ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

Study on Intrusion using Enhanced Sql-Injection and XSS on Web Applications

Neeraj Dagar [1], Pooja Yadav [2]

P.G. Student, Department of Computer Science & Engineering, SRCEM, Palwal, Haryana, India[1]

Assistant Professor, Department of Computer Science & Engineering, SRCEM, Palwal, Haryana, India [2]

ABSTRACT: Dynamic web applications are liable to a wide assortment of security dangers and endeavours. These security dangers exist in a few layers of the web server, including the server programming, the working framework and the web applications that give substance to the customer. Securing dynamic web frameworks is an unending procedure that must include all parts of a framework, including the improvement and execution of web applications, the fixing and refreshing of server programming, the solidifying of the framework arrangement and the checking of vindictive action under this scheme we proposed the new enhanced scheme to form an intrusion on web application using Sql-Injection and Cross Side Scripting

KEYWORDS: Security, Hacking, Sql-Injections, Cross Side Script, Secured Socket Layer.

I. INTRODUCTION

SQL Injection: Sql-Injection commonly means all those attacks to an application, usually Web, in which the program queries an SQL database using variables passed by the user without having previously checked them. As is clear from this first, minimal, explanation, this is not a problem directly attributable to PHP but more commonly applicable to all Web languages, whatever the SQL database used; MySQL is obviously not an exception. While it is extremely simple to avoid these kinds of problems, an incredibly high number of commercial and non-commercial applications are subject to this vulnerability due, as usual, to excessive trust in user input. Trust that, in this case, can truly be fatal and lead to unauthorized access, destruction of corporate databases and so on. One of the most typical examples of Sql-Injection is that of modifying a SELECT query to access data that would normally not be visible to the user or gain access to an authentication system with administrator privileges. Everything is extremely simple. In order to access this application by passing it off for another user, it will be sufficient to enter the login of these and the following password: "123 'OR' '='" (123 is inserted by pure example, there could be any value). In this way the SQL query executed by the program will be: "SELECT * FROM users WHERE username = 'mario' AND password = '123' OR '' = ''". By changing the password in this way, the execution of our script will no longer generate a MySQL error for incorrect syntax, but will authenticate the user correctly. The addition of OR '' = '' (the closing apex is added by the PHP query and, in the intentions, had to close the password field) in fact makes the control of the password superfluous. It is in fact asked to MySQL that the password is 123 or, alternatively, that '' = ''. This second clause is clearly always true and therefore MySQL will find at least one result. We will analyze in the following paragraphs how to avoid the risk of SQL Injection, it is worth stressing now that the two main limits that a hacker will encounter in trying to exploit this type of attack are: the lack of knowledge of the database structure (table names and fields) and, in this specific case, the identification of the administrator's username. In the first case the problem is hardly surmountable since PHP, in case of SQL error, in a standard configuration does not display the query that gave the error, thus not exposing the hacker valuable information on tables and fields. This "protection" is still missing for Open Source applications. In this case, in fact, anyone can download the application and see in a few seconds how the database tables are structured. In the second case it could be quite easy to get around the obstacle by trying, with another SQL Injection, to view the entire user database or alternatively looking for information on the website itself. Precisely for this reason it is highly recommended never give too trivial names to the fields and tables of a database and above try to use a little 'imagination for the username of the administrator trying to avoid terms like admin or root.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

Preventing SQL Injection is a relatively simple task and can be done in various ways depending on your preferences. Let's now analyze, one by one, all the possibilities to protect your web application from SQL Injection. It is advisable to always adopt at least one of the following advice if you do not want to have unpleasant surprises once your product is in the hands of customers. A first solution is provided directly by the php.ini configuration file. By setting true the magic_quotes_gpc entry PHP will take care to add escape characters in front of all the contents in cookies, Post and Get. This solution, however, is not optimal as it may cause incompatibility with many third-party web applications and also do not protect from "malicious" inputs not coming from cookies, Post or Get and in particularly complex applications it is not impossible that, studying the application with particular effort, a hacker can find other ways to put his SQL Injection somewhere. A second solution is to use addslashes () or similar functions like mysql_escape_string () on all the variables inserted in an SQL query. In this way you will always be sure that in all past values the quotes will be converted into \ 'making them so inoffensive. It should however be noted that this precaution, as the previous one is really safe ONLY if, in SQL queries, the apex is used even when numerical values are manipulated. In all queries it is advisable to always use syntax of the type id = '1'. Only in this case will you really be protected from SQL Injection, otherwise a hacker could still add an OR = 1 = after the desired id without our addslashes () can in any way help us. The problem is not limited to the use of the quotes only, but it is presented in the same way as a string delimiter in the query "(quotes) Fortunately, mysql_escape_string () escapes all the characters that can be used as delimiters of string solving many problems. A last alternative, the most complex but also the safest, is to check each variable inserted in a query with regular expression or other specific functions. If for example we know a priori that a variable must contain an integer we could use a solution of this type:

