websites as information platforms, assisting them in selecting appropriate and effective web security tools.

One of the frameworks used is OWASP, which is a structured framework with several steps in grouping information for security test plans for websites, assessments and verified and analyzed domain reports (31, 32). This framework is for analyzing various vulnerabilities issued by the OWASP Top 10 2021 (33, 34), including: Broken Access Control, Security Misconfiguration, Insecure Deserialization, Injection, Sensitive Data Exposure, XML External Entities, Broken Authentication, Cross Site Scripting, Using Components with Known Vulnerabilities, and Insufficient Logging and Monitoring. There are four tools used to find these vulnerabilities in this framework, namely: Whois, SSL Scan, Nmap, OWASP Zap, and SQL Map. Most tools used to detect vulnerabilities classify vulnerabilities into four categories: High, Medium, Low, and Information (32).

2. MATERIALS AND METHOD

In carrying out penetration testing on web servers, this research uses the OWASP framework to categorize threats and weaknesses in web servers (35, 36). Using the OWASP framework has the benefit of checking web-based application security standards, so that data theft and dissemination by irresponsible parties can be avoided so that it can continue to be used in the long term (37).

The use of the OWASP framework in Penetration Testing on web servers is important because it provides proven security standards, covers various types of threats, is relevant to current challenges, protects data and privacy, and ensures long-term sustainability. The stages in this research can be seen in Figure 1.

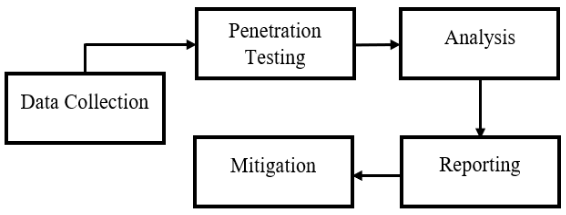


Figure 1. Research stages using the OWASP framework

Data Collection, in this step, collects information about the selected topic and completes the survey. Penetration Testing, at this stage testing is carried out on websites with the aim of security testing carried out by pentesters by imitating actual attacks to damage the security features of an application, system or network so that vulnerabilities are known. There are three stages in carrying out PENTEST, which can be seen in Figure 2. Analysis, at this stage, performs an analysis of the Web Server to find vulnerabilities. Reporting, at this stage describes in detail the results of the analysis that has been processed and writes a report. This mitigation stage involves assessing the website and providing recommendations regarding firewall configurations that need to be implemented to improve web security by preventing injection attacks.

Figure 2. involves the use of the OWASP framework. In the first stage, data is collected using Whois and SSL Scan. Then, in the second stage, network mapping is carried out by carrying out vulnerability scanning using the Nmap and OWASP ZAP tools. Once completed with a comprehensive vulnerability scan, the research proceeds to the third phase, which includes exploitation, using tools to identify SQL vulnerabilities.

The testing process using the OWASP Framework after the completion and implementation stages, the tools used and the functionality of the tools will be explained in Table 1.

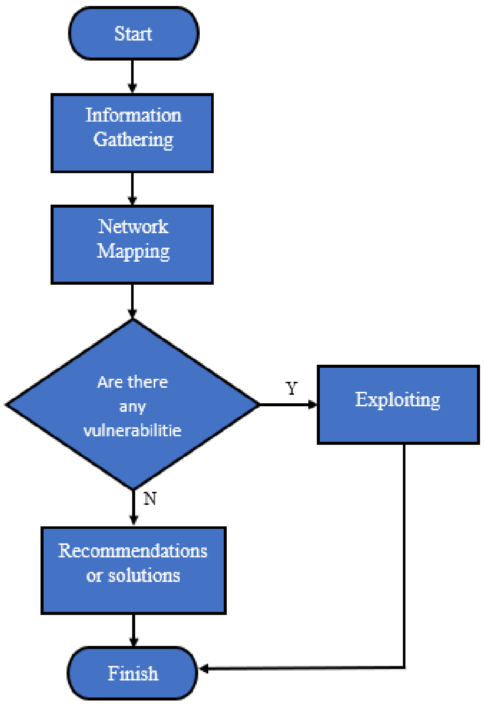


Figure 2. Flowchart of penetration testing stages using the OWASP framework

TABLE 1. Tools in the OWASP Framework

Tools	Information
Whois, and SSL Scan	Used to obtain data about a website.
Nmap	Functions to check the ports on the network.
OWASP Zap	Used to identify potential vulnerabilities in web application testing.
SQL Map	Used to identify the existence of a database on a website.

3. RESULTS AND DISCUSSION

According to the framework established by OWASP, the process of assessing and aggregating the risk level of vulnerabilities on a web server involves several structured stages. This stage includes Data Collection to understand the vulnerability, carrying out Penetration Testing to identify the vulnerability in more depth, analysis the findings found, compiling a Report with a summary of the results and recommendations, and finally implementing the necessary Mitigation steps to reduce the risk of the vulnerability.

