Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 3 (2011) 453–458

Procedia Computer Science

www.elsevier.com/locate/procedia

WCIT-2010

SQL-injection vulnerability scanning tool for automatic creation of SQL-injection attacks

Abdul Bashah Mat Ali [a] *, Ala' Yaseen Ibrahim Shakhatreh[b], Mohd Syazwan Abdullah[c], Jasem Alostad[d]

[a,b,c]*Division of Applied Sciences, College of Arts and Sciences, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia*
[d]*The Public Authority for Applied Education and Training (PAAET), College of Business, P.O. Box.23167, Safat 13092, Kuwait*

Abstract

Securing the web against frequent cyber attacks is a big concern as attackers usually intend to snitch private information, financial information, deface and damages websites to prove their hacking capabilities. This type of vandalism may drive many corporations that conduct their business through the web to suffer financial and reputation damages. One of the most dangerous cyber attacks is the Structured Query Language (SQL)-injection attack, whereby this type of attack can be launched through the web browsers. The vulnerability of SQL-injection attack can be attributed to inappropriate programming practice by the website developers, which leaves a lot of doors widely open for the attackers to exploit these and gaining access to confidential information that resides in the website server databases. In order to address this vulnerability, it must be feasible to detect the vulnerability and enhance the coding structure of the website to avoid being an easy victim to this type of cyber attacks. Detecting the SQL-injection vulnerability requires the development of a powerful tool that can automatically create SQL-injection attacks using efficient features (different attacking patters) to detect the vulnerability of the websites. This paper discuss the development of a new web scanning (MySQLIInjector) tool with enhanced features that will be able to conduct efficient penetration test on PHP (*started as Personal Home Page but now widely used as Hypertext Preprocesses*) based websites to detect SQL injection vulnerabilities. This tool will automate the penetration test process, to make it easy even for those who are not aware familiar about hacking techniques.

Keywords: Database Security; SQL Injection; SQL Vulnerability;

1. Background

Penetration testing or web auditing is one of the most important topics that security researchers are concerned about. It aims to prove the effectiveness of the website security system because application level attacks rank at the top of nowadays cyber attacks as these are preferred by attackers/hackers. The philosophy behind web auditing is to ensure having a single entry point to web applications by performing penetration tests represented by conducting sophisticated attacks on websites. Having more than one entry point to the system will be considered as a security flaw that attracts potential hackers to exploit it. Moreover, penetration testing covers checking against a wide range of web vulnerabilities which are related to web application level vulnerabilities such as cross-site-scripting (XSS),

* Abdul Bashah Mat Ali. Tel.: +0649284730; fax: +06049284753.
E-mail address: bashah@uum.edu.my.

1877-0509 © 2010 Published by Elsevier Ltd.
doi:10.1016/j.procs.2010.12.076

SQL injection, IFRAME (Inline Frame) flaws, Domain Name Server (DNS) attacks, web authentication flaws, remote code execution, and remote file inclusion [1]. Exploiting any one of these vulnerabilities may enable remote attackers to gain administrative access to the infected website which enable them having the control to deface, damage and snitch credentials [2].

Penetration testing is recommended for those critical or popular websites as it is similar to trying to break into the organization's IT systems. The aim of penetration testing is to demonstrate the robustness of the security system in order to expose the vulnerabilities and giving advice on how to recover these flaws [3]. Conducting penetration testing is an essential requirement for organizations that deal with critical or huge amount of data that may belong to hundreds and thousands of clients through an automated system or a website [4]. One of the most dangerous attacks that should be recovered in the penetration testing is the SQL injection attack that injects a malicious Javascript or HTML tags into the victim's website database, or in other words executing malicious SQL queries to inject malicious HTML tags [5,6,1], which is considered as IFRAME attack. On the other hand IFRAME attack can also be carried out through what is known as cross site scripting attack (XSS) as this kind of attack can embed one malicious website in the original website to snitch credentials and to download malware on the visitor's computer [4,2,1].

