INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

USING HASH ALGORITHM TO DETECT SQL INJECTION VULNERABILITY

***Maki Mahdi, Ahmed Hashim Mohammad**

*Al-Nisour University College/Department of Computer Technical Engineering/Baghdad-Iraq
*E-mail:- Dr.MakiMahdi@yahoo.com

Abstract: -The web application has become an important part of our lives and become of the popularity of the web increases, any web security vulnerability are easy to understand and to avoid, As a result will mostly be observed and be exploited by hackers. Web application security issues are mostly related to malicious input data and Web forms are the main interface to input these data. The class of vulnerability Know as SQL Injection continues to present an extremely high risk in the web application. Many traditional web security scanners that depend on pattern matching and Similarity have low pages coverage and cannot detect the all SQL injection vulnerabilities Exist in web. To overcome a problem of existing black box testing this research is focused on providing clear, simple, actionable guidance for scanning SQL Injection vulnerability in Web application by design anew black box testing analyzer which based on calculate the hash value for web response.
In addition, to verify the proposed analyzer method was conducted by applying the framework to running 4 web application scanners against packed five vulnerable web applications, Experiment result shows that this method can automatically and effectively discover SQL injection web application vulnerabilities .

Key words: SQL, SQL Injection Vulnerability

1.1 Introduction

The word wide web is largely based on Hypertext transfer protocol (HTTP), specifying the format of message exchanged by a client called in this case the user agent, and a server .the request format most familiar to internet users is what is commonly called a uniform resource locator (URL), when user center a URL into the browser window, the browser first checks the scheme of the URL to determine the protocol . the term "web application" terms like web application, web site, web-based system, web-based software or simply web and all may have
the same meaning. While web server is just a piece of software running on the operating of a server that allows connection to access a web application the most common web server are Internet Information Server (IIS) on the Windows server and Apache Hypertext Transfer Protocol (HTTP) Server on a Linux server. These servers have normal directory structure like any other computer, and it's these directories that house the web application.
Web application is often vulnerable to attacks, which can give attackers easily access to the application's underlying database. SQL injection attack occurs when a malicious user, through specifically crafted input, causes a web application to generate and send a query that function differently than the programmer intended. The next section (Section 2) describes a Security testing to emphasize the importance of testing, and it also explains security testing strategies. In section 3, introduce web application attack and how SQL injecting working, proposed system show deduction methods in section 4, section 5 the evaluation of model finally, section 6 concludes the paper.

1.2 Security Testing

Security testing as part of a quality assurance process is responsible for the discovery and correction of these types of flaws created during the implementation and design of the software. Software vulnerability analysis is the art and science of discovering security problems or other weaknesses in software or system. Vulnerabilities in software that is introduced by mistake or poor practices is serious problem. With most software today is that it contains numerous flaws and errors that are often located and exploited by attackers to compromise the software's security and other required properties. Due to the proliferation of commercial applications for use on the internet safety and security issues have taken on additional importance. If internet users believe that their personal information is not secure and is available to those with intent to do harm, the future of e-commerce is peril. Designing and testing software system to insure that they are safe and secure is a big issue facing software developers and test specialists. Security testing evaluates system characteristics that relate to the availability, integrity and confidentially of data and services server as a baseline for security requirements.

1.2.1 Security Testing Strategies

Various people organization around the world audit software. They may do this for quality assurance reasons, as third-party auditors, or as hackers' looking to find bugs for fun or for profit Security testing encompasses several strategies, there are there main approaches to testing software application for the presence of bugs and vulnerabilities:

1. Black Box Testing

One important testing strategy is black-box testing (also known as, data driven or input/output-driven testing) refers to testing that only operate by launching attacks against on application and observing it's response to these attacks. To use this method, view the program as a black box, the tester only knows the inputs that can be given to the system which comes into contact with the test item a program or program unit only.

White-Box Testing

Another testing strategy, white-box testing as the name implies, that the tester has access to the application's source code and therefore has significant knowledge of the internals of the application under test. It is easier to make sure the program will function as intended, as it is possible to directly test conditionals in the program flow. With white box testing it is possible to craft input that allows every line of code to be executed and tested. From a security perspective, it is important in white box testing to provide the program with input the program normally would not expect in order to see the program deals with this input.

2. Gray-Box testing (functional and structural)

Gray-Box testing is a combination of these two approaches and is usually applied during system testing. Gray Box testing, like Black Box testing, is concerned with the accuracy and dependability of the input and output of the program or module. However, the test cases, risk assessment, and test method involved in Grey Box testing are developed based on attempts to extract information about the application from the executable code by disassembling the binary files or even reverse engineering the application.