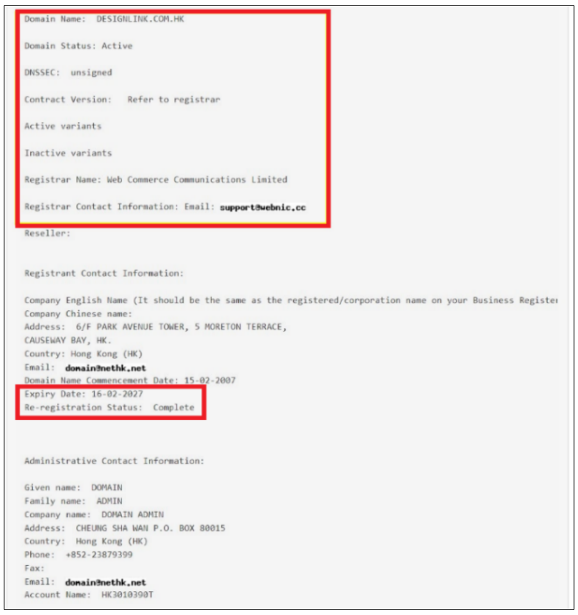
3. 1. Data Collection In this phase, the information collected to support the experiments in this research comes from data found on the website designlink.com.hk. The OWASP framework is used as a tool to identify vulnerabilities contained in the website.

3. 2. Penetration Testing Penetration testing is a testing process carried out on websites to detect potential vulnerabilities, identify poor system configurations, find hardware and software weaknesses, and uncover technical weaknesses in the information system being tested (38). Penetration testing is useful for identifying and addressing vulnerabilities in an infrastructure network, illustrating the level of vulnerability to malicious attacks on a network. There are three main steps in penetration testing, namely Information Gathering, Network Mapping, and exploitation.

3. 2. 1. Information Gathering The purpose of the information gathering is to better understand the web server and any potential issues with it, as shown in Figure 3.

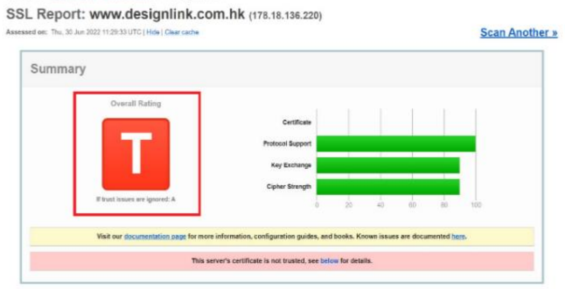
Figure 3 These results come from the process of searching for information via the online Whois tool. This stage aims to collect important data about a web server, including the domain name, registrar name, registrar contact email address, email address, telephone number, country, and related account names. Figure 4. illustrates the outcome of utilizing SSL Scan tools.

Figure 4 shows the use of SSL Scan tool aims to check information about SSL security on a web server;



```
Domain Name:  DESIGNLINK.COM.HK

Domain Status: Active

DNSSEC:  unsigned

Contract Version:   Refer to registrar

Active variants

Inactive variants

Registrar Name: Web Commerce Communications Limited

Registrar Contact Information: Email: support@webnic.cc
Reseller:


Registrant Contact Information:

Company English Name (It should be the same as the registered/corporation name on your Business Registe
Company Chinese name:
Address:  6/F PARK AVENUE TOWER, 5 MORETON TERRACE,
CAUSEWAY BAY, HK.
Country: Hong Kong (HK)
Email:  domain@nethk.net
Domain Name Commencement Date: 15-02-2007
Expiry Date: 16-02-2027
Re-registration Status:  Complete


Administrative Contact Information:

Given name:  DOMAIN
Family name:  ADMIN
Company name:  DOMAIN ADMIN
Address:  CHEUNG SHA WAN P.O. BOX 80015
Country:  Hong Kong (HK)
Phone:  +852-23879399
Fax:
Email:  domain@nethk.net
Account Name:  HK3010390T
```

Figure 3. Results of a search using Whois tools



SSL Report: www.designlink.com.hk (178.18.136.220)
Assessed on: Thu, 30 Jun 2022 11:29:33 UTC | Hide | Clear cache
Scan Another »

Summary

Overall Rating: T
If trust issues are ignored: A

Visit our documentation page for more information, configuration guides, and books. Known issues are documented here.

This server's certificate is not trusted, see below for details.

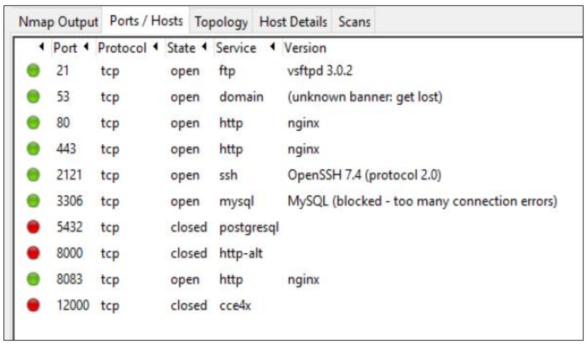
Figure 4. Results of employing tools for online SSL scans

findings indicating inactivity of the SSL security layer on a web server may increase the risk of sensitive data theft, Man-in-the-Middle attacks, vulnerability to credential theft, data integrity risks , and vulnerability to malware attacks.

3. 2. 2. Network Mapping Network mapping is the stage of a web network's mapping process where it is determined which ports have security flaws. The Nmap utility can be used to scan. The outcomes are shown in Figure 5.

Figure 5 depicts the outcome comes from utilizing the Nmap program to perform a port scan on the web server. Ten ports in all were examined based on the scan results. Out of the ten ports, seven are marked as "open," meaning that you can access them; the remaining three are marked as "closed," meaning that you cannot access them.