Web applications usually interact with backend databases and when the application receives a request from the user, it fetches the database by generating and executing SQL queries to interact with the relational database. These queries look for the requested data to be displayed in generated HTML pages to the user. In this normal scenario, the user inputs are treated as lexical entities. However, when the user inserts unexpected inputs that are not addressed in the web application's dictionary, they will lead the web server to react abnormally. This may cause the web application to display unexpected data which may be classified as confidential and may be useful for the attacker. This is known as command injection attack, and such commands could be SQL queries or operating system commands or Javascript & HTML tags. Command injection attack is wide spread and more commands injection vulnerabilities will be discovered in the future [7,5,8]

2. SQL Injection Vulnerability Problems

SQL injection vulnerability results from the fact that most web application developers do not apply user input validation and they are not aware about the consequences of such practices [6]. This inappropriate programming practices enable the attackers to trick the system by executing malicious SQL commands to manipulate the backend database [6, 1]. One of the most important properties of SQL injection attack is that it is easy to be launched and difficult to be avoided. These factors make this kind of attack preferred by most cyber criminals, and it is getting more attention in the recent years [1]. Furthermore, the available scanning tools have limited features in shaping efficient attacking patterns which are required to detect hidden SQL injection vulnerability [6, 1]. Moreover, the available scanning tools use brute force techniques to extract data from the targeted websites. These tools do not show meaningful and detailed information about the detected vulnerability. Obtaining this critical detailed information would be very useful for web developers who are not aware about hacking techniques in helping them to fix the bugs, thus to eliminating these vulnerabilities. The lack of penetration testing scanning tools built with enhanced features that are able to conduct efficient penetration test is the main problem addressed in this study. Therefore, this study aim is in developing a web scanning tool with enhanced features to detect SQL injection vulnerabilities of website databases using different types of attacking patterns, vectors and modes in shaping the attacks. The SQL injection vulnerability problems can be addressed by developing a new web scanning (MySQLIInjector) tool with enhanced features that is able to conduct efficient penetration test on PHP based websites to detect SQL injection vulnerabilities and getting the web developers to fix these vulnerabilities . The development of the tool is based on the following steps: (1) review the literature on current penetration testing tools for SQL injection attacks on databases for identifying the important types of attacking patterns, vectors and modes; (2) develop penetration testing styles for SQL injection attack on databases based on the different types of attacking patterns, vectors and modes based on information from the literature; (3) develop a web scanning tool (MySQLInjector) to detect SQL injection vulnerabilities based on the identified styles; and (4) test and validate the MySQLInjector tools for penetration testing on different websites.

3. SQL Injector Tool Development Methodology

The development of the new web scanning tool (MySQLInjector) to detect SQL injection vulnerabilities was based on the Rational Unified Process (RUP) for secure software systems development proposed by Heijstek & Choudron (2008). RUP methodology has four phases and these are Inception, Elaboration, Construction, and Transition. The first phase is the inception phase that emphasizes on business modeling and requirements gathering in the beginning of the software life cycle. In this study, the first phase focuses on various SQL injection vulnerability which was gathered and analysed from the systems development perspective in view to develop a web scanning tool that can detect SQL injection vulnerabilities in web applications. The other step in the first phase is requirements gathering, which was conducted by studying different web frameworks and platforms from different vendors supporting web development programming languages and environments to understand their structure. The second phase is the elaboration phase that translate the tool requirements gathered from the first phase and the SQL injection attacking model requirements to software design models. This was carried out by analyzing the gathered requirements and developing the initial prototype of the tool. The third phase is the construction phase, where the initial prototype tool was developed to be fully operable based on the tool requirements and design. During this stage, the tool was tested throughout the development activities to ensure its correctness and compliance to the SQL injection attacking model. The fourth and the last phase is the transition phase, which emphasizes on deploying the tool for final testing on the hosting computer or the environment that the tool is designed to operate, which are the web servers. The tool was tested on attacking different websites to ensure that the tool is capable of exposing the SQL injection vulnerability that must be fixed. These websites are operational websites that are used commercially, thus the testing was on real life attacking of the database servers.

