ISSN: 1978-0524

Vulnerability of injection attacks against the application security of framework based websites open web access security project (OWASP)

Imam Riadi[a,1,*], Rusydi Umar[a,2], Wasito Sukarno[a,3]

[a] Department of Information System, Universitas Ahmad Dahlan, Yogyakarta 55164, Indonesia
[1] *imam.riadi@is.uad.ac.id*;* [2] *rusydi_umar@rocketmail.com;* [3] *wasitomti@gmail.com*
** corresponding author*

ABSTRACT

The development of website applications is currently growing rapidly, but it is not followed by a good security system that can cause the number of security holes that can be entered by the attacker. The number of website applications that are vulnerable to injection attacks to make managers must be aware of and often update and immediately close the security gap. Website applications that have good security will become more secure but the application is still vulnerable to injection attacks. Updating and changing passwords periodically will be better than in fix. Many security hints and risks are released by Open Web Application Security Project (OWASP) TOP 10-2017 as well as a reference in wary of security risks in the application.

Keywords:
Website Applications
Injection
Security
OWASP

I. Introduction

The Internet is a place where Information is very complete and updated. To provide information then the required website. This website contains a database that displays or stores data information, this requires the security of the website to prevent any attacks that steal or delete the database in the website. Website Applications are very vulnerable to various attacks one of them using SQL Injection, if a website there is a gap then the attacker can access to the database.

Overcoming the injection attacks that use SQL Injection method to know the website there is a security hole or not. As well as an application sms broadcast website owned by Bimawa University Ahmad Dahlan 172.10.23.161. Where the website is praised its security hole using SQL Injection, so it will be known earlier if trdapat gap keamnannya, so as not to be entered by the attacker.

Sms broadcast is used to convey messages or information to students so that information terkai activities or announcements can be quickly responded by students. If this broadcast sms is the target of the attacker then the consequences will be detrimental to the student and if the malicious attacker will deceive the students and use this broadcast sms. Testing of sms broadcast application is then used SQL Injection to determine the vulnerability.

II. Literature review

SQL Injection one of the tools used to find or enter a website application by attackers is misused to steal deleting or editing data from a database that causes harm to its owner [1]. Attackers take advantage of website application vulnerabilities by entering or injecting malicious characters into the login form where the application of the website has a security gap [1].

An attack that gets its way in a Web application that has to log in to the entire information system and is likely to get into the control server. This requires patches that can prevent weaknesses, so as to avoid various attacks. Identifying the program code can be done by using Patch, the result is used to fix the patch code that caused the attack. Updating the Program Code periodically to minimize attacks thus website information can be guaranteed [2].

DOI: 10.26555/jifo.v12i2.a8292

Software or hardware can detect a variety of malicious activities in a network system. This can be done by using Intrusion Detection System (IDS), it can detect inbound and outbound traffic in network systems. Passive IDS only detects the presence of an attacker to be informed on webadmin that there is an attack or interruption of the network system [3].

Use of the Open Project Open Web Application (OWASP) framework for Forensic Analysis prevents Cross Site Scripting from having three important stages: Attacking, using OWASP Xenotix XSS Attack Exploit framework v6.2 with single victim method for attack in collecting Keylogger information, Download spoofer and live Webcam screenshots for victims via Mozilla Firefox browser. Analyze, Analyze during live forensics using wireshark, live HTTP Header and TCPDump, with live forensic methods will capture all types of attacks. At this point the hash value file is tested using the app to compare the value of the file integrity downloaded by the victim with the fole that is still stored on the server. The patch process, from the user side is by first installing the extension add-ons in the Mozilla Firefox browser with the name extension XSSFilterAde. XSSFilter will provide early warning, plugin disabled, restricted, possible payload / script on victim when opening website address. The three settings in XSSFilerAde are allowing temporarily for all pages, and allowing malicious blobal scripts [4].

Websites that do not have good security have potential security risks that can be used by attackers. There are ten vulnerabilities contained in the website released by Open Web Application Security Project (OWASP) top 10-2017, the lack of website developers on website security poses a great risk that attackers will exploit to destroy, retrieve or delete databases on the website [5].

Malicious attacks in web applications one of which is SQL Injection, all web applications vulnerable to an attack and the most widely used to attack is SQL Injection. Abuse of SQL Injection security hole is performed by attacker by entering characters into SQL command in web application. SQL commands like Update, Insert, Select, Where and Delete are used by attackers to modify and execute weak SQL code in web applications [5].