3. 2. 3. Exploit The data gathered in the first phase can be used as test data in the exploitation phase, which entails testing the identified security flaws. OWASP Zap

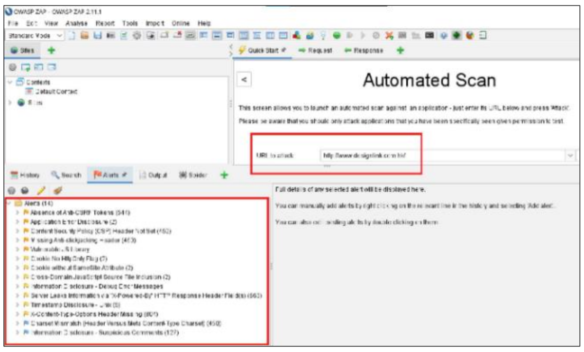


```
Port   Protocol  State   Service     Version
21     tcp       open    ftp         vsftpd 3.0.2
53     tcp       open    domain      (unknown banner: get lost)
80     tcp       open    http        nginx
443    tcp       open    http        nginx
2121   tcp       open    ssh         OpenSSH 7.4 (protocol 2.0)
3306   tcp       open    mysql       MySQL (blocked - too many connection errors)
5432   tcp       closed  postgresql
8000   tcp       closed  http-alt
8083   tcp       open    http        nginx
12000  tcp       closed  cce4x
```

Figure 5. Utilizing Nmap tools results

and SQL Map are two tools that are currently in use in this situation. Figure 6 demonstrates how users can examine the scan results using the OWASP Zap tool.

Figure 6 shows the purpose of this scanning procedure is to perform a thorough assessment of the web server in order to determine the degree of security or vulnerability of the data and information passing through the server. This procedure is also intended to identify potential risks related to the web server and find any threats or vulnerabilities that might be concealed in the setup and configuration of the web server, allowing for the implementation of the necessary mitigating actions to improve security and lower existing risks. The scanning results can be seen in Figure 7.



Automated Scan

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.

Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://www.designlink.com.hk/

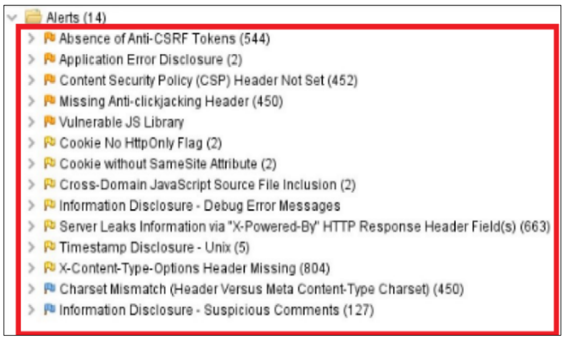
Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.

Alerts (14)
- Absence of Anti-CSRF Tokens (544)
- Application Error Disclosure (2)
- Content Security Policy (CSP) Header Not Set (452)
- Missing Anti-clickjacking Header (450)
- Vulnerable JS Library
- Cookie No HttpOnly Flag (2)
- Cookie without SameSite Attribute (2)
- Cross-Domain JavaScript Source File Inclusion (2)
- Information Disclosure - Debug Error Messages
- Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) (663)
- Timestamp Disclosure - Unix (5)
- X-Content-Type-Options Header Missing (804)
- Charset Mismatch (Header Versus Meta Content-Type Charset) (450)
- Information Disclosure - Suspicious Comments (127)

Figure 6. Technique of scanning using the OWASP Zap tool



Alerts (14)
- Absence of Anti-CSRF Tokens (544)
- Application Error Disclosure (2)
- Content Security Policy (CSP) Header Not Set (452)
- Missing Anti-clickjacking Header (450)
- Vulnerable JS Library
- Cookie No HttpOnly Flag (2)
- Cookie without SameSite Attribute (2)
- Cross-Domain JavaScript Source File Inclusion (2)
- Information Disclosure - Debug Error Messages
- Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) (663)
- Timestamp Disclosure - Unix (5)
- X-Content-Type-Options Header Missing (804)
- Charset Mismatch (Header Versus Meta Content-Type Charset) (450)
- Information Disclosure - Suspicious Comments (127)

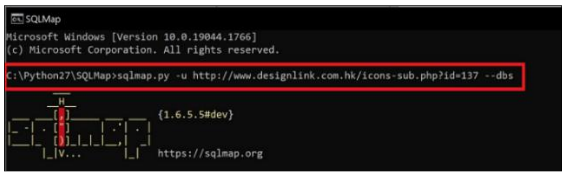
Figure 7. OWASP Zap results of scanning

There were a total of 14 vulnerabilities found in the web server's vulnerability assessment reports (see Figure 7). Five of these vulnerabilities have a moderate risk (medium), seven have a low risk (low), and two have information that is just marginally useful (informational). This scan did not find any vulnerabilities with a high risk level, however it did find a number of medium risk vulnerabilities that needed to be fixed right away to improve server security.

Using SQL Map is the next step in the web server security evaluation procedure. After using OWASP Zap for an initial scan, this SQL Map is used. SQL Map is especially used on web servers to assess security risks linked to SQL. This allows for more thorough testing to be done in order to find possible SQL-related security flaws and take the necessary mitigation steps to improve server security. The SQL Map command used in this procedure is shown in Figure 8.

Figure 8 illustrate an attempt was made to execute SQL statements into the server using the *-u* command to indicate the target URL and the *-dbs* command to query a list of accessible databases in an effort to test the web server's security. Thus, it was effective in locating the *'id'* parameter, which was open to SQL injection. Table 2 contains more details regarding the experiment's outcomes.

Table 2 demonstrates the *designlink_jack* and *information_schema* databases are present on the web server as a result of the *--dbs* command utilizing SQL Map. Web servers are not protected in a way that makes it simple for attackers to get access to such data. Once the database has been located, provide a command to retrieve the table's contents from the database. Figure 9 demonstrates it clearly.