4. MySQLInjector Tool

MySQLInjector is new scanning tool that is capable of conducting efficient penetration tests on PHP based websites to detect the hidden SQL injection vulnerabilities. The new tool involves more variables and features in the scanning process, where a wide range of attacking vectors is used in exposing the vulnerability in the targeted websites. These variables and features are due to the varieties of the available frameworks used in building websites and the variety of database structures used in the development process. This is challenging as conducting an efficient penetration test, especially automating the penetration testing process to innovate a new scanning tool. The most important aspects of the attacking patterns used in the tool are the ability to trick the web server to display error notifications if it was inappropriately programmed. There are a total of 10 such attacking patterns in this tool. If one pattern fails to expose the vulnerability, the others will succeed in tricking the web server if it is vulnerable. Moreover, this strategy will not leave any doubt that there is a possible vulnerability without exposing it. Furthermore, injecting the attacking patterns comes in in different levels, the first level is to check the web path against SQL injection vulnerability, and the second level is to being sure that the website is vulnerable, the injection of the other type of attacking vectors starts to take place to check the database version of MYSQL using normal and encoded attacking patterns based on true/false responses and to analyze the website against further patches and protections such as forbidden protection and filters to prevent the attackers to execute SQL commands, and to check whether these protections can be broken or not, and is performed by injecting the website with the third level attacking pattern.

After checking the web server against the existence of the protections, MySQLInjector tries to predict the number of infected columns in the database by injecting the fourth stage attacks, where these defective columns can be exploited by the attackers by executing several SQL queries through them to extract the data from the database to be shown on the web page. The prediction process may take few seconds because MySQLInjector intends to calculate the byte size of each HTML page source after injecting the pattern using HTML parser technique and it starts comparing the results to locate the nearest one to the original page source that is always a true response. This is followed by, and then adding the number of defective columns to the URL, which will return a true value or will always return a false value or a error notification. Then, the MySQLInjector tool will identify the possible

vulnerabilities in the database of the website, which is considered as a serious security flaw using the number of defective columns. MySQLInjector incorporates valuable features in shaping the attacking vectors in order to increase the ability to trick the web server and to utilize the response of the web server after injecting the attacking patterns to expose the SQL injection vulnerability. These features are blind SQL injection using true/false response, blind SQL injection using true/error response, and blind SQL injection using order by, which are not used in the scanning tools listed in table 1. Furthermore, using wide variety of attacking vectors and types that were shaped depending on the three motioned features in multiple levels with the high sensitivity of HTML parser in utilizing and analyzing the web server responses will certainly expose the hidden SQL injection vulnerabilities. Moreover, using MySQLInjector is not considered as an additional load on the web server because it is operated remotely as a client side tool, and the use of MySQLInjector is not conducted frequently as the Intrusion Detection Systems (IDS). IDS systems are executed in the run time in every web page request and considered as server side code. The table 2.1 below shows the mentioned scanning tools with their features and properties.

Table 1. SQL Scanning Tools Features

	Features								
Scanning Tools	Variety of Attacking Patterns and Vectors	Classical SQL Injection	Blind SQL Injection based on True/False Response	Blind SQL Injection based on True/Error Response	Blind SQL Injection based on Order by	Provide detailed info	Cross site scripting XSS attack	Second order XSS attack	Run in the real-time (Server side)
SecuBat	√	√					√		
SQL-IDS									√
SAFELI									√
ARDILLA	√	√					√	√	
AMNESIA									√
MySQLInjector	√		√	√	√	√			

Several studies by [9, 6, 1, 5] have been conducted to facilitate detecting and measuring web vulnerabilities such as SQL injection vulnerability and have proposed tools and techniques for these purposes in order to help securing the web from serious threats. Some of these tools are to detect SQL injection vulnerabilities such as SecuBat, ARDILLA tools. On the other hand, some tools are used to detect SQL injection attacks in the real time, such as SQL-IDS, SAFELI and AMNESIA techniques.

Figure 1,2, and 3 show how SQL attacks can be done manually. Error in the website can be generated (see Fig. 1) when submitting order by 14. It seems that the website is vulnerable to SQL injection attack, and now it is time to see how much data can be collected from the infected website. Extracting the number of infected columns in the database is good enough to manipulate the backend database and will enable the penetration tester to gain more information about database version, database name, the system user, tables and columns names. In order to get the number of infected columns, the order by attack is required to be applied. The order by value will be increased until a false value returned as shown in Fig 1. Here, it is easy to know that the number of defected columns is 13, because before reaching the value 14 it was 13 and the page was loading normally. Now, using the *union* with *select* on all statements on all the infected columns will enable the penetration tester to extract critical information about the website, this will return true value and the page will load normally. As shown in Fig.2, using the *union* statement will allow appending other SQL statement where the website executes select statement for the variable id and another select statement is appended by penetration tester using union statement. Now to know the column that data can be read from, a "-" minus operator is added before the value of the variable id. More data such as the user of the server and the database name can be extracted by replacing VERSION() or USER() or with DATABASE() with one of the infected column. Furthermore, database version 5 can be exploited through using information schema as the default schema for the database, to extract the tables' names and columns names as shown in Fig. 3. Now automating the penetration test process by using MySQLInjector will save time and effort and does not need a

knowledgeable penetration tester. The only thing that MySQLInjector user needs is a suspected URL and it starts shaping attacking patterns and injecting them. Once the attack is completed, it will return the results, as shown in Fig. 4.