The commands used in SQL Injection are Where, Select, Indert, Update, and Delete. SQL code can be changed by the attacker to inject to the Web Application. If the attack can be done and successful then the attacker can access the important data, can change or open the passworded data, the attacker can also damage the application [3].

The methods used in SQL Injection have various ways, the most commonly used by the attackers is by the use of SQL characters such as (,, OR) 1 = - *). SQL Injection is also a variety of types of them are UNION query, Blind SQL, Timing Attack and so on [6][7][8].

The types of SQL Injection attacks are as follows:

- Classical SQL Injection

  UNION method to unify the two queries in informing important data from the database. The attacker must understand the query to be executed to provide important information. Attackers include malicious characters such as single quote tags, double minus and so forth, in order to obtain error messages that will be used to exploit the Website[9].

- Blind SQL Injection

  When the attacker does the injection then succeeds and does not display an error message, but will return to the page itself, displayed some or all of the content or not displaying anything from the website. This method is very long time required because the attacker guess the information contained in the database, and the response is true or false. If manipulated urls produce true then displayed content, if manipulated url produces false then the request will not be processed[9].

- Double Blind SQL Injection / Time-Based

  Merging blind sql injection or classical sql injection with time delay. At the time the url is combined with the delay time generates the command as expected by the attacker then the website can be injected with sql injection[9].

Figure 1 is a release notes for application security that are various weaknesses in the website and handling according to OWASP [10].

OWASP Top 10-2013 (Previous)			OWASP Top 10-2017 (New)		
A1	-	Injection	A1	-	Injection
A2	-	Broken Authentication and Session Management	A2	-	Broken Authentication and Session Management
A3	-	Cross Site Scripting (XSS)	A3	-	Cross Site Scripting (XSS)
A4	-	Insecure Direct Object Reference-Merged with A7	A4	-	Broken Access Control (Original Category in 2003/2004)
A5	-	Security Misconfiguration	A5	-	Security Misconfiguration
A6	-	Sensitive Data Exposure	A6	-	Sensitive Data Exposure
A7	-	Missing Function Level Access Control- Merged with A4	A7	-	Insufficient Attack Protection (NEW)
A8	-	Cross Site Request Forgery (CSRF)	A8	-	Cross Site Request Forgery (CSRF)
A9	-	Using Components with Known Vulnerabilities	A9	-	Using Components with Known Vulnerabilities
A10	-	Unvalidated Redirects and Forwards-Dropped	A10	-	Underprotected APIs (NEW)

Fig. 1. TOP 10 Application Security Risks 2017

Weaknesses in injection, for example SQL injection, OS, LDAP can occur when malicious data is sent into the intepreter that is part of the command or query. The data can manipulate the interpreter to execute non-programmed commands or unauthorized data files.

In Figure 2 there are various attacks, threats, weaknesses and their impact on website applications [10]. Attackers can potentially use a method or technique in an attack against an application that could harm a database. Any action taken by an attacker is a risk that may or may not be of more concern [10].