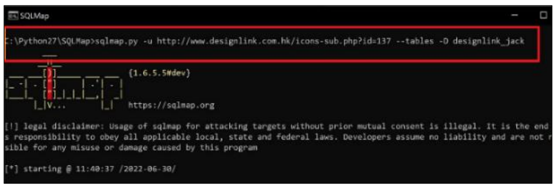


```
Microsoft Windows [Version 10.0.19044.1766]
(c) Microsoft Corporation. All rights reserved.

C:\Python27\SQLMap>sqlmap.py -u http://www.designlink.com.hk/icons-sub.php?id=137 --dbs

                                   {1.6.5.5#dev}

                                   https://sqlmap.org
```

Figure 8. Using SQL Map as a SQL command

TABLE 2. Search results -dbs

No.	Database name
1	designlink_jack
2	information_schema



```
C:\Python27\SQLMap>sqlmap.py -u http://www.designlink.com.hk/icons-sub.php?id=137 --tables -D designlink_jack

                                   {1.6.5.5#dev}

                                   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end
s responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not r
sible for any misuse or damage caused by this program

[*] starting @ 11:40:37 /2022-06-30/
```

Figure 9. Activate the -tables command.

Figure 9 shows the use of *-tables* command on the *designlink_jack* database to access SQL Map tools. This command is used to view the database's contents. The outcomes are shown in Table 3.

Table 3 displays the outcomes of the query after identifying eight tables in the *designlink_jack* database using the *-tables* command. Figure 10 shows a method for calling a table's contents from *dnd_adminlogin*.

Figure 10 shows the *dnd_adminlogin* table's contents can be viewed by using the *-T* command, and its columns' contents can be viewed by using the *-dump* command. The outcomes are shown in Table 4.

Table 4 summarized the performing SQL requests in the SQL Map tool, it can be shown that it was successful in retrieving the data or information present in the web server. If there is insufficient security, this leaves the program open to the web server. Mysqli security makes it very difficult for injection attacks to succeed since the validation is so rigorous that the SQL command that is being injected cannot run.

3. 3. Analysis

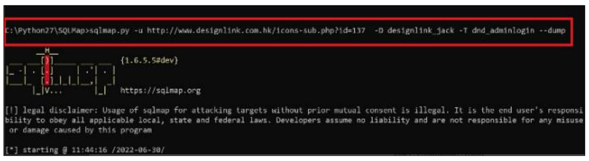
A thorough analysis process will be applied to vulnerabilities found during the earlier phases of web server testing, with a special emphasis on the web server under investigation. The OWASP Framework was used to conduct this analysis, and the results are available in Table 5 and will be thoroughly documented.

Table 5 shows that three ports are listed as "closed" and seven ports are listed as "open" on the website's port scan results. This data offers crucial insight into the security flaws of the website and the degree to which its security systems are ready to thwart unauthorized access attempts. Attackers may attempt to gain access to resources or services on a website by trying to leverage open ports as possible points of entry. A closed port, on the other hand, shows that the security system has effectively controlled access to the port, lowering the likelihood of attacks or penetration that could jeopardize the confidentiality and integrity of data on the website. Website owners will therefore be able to take proactive measures to fortify their security defenses and protect the integrity of their websites from potential threats by having a better understanding of the status of these ports. Table 6 summarized the results of the vulnerability recapitulation.

Based on Table 6. which is depicted in percentage form, namely the risk level of vulnerability at high gets a value of 0% or no vulnerability was found in it, at the medium level it gets a value of 35% (5), namely: Absence of Anti-CSRF Tokens, Application Error Dislosure, Content Security Policy (CSP) Header Not Set, Missing Anti-Clickjacking Header, Vulnerable JS Library. At the low level, it gets a score of 50% (7), namely: Cookie Without Samesite Altribute, Cross-Domain Javascript Source File Inclusion, Information Disclosure-Debug Error Messages, Server Leaks Information Via "X-Powered-By" HTTP Response Header Field(s), Timestamp Disclosure-Unix, X-Content-Type-Options Header Missing. And at the informational level, it gets a value of 14% (2) vulnerabilities, namely: Charset Mismatch (Header Versus Meta Content-Type Charset),

TABLE 3. Search results from the –tables command

Number	Table names
1	Dnd_adminlogin
2	Dnd_blog
3	Dnd_blog_categories
4	Dnd_brands
5	Dnd_contactus



```
C:\Python27\SQLMap>sqlmap.py -u http://www.designlink.com.hk/icons-sub.php?id=137  -D designlink_jack -T dnd_adminlogin --dump
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.6.5.5#dev}
|_ -| . [.]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsi
bility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse
 or damage caused by this program

[*] starting @ 11:44:16 /2022-06-30/
```

Figure 10. Call a table's contents "dnd_adminlogin"

TABLE 4. Table results "dnd_adminlogin"

Id	Username	Password
1	dndlink	SSGBCj6Tuy

TABLE 5. Results of port mapping found

Port	Protocol	Status	Service
21	tcp	Open	ftp
53	tcp	Open	Domain
80	tcp	Open	http
443	tcp	Open	http
2121	tcp	Open	Ssh
3306	tcp	Open	Mysql
5432	tcp	Closed	Postgresql
8000	tcp	Closed	http-alt
8083	tcp	Open	http
12000	tcp	Closed	Ccc4x

TABLE 6. Recapitulation of discovered vulnerabilities

Risk Level	Number of Alerts	Presentase
High	-	0%
Medium	5	35%
Low	7	50%
Informational	2	14%

Information Disclosure-Suspicious Comments. The results of security testing analysis based on the OWASP Top 10 in 2021 are presented in Table 5.