1.3 SQL Injection Vulnerabilities

SQL injection is the most common attack vector because of the popularity of relational database that speak SQL. Many web application both large and small, use database to store the various kinds of information that it needs in order to operate. For example, a web application deployed by an online retailer might use a database to store user accounts, credentials, and personal information. SQL injection attacks use command sequences from structured Query Language (SQL) statements to control database data directly. Application often use SQL statements to users to the application, validate roles and access levels, store and abstain information for the application and user, and link to other data sources. In most cases database are accessed using structured query languages (SQL). SQL is a universal language suitable for all databases. However it's syntax can be slightly different in different types of database servers. The availability of these systems and the sensitivity of the data that they store and process are becoming critical to almost all major businesses, not just those that have online e-commerce store. SQL injection is one of the most devastating vulnerabilities to impact a business, as it can lead to exposure of all of the sensitive information stored in an application's database, including handy information such as usernames, password, names addresses, phone number, and credit card details. Discover of SQL inject bugs bringer necessary information to

further attack. Before SQL attack, the attackers need to identify the aim database platform and decide what SQL attack statements or methods should utilize. SQL injection

Attack can allow an attacker to do the following:

* Log in the application without supplying valid credentials.
* Perform queries against data in the database, even to which the application would not normally have access.
* Modify the database contents, or erase the database altogether
*Use the trust relationships established between web application components to access other database

SQL injection attacks work because the application does not properly validate input before passing it to an SQL statement. For example, the following SQL statement:

[1] SELECT * FROM table name WHERE User ID =2302 becomes the following with a simple SQL injection attack.

SELECT * FROM table name WHERE User ID = 2302 OR 1=1

The expression "OR 1=1" evaluates to the value "TRUE", often allowing the enumeration of all User ID values from the database. SQL injection attacks can be entered from the address bar, from within application fields, and through queries and searches

1.4 Proposed System

After each requesting process of payload, the scanner monitors HTTP response status code that provides information about the status of the request, if status code is 5xx, we will get an error message. If status code is 4xx, request source is invalid. If status cod 3xx, we need do further work for redirection. If status code is 1xx, we will wait. A normal web page gets the status code as 2xx and can get html contents message then traverse each input field vulnerability type, if that message matches the an error message generated as in table (1) by pattern matching approach that vulnerability found because that the corresponding request has not been sanitized by the application and it is easy to hacker reconstruct the logic of the original query and, therefore, understand how to perform the injection correctly :

Table 1 examples of error message	
1	"supplied argument is not a valid MySQL",//MySQL
2	"mysql_fetch_array\\(\\)",
3	"on MySQL result index",
4	"You have on error in your SQL syntax;",