Fig.1. Generating Error When Submitting Order By With 14

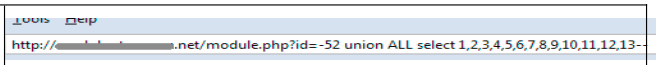

Fig.2. Exposing the Infected Columns

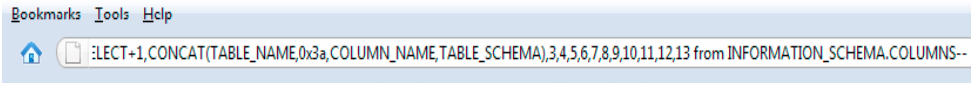

Fig.3. Revealing the Structure of Tables and Columns.

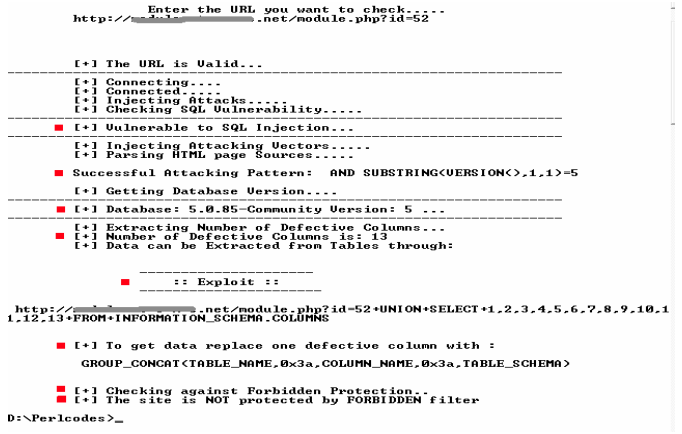

Fig.4. Conducting Penetration Test Using MySQLInjector to Detect SQL Injection Vulnerability

5. Conclusion and Future Work

MySQLInjector is new scanning tool that is capable of conducting efficient penetration tests on PHP based websites to detect the hidden SQL vulnerabilities in web server databases. The tools has a combination of attacking patterns, vectors and modes that allows web developers that are illiterate about hacking techniques in conducting penetration testing on their web database servers. The future wok in this area involves expanding the tools capability in conducting penetration test on Active Server Pages (ASP), and Java Server Pages (JSP) based websites.

References

1. Kiezun, A., Guo, P. J., Jayaraman, K., & Ernst, M. D. (2009). Automatic Creation of SQL Injection and Cross-Site Scripting Attacks. *ICSE '09*. 199-209. Vancouver, Canada.
2. Wright, C., Freedman, B., & Liu, D. (2008). *The IT Regulatory and Standards Compliance Handbook.* Burlington, MA, USA: Syngress.
3. Midian, P. (2003). How to ensure effective penetration test. *Information Security Technical Report*, 8(4), 65-77.

4. Basta, A., & Halton, W. (2008). *Computer Security and Penetration Testing*. USA: Thomson Course Technology.
5. Halfond, W. G. J., & Orso, A. (2005). EMNESIA: Analysis and Monitoring for Neutralizing SQL-Injection Attacks. *ASE '05*, 174-183. Long Beach, California, USA.
6. Kemalis, K., & Tzouramanis, T. (2008). SQL-IDS: A Specification-based Approach for SQL-Injection Detection. SAC '08. 2153-2158. Fertaleza, Ceara, Brazil.
7. Newson, A. (2005). Network Threats and Vulnerability Scanner, *Network Security*, pp. 13-15.
8. Su, Z., & Wassermann, G. (2006, January 11). The Essence of Command Injection Attack in Web Applications. *POPL '06*, 372-382, Charleston, South California, USA.
9. Kals, S., Kirda, E., Kruegel, C., & Jovanovic, N. (2006). SecuBat: A Web Vulnerability Scanner. International World Wide Web Conference Committee IW3C2, 2, pp. 247-256, Edinburgh, Scotland.