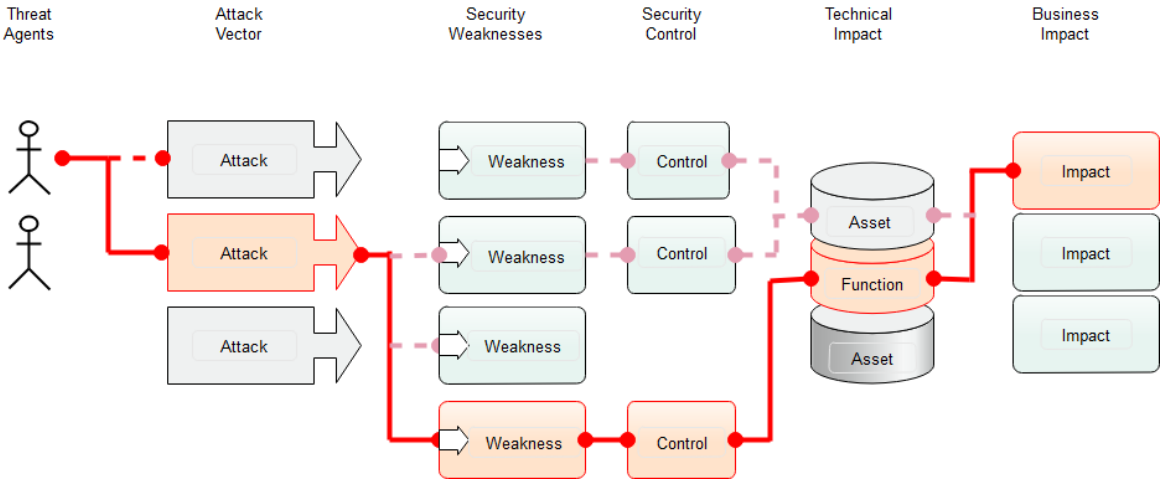

Fig. 2. Application Security Risks

This method is easy to find and exploit but sometimes difficult. Similarly, due to the danger posed. determine the risks to application security risks Evaluate the possibilities associated with each of the threat agents, attack vectors, and security flaws and combination with estimates Technical and business impacts. This factor determines the overall security risk of the Application [10].

OWASP Top 10 that focuses on identifying yamh has the most serious risks for an Organization. at each of these risks, provides a general information on the Possibility and technical impacts of using a simple scoring scheme. Figure 3 provides information Based on the OWASP Risk Assessment Methodology [10].

In each application there is likely to be no threat of attack or technical impact that may change. Therefore, it should be able to evaluate risk-centered threats to security controls and impacts on them. The risks in OWASP TOP 10 stem from the attacks, weaknesses and impacts [10]. Figure 3 [10] describes the Methods in Assessment Application security risks released by OWASP TOP 10-2017.

Threat Agent	Attack Vector	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact

Application Specific	Easy	Widespread	Easy	Severe	Application/ Business Specific
	Average	Common	Average	Moderate	
	Difficult	Uncommon	Difficult	Minor	

Fig. 3. Risk Assessment Methodology of OWASP

A. Vulnerability in website application injection

Figure 4 [10] provides an explanation of the vulnerability of injection attacks. To know that the website application is vulnerable to injection attacks is to verify the use of interpreters to separate malicious data from the query. In some cases it is strongly recommended not to use the interpreter or not to enable it. SQL Call uses bind variables to prepare statements and storage procedures or do not use dynamic commands. Code checking is the fastest and most precise way to know which apps are using interpreters safely. The use of code analysis helps analyze security in finding the use of interpreters and tracking data through applications. Intrusion is tested in order to validate the theme by exploiting it in confirming a vulnerability. Dynamic automation processes test applications that provide information about injection that can be exploited. There is a lack of processing that does not always reach the interpreter and it is difficult to detect the success of injection attacks. Injection weakness is easy to find in poor error handling [10].