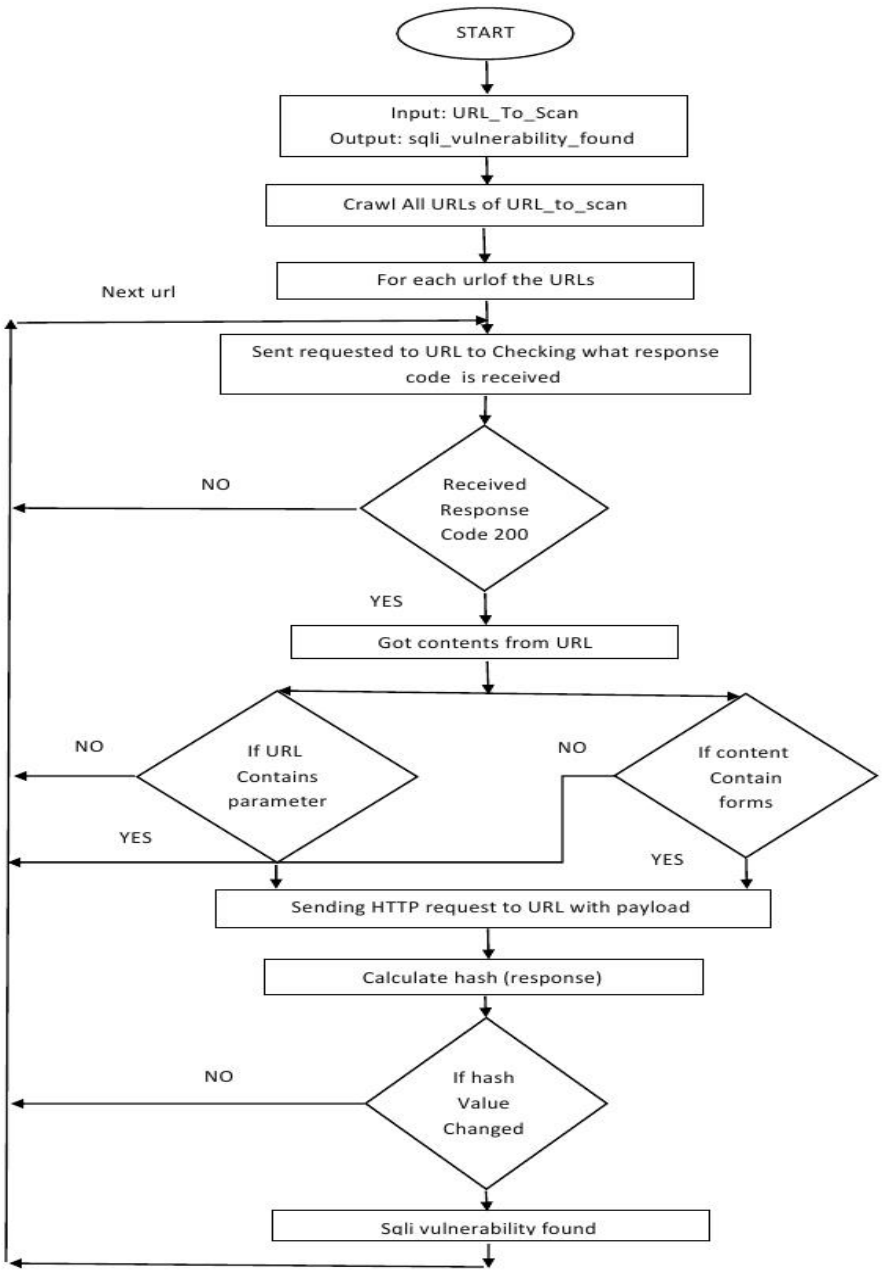
Table 1 examples of error message

Other approach to detect sql independ on sending different malicious requests to the Web application and comparing the similarity of the corresponding responses using a textual distance, in order to identify attacked pages among the response. The problem with current method it cannot detect new error not defined previously also Multilanguage French, Arabic can used to written text of web pages so we need to define all response expected get in many language for different database, for example test broken web authentication by inject the payload (1' or '1'=1……) and check the response if contain (logout, disconnection, that mean is time consume to search all specific data in with the one response, to solve this problem we suggest computing the hash value for the html page that have been returned in order to achieve better detection and less time. The architecture of the proposed system test sql injection (T-SQLI) is showed in figure 1. The system is divided into four main functional models which include web crawler, parser and scanner.

A. The Spider models

This model takes the function of interaction between detection system and web application it first generates web request by the given URL. When the web server receives this request, it will return a response to the client. In this response, we mainly get the following information: status-code, cookies, and message-body and HTML content in the message body then we extract forms sequentially.

Figure 1 proposed system



B. injector Models:

This model send Invalid payload to the affected parameter in the HTTP request, the proposed method it send set of payload attack instead of send single payload attack to reduce the time consume in testing process as in the table, the payload divide into sets and the length of set is determined by max length of input field if there is no limit to input we take the maximum set length can accept by input field, after testing all set of payload and no vulnerability found each set is split equal sets by substring function and retry the testing with new set reach the single character of payload, with this injector method vulnerability detection can be happen by send only one request of set payload rather than may send many request of payload.

Table 2 example of attacking code		
	single	**Set of payload**
1	'OR'1	'OR'1'OR1-- - "OR""="OR 1=1 -- - '='
2	'OR 1 -- -	'OR '1' OR 1 -- - "OR""="
3	"OR""="	"OR 1 = 1 -- - '='
4	"OR 1 = 1 -- -	"OR 1=1 -- - "OR""="

Table 2 example of attacking code

C. Analyzer Model

The proposed method is try to use the hash value to detect if web has vulnerability, the first request has been send is normal request such as http://localhost/index.php?id=1 then compute hash value for response of the web page and consider as signature value. Next all the malicious request will be send one by one and for each response we compute the hash value and compare it with normal signature, any change in the web application by this method any error result by the attacker payload that can affect the response this effect by the value of hash, even the change in the response is two small and error has not known before as shown in algorithm (1).

Algorithm test sql injection
Input: url_to_scan Output: no_vulnerability_found Connecting to database Generating next test ID Begin crawl urls of URL_to_scan For each all urls of urls found testing SQL injection Checking what response code is received fromurl Sending HTTPrequest to url IF received response code is 200 Compute nom_hash value for the current response End if If URL contains forms If from contains input field For each payload Submitting payload to input field Senging HTTP request to URL by post or get method If response is not emptyn Compute mal_hash value for the current response If the mal_hash<>nor_hash or mal_hash<>previous hash Vulnerability found Next payload Next URL Finished SQL injection testing of all URLS Displaying summary reports for test

1.5 Experiment Results

There are several vulnerable target applications that include intentionally Introduced vulnerabilities for ethical testing such DVWA, mutillidae, BWAPP and BRICKS for testing the proposed system we apply many change To web application such inserting new error not known before or print the error Message reporting Arabic or in

different context and this message can be Exploited by an SQL injection. also change the authentication from by display Different message in different language.