Table 7. The results of system testing using tools such as OWASP ZAP have identified four vulnerabilities that require urgent repair, namely "broken access control," "security misconfiguration," "vulnerable and outdated components," and "software and data integrity failures." Not addressing these vulnerabilities may result in serious risks to the system and associated applications.

Failure to address vulnerabilities such as "Broken Access Control" may result in unauthorized access and privacy breaches, "Security Misconfiguration" may result in exposure of sensitive data and denial of service, "Vulnerable and Outdated Components" increases the risk of vulnerability exploitation and data theft, and"Software and Data Integrity Failures" can result in data loss and reduced service quality and damage data integrity.

Addressing and remediating these vulnerabilities is critical to maintaining system security, integrity, and availability, as well as protecting user data and privacy.

TABLE 7. Vulnerability results on web servers based on OWASP Top 10 2021

Vulnerability	Solution
Broken Access Control	Implement access control system work and reuse it in all applications so as to minimize CORS (Cross Origin Resource Sharing) users, so that users cannot create, read, update or delete records freely, the access control model should limit this by using ownership for each record, and disable the web server directory listing.
Cryptographic Failures	Not found
Injection	Not found
Insecure Design	Not found
Security Misconfiguration	System management can review and update the appropriate configuration for all security notes, updates and patches as part of the patch management process
Vulnerable and Outdated Components	Remove unused dependencies, unnecessary features, components, files and documentation.
Identification and Authentication Failures	Not found
Software and Data Integrity Failures	Use digital signatures or similar mechanisms to verify that software or data comes from the intended source and has not been manipulated
Security Logging and Monitoring Failures	Not found
Server-Side Request Forgery	Not found

That's why fixing these vulnerabilities must be prioritized in information security and application development strategies.

3. 4. Reporting At this stage, take an important step in summarizing the results of the analysis that has been carried out during the OWASP Framework implementation process. This report aims to present the key findings found on the web servers that have been studied. Table 8. presented below, is a representation of research data obtained through the implementation of the OWASP Framework. This report will provide a more detailed and structured picture of the security condition of the web servers that have been analyzed.

Table 8. shows the results of a report on a series of penetration tests and vulnerability analysis carried out using a framework that complies with the security guidelines and standards recommended by OWASP. Through this report, we hope to provide deeper insight into the state of web server security that has been researched, so that appropriate corrective steps can be taken to reduce risks and improve system integrity.

3. 5. Mitigation According to CERT/CC, 90% of security flaws are found during the program development process. This is a result of most developers' ignorance of security-related information and an ineffective strategy for maintaining software security. Web-based assaults are increasing as a result of a number of causes, including poor design, insufficient security requirements, configuration issues, unsafe coding practices, uncontrolled input controls, and password management flaws.

Implementing a framework for "secure software design" is one efficient remedy. But diverse groups of security experts have different perspectives, which makes developing secure software difficult. Network administrators typically concentrate on the use of firewalls, intrusion detection systems, and management systems, whereas software engineers frequently concentrate on aspects of code quality with an eye toward security. The concept of security may be abstracted by software developers from design or architectural principles, security patterns, or other elements.

TABLE 8. Summarize research findings using the OWASP Framework.

Stage	Parameter	Tools	Result
Information Gathering	Domain name, domain status, email expiry date	Whois	Success
	Overall Rating	SSL Scan	Success
Network Mapping	Vulnerability	OWASP Zap	Success
Exploiting	Username and password	SQL Map	Success

It is crucial to keep in mind that developing secure software requires appropriate data security requirements, careful design, meticulous execution, and thorough testing. To achieve effective protection, security must be considered at every stage of the system development life cycle (SDLC). Keep in mind that by offering an additional, comprehensive layer of protection, investing in information security can help lower future expenditures (39, 40).

One step that may be taken to lessen the risk of web attacks is to install a firewall, as depicted in Figure 11. In a defense system, firewalls function as the first line of defense, regulating incoming and outgoing data traffic to guarantee that only approved and permitted traffic may get to the server. As a result, it is a crucial step in enhancing system security and guarding against future cyberattacks and threats.

Figure 11 demonstrates attackers who have access to the Internet attempt to initiate assaults against network-connected websites and web servers. The firewall that had been installed on the network, however, prevented the attack attempt. This firewall acts as a powerful defense against intrusions that could jeopardize the availability, integrity, and confidentiality of the services being offered. Attackers are prevented from accessing or corrupting data kept on web servers and from interfering with the availability of such websites to authorized users when an effective firewall is in place. This adds a crucial layer of security for preserving the network's security and stability as well as safeguarding any sensitive data that might be kept on the web server. In other words, the presence of a firewall has successfully stopped assaults on websites and web servers running on the firewall-protected network. It can be seen in Figure 12 attacks on websites that have firewalls.

Figure 12 shows the research attempts to conduct a SQL Injection (SQLi) attack using the SQL Map tool in order to evaluate the efficacy of the firewall that has been put in place on the website. The experiment's findings demonstrate that, despite careful application, the SQL Map tool is unable to locate the "id" parameter on websites that have firewalls.

These unfavorable outcomes offer an intriguing hint about how reliable the security system that was put in place is. This shows that the firewall that was put in place has been successful in thwarting attempts at SQLi attacks, which typically look for and take advantage of parameters like "id" in an effort to replace or access private information on the internet. Consequently, these results bolster the notion that the security mitigation strategy employed in this study—installing a firewall, for example—can be regarded as a successful means of shielding websites against potentially harmful SQL Injection attacks. Table 7. presents a comparison of research findings from other studies that used the OWASP framework.