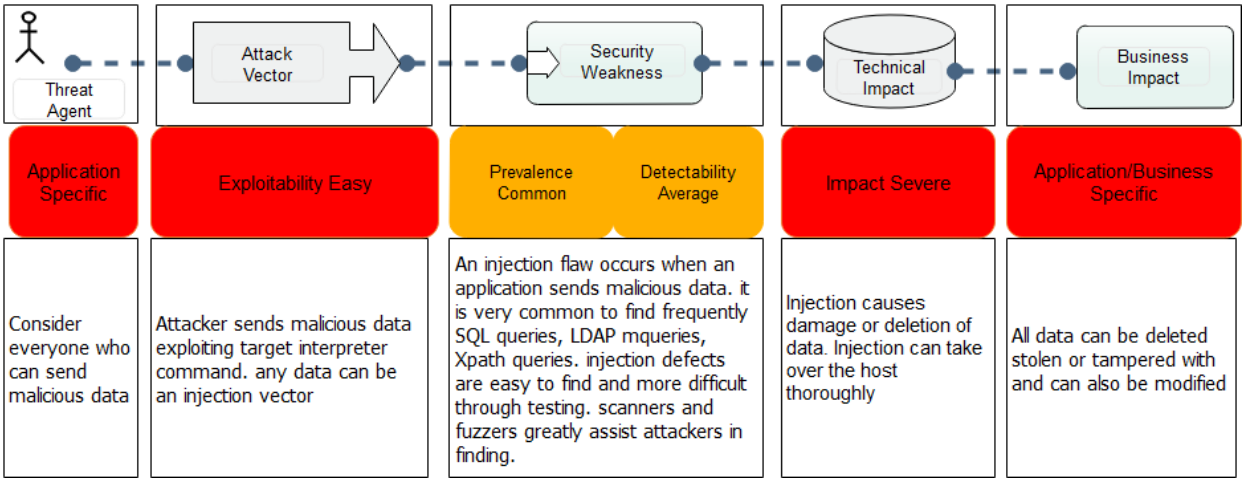

Fig. 4. Vulnerability to injection attacks

Example of Injection Attack Scenario

```
The first scenario
Use of malicious data on SQL Call vulnerability development:
String Query = "SELECT * FROM accounts WHERE custID = '" +
request.getParameter ("id") + "'";
Second Scenario
Similarly, Blind Application frameworks can generate vulnerable
queries (such as for example Hibernate Query Language (HQL);
QueryHQLQuery = session.createQuery ("FROM accounts
WHERE custID = '"+ request.getParameter (" id ") +"' ");
```

In both cases, the attacker can modify the 'id' parameter into the browser sending the code: 'or'1' = '1 as the example:

http://example.com/app/accountview?id= 'or'1' = '1

This can change the meaning of the two queries to return a record from a more dangerous attack table, the data can be modified or the stored procedure can be called [10].

III. Conclusion

Use of a secure Application Programming Interface (API) can avoid interpreters, parameterized interfaces and wary of stored procedures even though they are parameterized but there are still loopholes. In the absence of a parameterized API, be careful in passing specific use-specific characters to pass interpreter commands. Techniques to release from the above are on the OWASP Java Encoder. A positive validation or white list recommendation requires special characters. ESAPI OWASP provides white list input validation.

References

[1] A. Dezfouli, F and Dehghantanha, "Digital forensics trends and future_GERMAN," 2014.

[2] I. Riadi and E. Irawan Aristianto, "An Analysis of Vulnerability Web Against Attack Unrestricted Image File Upload," *Comput. Eng. Appl.*, vol. 5, pp. 19–28, 2016.

[3] D. Mualfah and I. Riadi, "Network Forensics For Detecting Flooding Attack On Web Server," *IJCSIS) Int. J. Comput. Sci. Inf. Secur.*, vol. 15, no. 2, 2017.

[4] A. KURNIAWAN, … I. R.-J. of T. &, and undefined 2017, "Forensic Analysis and Prevent of Cross Site Scripting in Single Victim Attack Using Open Web Application Security Project (Owasp)," *Search.Ebscohost.Com*, vol. 95, no. 6, pp. 1363–1371, 2017 [Online]. Available: http://search.ebscohost.com/login.aspx?direct=true&profile=ehost&scope=site&authtype=crawler&jrnl=19928645&asa=Y&AN=122292244&h=aFsBDtt9GmkqRxlSVKE4BMibaY1SQLWLEI7kSOfqm9b5HTS0AThxsxW7o2gIAPKxfYepAIcWRHbYugWXoNojHg%3D%3D&crl=c

[5] A. Sagala, E. Manurung, B. Siahaan, and R. Marpaung, "Deteksi , Identifikasi Dan Penanganan Web Menggunakan Sql Injection Dan Cross-Site Scripting," *Inst. Teknol. Del*, vol. 2014, pp. 20–24, 2014.

[6] R. Ellysa, M. Husni, and A. Pratomo, "Pendeteksi Serangan SQL Injection Menggunakan Algoritma SQL Injection Free Secure pada Aplikasi Web," *Tek. Pomits*, vol. 2, no. 1, pp. 1–6, 2013.

[7] W. G. J. Halfond and A. Orso, "AMNESIA: Analysis and Monitoring for NEutralizing SQL-injection Attacks," *Proc. 20th IEEE/ACM Int. Conf. Autom. Softw. Eng.*, pp. 174–183, 2005 [Online]. Available: http://doi.acm.org/10.1145/1101908.1101935

[8] OWASP, "SQL Injection," 2009. [Online]. Available: https://www.owasp.org/index.php/SQL_Injection

[9] I. Riadi, R. Umar, and W. Sukarno, "Analisis Forensik Serangan Sql Injection Menggunakan Metode Statis Forensik," *Pros. Interdiscip. Postgrad. Student Conf. 1st*, vol. I, no. I, pp. 102–103, 2016.

[10] "Open Web Application Security Project (OWASP) TOP 10 Application Security Risks." [Online]. Available: https://www.owasp.org/index.php/Top_10_2017-Top_10.