Table 3 comparison				
WEB / SCANNER	DVWA	MUTILLIDAS	BWAPP	BRICKS
Acunetix				
Sqlmap				
W3af				
Wapiti				
Skipfish				
T-sqli				

Table 3 comparison

Many automatic web vulnerability scanners that can locate sql injection vulnerabilities, some of which are conveniently included in the backtrack, OWASP Broken Web App, SAMURAI, DOJO software package, so we select four web vulnerability and five web vulnerability scanners to study the effectiveness of proposed mechanisms. Table (3) show the result of experiment where each row in this table have web vulnerability scanners and number of vulnerability found in the vulnerable target overall, the evaluations indications that the proposed system has advantages over the current security assessment in terms of test quality.

Regarding the change in the error message or content, T-SQLI scanner is able to detect it. Moreover, it is the scanner that is able to test the sql injection. T-SQLI is a testing tool written in php version 5.2.1, it compatible with several host operating systems. The experiment is made on a personal computer with a 3GHZ processor. The operating system is windows XP SP2 and web server is Apache (version 2.2.4). the database system is MYSQL sever (version 5.0.27).

1.6 CONCLUSION

There are a lot of techniques and tools to find bugs, errors or vulnerabilities in web application. The target of studying the web vulnerability detection mechanisms is to enhance the ability of web scanner and raise the web page vulnerability detected based on using hash algorithm, in this paper SQL injection vulnerability detection system it is very easy to find SQL injection vulnerability at certain vulnerable parameter or string, and the experiments shows well expected result In the future, research wills includes detect of XSS vulnerability and improving the state of the art on detecting web security vulnerabilities.

REFERENCES

[1] pierre B, paolo F and padhraic S. modeling_the_internet_and_the_web John wiley & Sons Lt,2003
[2] Joe C, sams_teach_yourself_tcpip_in_24_hours_4th_edition, sams, 2012.
[3] David G and Brian T, http_the_definitive_guide ,Oreilly,2002
[4] Josh p, The Basics of Web Hacking, syngrees, 2013.
[5] JOEL S,VINCENT L and CALEB S, Hacking Exposted-Web Applications – Web Application security secrets & solutions ,McGraw-Hill, 2011.
[6] PuleiX, A Model-Driven penetration Test Framework for web Applications, Ph.D. Thesis, Department of computer Science, University of Ottawa,Ottawa,Ontario, Canada, 2012.
[7] Akaren M, TheodoreW, Holly L and Michal c, software security Assurance, IATAC, 2007.
[8] ILENE BURNSTEIN, Practical software testing , Springer,2003.

[9] CHARLES H and GRAND L , Software Security Assurance Framework Mgt Audit , CHL Global Associates and Ounce Labs , Inc,2005.
[10] Ari T , Jared D and Charlie Miller, Artech House Fuzzing for software security testing and_quality_assurance ,ARTECH HOUSE, INC,2008.
[11] GLENFORD J, TOM B and COREY S, THE ART OF SOFTWARE TESTING, john Wiley & Sons, Inc, 2012
[12] Paco H and Ben w, Web security Testing Cookbook Systematic Techniques to Find Problems Fast, Oreilly, 2009.
[13] B. B AGARWAL, S. P. TAYAL and M. GUPTA, An Introduction Software Engineering and Testing, Jones and Bartlett Publishers, 2010.
[14] Gadi E, David M,Noam R, Charlie M,, Robert F, Yoav N and A viram J, Open_Source_fuzzing_Tools, Singers Publishing, Inc, 2000.
[15] William E. Lewis, Software Testing and Continuous Quality Improvement , 3rd +edition , Taylor & Francis Group, LLC, 2009
[16] Manfred R and Clinton D, Software Testing Internationalization, Galileo Press Gmb H, Bonn, 2003
[17] Hadi N and Ronald L. Krutz, Web Commerce Security Design and Development, John Wiley & Sons, Inc, 2011.
[18] Dafydd S and Marcus P , The web Application Hackers Handbook, Wiley Publishing, Inc, 2007.
[19] Sanjay B, Ethical Hacking and Countermeasures- Web Application and Data Servers, EC-Council, 2010.
[20] MarseIN, Hecker Web Exploitation Uncovered, A-LIST Publishing, 2005
[21] Justin C, SQL Injection Attacks and Defense, Elsevier, Inc, 2009.
[22] Christors K, Security Enhanced Application for Information Systems, In Tech, 2012.
[23] Michal Z, The-Tangled-web-a-guide-to-sefffcuring-modern-web-application, No Start Press, Inc, 2012.
[24] Dafydd S and Marcus P, The Web Application Hacker's Handbook-Finding and Exploiting Security Flaws, 2nd John Wiley & Sons, Inc, 2011