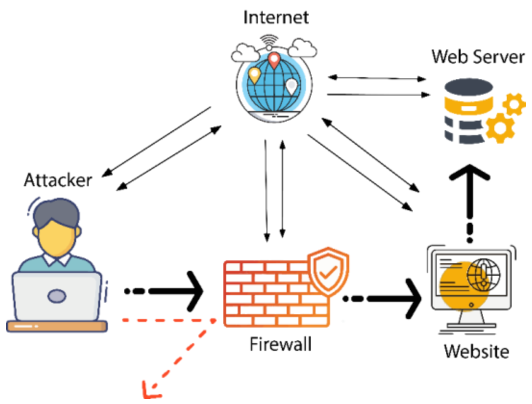
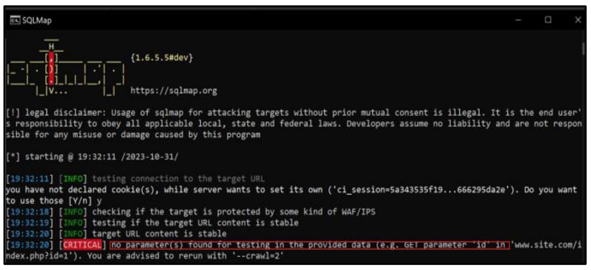


Figure 11. Mitigation of hacker attacks



```
        ___
        __H__
 ___ ___[,]_____ ___ ___  {1.6.5.5#dev}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user'
s responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not respon
sible for any misuse or damage caused by this program

[*] starting @ 19:32:11 /2023-10-31/

[19:32:11] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('ci_session=5a343535f19...666295da2e'). Do you want
to use those [Y/n] y
[19:32:18] [INFO] checking if the target is protected by some kind of WAF/IPS
[19:32:19] [INFO] testing if the target URL content is stable
[19:32:20] [INFO] target URL content is stable
[19:32:20] [CRITICAL] no parameter(s) found for testing in the provided data (e.g. GET parameter 'id' in 'www.site.com/i
ndex.php?id=1'). You are advised to rerun with '--crawl=2'
```
Figure 12. Web server with firewall installed

Table 9. illustrates an interesting comparison various studies regarding the use of the OWASP framework in carrying out SQLi attacks. Different research results show various levels of security gaps on the web. In 2020, a study by Agreindra et al. (8) in testing security obtained a security level of 60%. Then Alanda, et al. (28) in 2020, obtained test results with a vulnerability value of 80%. Priyawati et al. (29) in 2022 found 12 vulnerabilities. In 2021, research conducted by Alanda, et al. (30) found that 80% of the websites tested had weaknesses.

TABLE 9. Comparison with earlier studies

Ref.	Method	Object	Tools	Results	Year
[8]	OWASP	Website	Nmap and OWASP Zap	This website is generally safe from security testing, but some subdomains have a security level of 60%.	2020
[28]	OWASP	Mobile Application	Burp suite	The test results show a vulnerability with a value of 80%	2020

[29]	OWASP	Website	OWASP Zap	The test results show that the target application website has 12 vulnerabilities with details of 8.3% at a high level of vulnerability or 1 warning, 41.7% at a medium level or 5 warnings, 33.3% at a low level or 4 warnings, and 16.7 at information level or 2 warnings.	2022
[30]	OWASP	Website	SQL Map	The test results carried out were 80% of the tested website is vulnerable to SQL injection attacks.	2021
This research	OWASP	Web Servers	Whois, SSL Scan, Nmap, OWASP Zap, and SQL Map	The results of using the Whois tool get a web identity, SSL Scan gets an Overall rating value, Nmap finds three ports with closed status and seven ports with open status, OWASP Zap finds 14 vulnerabilities including; five medium levels, seven low levels, and two information levels, and the SQL Map tool successfully retrieved user names and passwords on the web	2023

This research in 2023 found 14 vulnerabilities including; five medium levels, seven low levels, and two information levels. Then successfully get the username and password on the website. This research also provides a mitigation solution by installing a firewall on the web to protect against injection attacks.

4. CONCLUSIONS

SQLi attacks are one of the main threats that lurk on any web server that is not sufficiently protected. Based on the results of analysis of testing against SQLi attacks using the OWASP framework. The results of using the Whois tool get a web identity, SSL Scan gets an Overall rating value, Nmap finds three ports with closed status and seven ports with open status, OWASP Zap finds 14 vulnerabilities including; five medium levels with a percentage value of 35%, seven at the low level with a percentage value of 50%, and two at the information level with a percentage value of 14%, and the SQL Map tool succeeded in retrieving user names and passwords on the web. This illustrates that the web server does not have adequate security and validation layers to protect itself from various attacks, especially injection attacks. On websites that have a firewall installed, attacks often fail and attempts to find the ID parameter on the web are unsuccessful.

For future research, more in-depth research is needed to reveal mitigation measures other than firewalls in maintaining website security from SQLi attacks. This approach can include the implementation of various frameworks and tools used. Thus, future research will provide deeper and more comprehensive insight into stronger efforts to reduce the risk of attacks on the web.

5. REFERENCES

1. Wang B, Yao Y, Shan S, Li H, Viswanath B, Zheng H, et al., editors. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. 2019 IEEE Symposium on Security and Privacy (SP); 2019: IEEE. 10.1109/SP.2019.00031
2. Bora A, Bezboruah T. Investigation on reliability estimation of loosely coupled software as a service execution using clustered and non-clustered web server. International Journal of Engineering, Transactions A: Basics,. 2020;33(1):75-81. 10.5829/ije.2020.33.01a.09
3. Abdullah HS. Evaluation of open source web application vulnerability scanners. Academic Journal of Nawroz University. 2020;9(1):47-52. 10.25007/ajnu.v9n1a532
4. Sargolzaei A, Yazdani K, Abbaspour A, Crane III CD, Dixon WE. Detection and mitigation of false data injection attacks in networked control systems. IEEE Transactions on Industrial Informatics. 2019;16(6):4281-92. 10.1109/TII.2019.2952067
5. Thepade S, Dindorkar M, Chaudhari P, Bang S. Enhanced face presentation attack prevention employing feature fusion of pre-trained deep convolutional neural network model and thepade's sorted block truncation coding. International Journal of Engineering, Transactions A: Basics,. 2023;36(4):807-16. 10.5829/ije.2023.36.04a.17
6. Thakre S, Bojewar S, editors. Studying the effectiveness of various tools in detecting the protecting mechanisms implemented in web-applications. 2018 International Conference on Inventive Research in Computing Applications (ICIRCA); 2018: IEEE. 10.1109/ICIRCA.2018.8597363
7. Nasiraee H, Ashouri-Talouki M. DoS-Resistant Attribute-Based Encryption in Mobile Cloud Computing with Revocation. International Journal of Engineering, Transactions C: Aspects. 2019;32(9):1290-8. 10.5829/ije.2019.32.09c.09
8. Helmiawan MA, Firmansyah E, Fadil I, Sofivan Y, Mahardika F, Guntara A, editors. Analysis of web security using open web application security project 10. 2020 8th International Conference on Cyber and IT Service Management (CITSM); 2020: IEEE. 10.1109/CITSM50537.2020.9268856
9. Dalai AK, Jena SK. Neutralizing SQL injection attack using server side code modification in web applications. Security and Communication Networks. 2017;2017. 10.1155/2017/3825373
10. Lala SK, Kumar A, Subbulakshmi T, editors. Secure web development using owasp guidelines. 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS); 2021: IEEE. 10.1109/ICICCS51141.2021.9432179
11. Alghawazi M, Alghazzawi D, Alarifi S. Detection of sql injection attack using machine learning techniques: a systematic literature review. Journal of Cybersecurity and Privacy. 2022;2(4):764-77. 10.3390/jcp2040039
12. Kareem FQ, Ameen SY, Salih AA, Ahmed DM, Kak SF, Yasin HM, et al. SQL injection attacks prevention system technology. Asian Journal of Research in Computer Science. 2021;10(3):13-32. 10.9734/ajrcos/2021/v10i330242

13. Chen D, Yan Q, Wu C, Zhao J, editors. Sql injection attack detection and prevention techniques using deep learning. Journal of Physics: Conference Series; 2021: IOP Publishing. 10.1088/1742-6596/1757/1/012055

14. Mokbal FMM, Dan W, Imran A, Jiuchuan L, Akhtar F, Xiaoxi W. MLPXSS: an integrated XSS-based attack detection scheme in web applications using multilayer perceptron technique. IEEE Access. 2019;7:100567-80. 10.1109/ACCESS.2019.2927417

15. Hu J, Zhao W, Cui Y, editors. A survey on sql injection attacks, detection and prevention. Proceedings of the 2020 12th International Conference on Machine Learning and Computing; 2020. 10.1145/3383972.3384028

16. Pattewar T, Patil H, Patil H, Patil N, Taneja M, Wadile T. Detection of SQL injection using machine learning: a survey. Int Res J Eng Technol(IRJET). 2019;6(11):239-46.

17. Alenezi M, Nadeem M, Asif R. SQL injection attacks countermeasures assessments. Indonesian Journal of Electrical Engineering and Computer Science. 2021;21(2):1121-31. 10.11591/ijeecs.v21.i2.pp1121-1131

18. Ravindran U, Potukuchi RV. A Review on Web Application Vulnerability Assessment and Penetration Testing. Review of Computer Engineering Studies. 2022;9(1). 10.18280/rces.090101

19. Syamasudha V, Syed A, Gayatri E. The Solutions of SQL Injection Vulnerability in Web Application Security. no. 2019;6:3803-8. 10.35940/ijeat.F9395.088619

20. Sharma K, Bhatt S. SQL injection attacks-a systematic review. International journal of information and computer security. 2019;11(4-5):493-509. 10.1504/IJICS.2019.101937

21. Akbar M, Ridha MAF. Sql injection and cross site scripting prevention using owasp modsecurity web application firewall. JOIV: International Journal on Informatics Visualization. 2018;2(4):286-92. 10.30630/joiv.2.4.107

22. Jana A, Maity D, editors. Code-based analysis approach to detect and prevent SQL injection attacks. 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT); 2020: IEEE. 10.1109/ICCCNT49239.2020.9225575

23. Alotaibi FM, Vassilakis VG. Toward an SDN-Based Web Application Firewall: Defending against SQL Injection Attacks. Future Internet. 2023;15(5):170. 10.3390/fi15050170

24. Rankothge W, Randeniya M, Samaranayaka V, editors. Identification and mitigation tool for Sql injection attacks (SQLIA). 2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS); 2020: IEEE. 10.1109/ICIIS51140.2020.9342703

25. Xiao M, Guo M. Computer network security and preventive measures in the age of big data. Procedia Computer Science. 2020;166:438-42. 10.1016/j.procs.2020.02.068

26. Krishnan S, Zolkipli MF. Survey on SQL Injection and Cross-Site Scripting Malware Injection Attacks. 10.35629/5252-0502822833

27. Hasan M, Balbahaith Z, Tarique M, editors. Detection of SQL injection attacks: a machine learning approach. 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA); 2019: IEEE. 10.1109/ICECTA48151.2019.8959617

28. Alanda A, Satria D, Mooduto H, Kurniawan B, editors. Mobile application security penetration testing based on OWASP. IOP Conference Series: Materials Science and Engineering; 2020: IOP Publishing. 10.1088/1757-899X/846/1/012036

29. Priyawati D, Rokhmah S, Utomo I. Website Vulnerability Testing and Analysis of Internet Management Information System Using OWASP. International Journal of Computer and Information System (IJCIS) Peer Reviewed-International Journal. 2022;3(03):2745-9659. 10.29040/ijcis.v3i3.90

30. Alanda A, Satria D, Ardhana MI, Dahlan AA, Mooduto HA. Web application penetration testing using SQL Injection attack. JOIV: International Journal on Informatics Visualization. 2021;5(3):320-6. 10.30630/joiv.5.3.470

31. Wiradarma AABA, Sasmita GMA. IT Risk Management Based on ISO 31000 and OWASP Framework using OSINT at the Information Gathering Stage (Case Study: X Company). International Journal of Computer Network and Information Security. 2019;10(12):17. 10.5815/ijcnis.2019.12.03

32. Wijayanto A, Utami E, Prasetio AB, editors. Analysis of Vulnerability Webserver Office Management of Information And Documentation Diskominfo using OWASP Scanner. 2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS); 2020: IEEE. 10.1109/ICORIS50180.2020.9320833

33. Wibowo RM, Sulaksono A. Web vulnerability through Cross Site Scripting (XSS) detection with OWASP security shepherd. Indonesian Journal of Information Systems. 2021;3(2):149-59. 10.24002/ijis.v3i2.4192

34. Ferrara P, Mandal AK, Cortesi A, Spoto F. Static analysis for discovering IoT vulnerabilities. International Journal on Software Tools for Technology Transfer. 2021;23:71-88. 10.1007/s10009-020-00592-x

35. Qian K, Parizi RM, Lo D, editors. Owasp risk analysis driven security requirements specification for secure android mobile software development. 2018 IEEE Conference on Dependable and Secure Computing (DSC); 2018: IEEE. 10.1109/DESEC.2018.8625114

36. Mateo Tudela F, Bermejo Higuera J-R, Bermejo Higuera J, Sicilia Montalvo J-A, Argyros MI. On Combining Static, Dynamic and Interactive Analysis Security Testing Tools to Improve OWASP Top Ten Security Vulnerability Detection in Web Applications. Applied Sciences. 2020;10(24):9119. 10.3390/app10249119

37. Kellezi D, Boegelund C, Meng W. Securing open banking with model-view-controller architecture and OWASP. Wireless communications and mobile computing. 2021;2021:1-13. 10.1155/2021/8028073

38. Ghanem MC, Chen TM. Reinforcement learning for efficient network penetration testing. Information. 2019;11(1):6. 10.3390/info11010006

39. Silva RF, Barbosa R, Bernardino J. Intrusion detection systems for mitigating sql injection attacks: review and state-of-practice. International Journal of Information Security and Privacy (IJISP). 2020;14(2):20-40. 10.4018/IJISP.2020040102

40. Khalaf M, Youssef A, El-Saadany E. Joint detection and mitigation of false data injection attacks in AGC systems. IEEE Transactions on Smart Grid. 2018;10(5):4985-95. 10.1109/TSG.2018.2872120

Persian Abstract

چکیده

تزریق SQL (SQLi) یکی از رایج ترین حملات علیه سرورهای پایگاه داده است و با استفاده از دستورات SQL برای تغییر، حذف یا جعل داده ها، می تواند خدمات سرور را تهدید کند. در این مطالعه، محققان با استفاده از تعدادی ابزار، از جمله Whois، SSL Scan، Nmap، OWASP Zap و SQL Map، حملات SQLi را علیه وب سایت ها آزمایش کردند. سپس، محققان آسیب‌پذیری‌های SQLi را در وب سرور آزمایش‌شده شناسایی کردند. در مرحله بعد، محققان اقدامات کاهشی را برای محافظت از وب سایت در برابر حملات SQLi ایجاد و اجرا کردند. نتایج آزمایش با استفاده از OWASP Zap 14 آسیب پذیری را شناسایی کرد که پنج مورد از آنها در سطح متوسط ۳۵٪، هفت مورد در سطح پایین ۵۰٪ و دو مورد در سطح اطلاعاتی ۱٤٪ بودند. در همین حال، آزمایش با استفاده از SQL Map موفق به دسترسی به پایگاه داده و نام کاربری در وب سرور شد. گام بعدی در این تحقیق ارائه توصیه هایی برای نصب فایروال در وب سایت به عنوان یک اقدام کاهشی برای کاهش خطر حملات SQLi است. سهم اصلی این تحقیق توسعه یک روش ساختار یافته برای شناسایی و رسیدگی به آسیب‌پذیری‌های SQLi در سرورهای وب است که نقش مهمی در حفظ امنیت و یکپارچگی داده‌ها در یک محیط آنلاین به سرعت در حال تکامل دارند.