Cross Site Scripting : Common translations include "cross-site scripting attacks" and "cross-site scripting attacks". Generally referred to as XSS attacks, the description of the domestic web security expert Chen Xiaozhong is a kind of "you out of the bag, I'm sorry, he Really cool!" Attack! Different from the general attack pattern directly against web applications, the object of XSS attack is not the website itself, but the computer and user browsing the website, which is somewhat similar to the attack mode of "dashing across the hill"; because many websites do not check the input value well.,allowing attackers to send out malicious code and outputting the problematic HTML data through the website. The final processing and execution of this problematic HTML data is the browser on the user's computer. The mode of fighting cattle across the hill allows users to execute problematic code through a trusted website without knowing it, in order to achieve the purpose of the attack. What about the website itself? Because it is only responsible for outputting the malicious instruction code sent by the attacker into HTML data, the website itself will not generate an error message, and therefore, the manager often cannot timely notice the problem of his own website! Recently, many domestic corporate websites have been revealed to have XSS problems, including even domestic large-scale security companies, and the vulnerabilities may exist for many years, whether it has been used for further attacks, it is difficult to prove! However, according to the records and comments on the website of the Grand Cannon, we can also find that most companies are not precise enough to fix XSS loopholes, and there is no systematic repair mode (Note 4). This also highlights the fact that many domestic business units, even security companies, still lack the correct concepts and understanding of XSS weaknesses.

II. LITERATURE SURVEY

The utilization of PCs, tablets and advanced cells has incredibly expanded in the course of recent years, as substantiated by Stuttard and Pinto (2011) web applications have been produced to perform for all intents and purposes each valuable capacity you could actualize on the web. These incorporate Online Shopping, Social Networking, Gambling and Online club, Banking, Web seek, Auctions, Webmail, and Interactive site pages among others. In a report distributed by Whitehat "86% of all sites tried by Whitehat Sentinel had no less than one genuine defenselessness, and more often than not, significantly more than one – 56% to be exact. (Whitehat, 2015)

As indicated by Shema (2011), numerous associations depend upon altered web applications to actualize business forms. These may incorporate out and out applications, or comprise of modules, for example, on the web, login pages shopping baskets, and different sorts of dynamic substance. A portion of these product applications in your system could be created in-house. What's more, some might be inheritance sites with no assigned proprietorship or support. Physically examining these for provisos and organizing their significance for remediation can be an overwhelming assignment without sorting out endeavors and utilizing robotized devices to enhance precision and productivity.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

Representatives are constantly reacting to demands from both inside and outside the association's corporate system utilizing devices, for example, tablets, cell phones or PCs. While this has huge advantages, the negative downside is the way that programmers may exploit availability to increase unapproved access to essential organization data. Hence, it is basic for any organization to guarantee that they ensure their web applications and lessen the likelihood of a security break to their electronic framework. Testing the shortcoming of web applications with computerized infiltration testing instruments delivers generally brisk outcomes. As of now, there are numerous such devices, both business and open source.

III. PROPOSED SYSTEM

Under the proposed scheme we will form the validation input, encoding input, filtering input and evaluation inputs vide sql-injections and cross side scripts in the form of payloads with command rules which will exhaust the web application vide http or https connectors the below is the proposed workflow :-

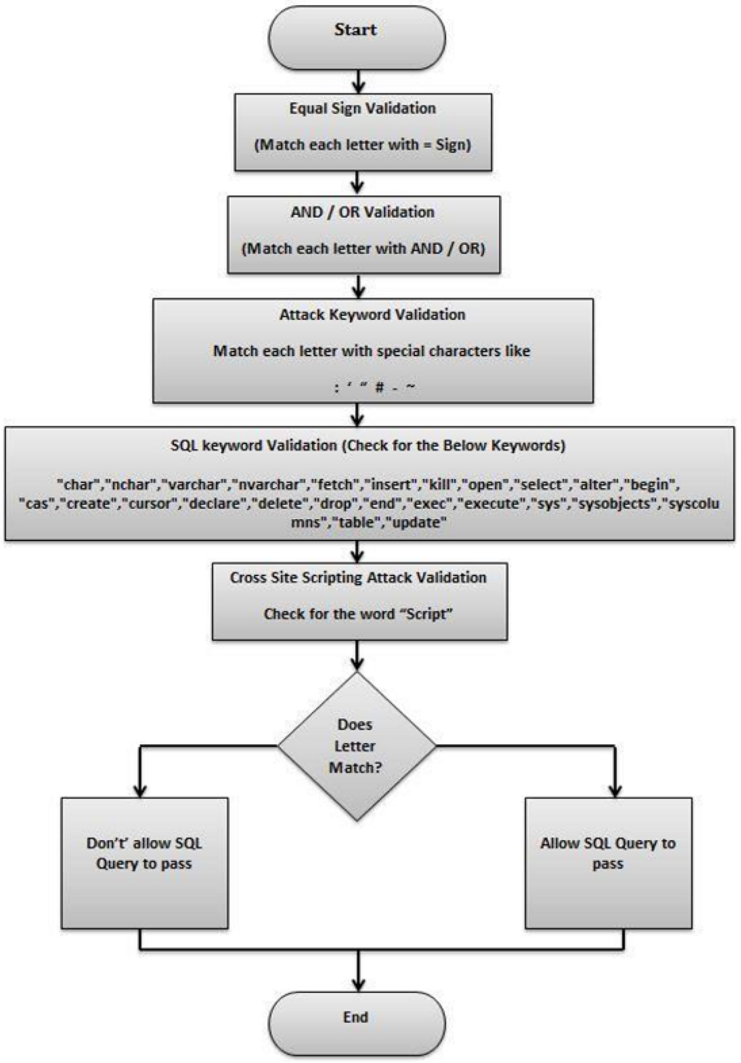
Start
Equal Sign Validation
(Match each letter with = Sign)
AND / OR Validation
(Match each letter with AND / OR)
Attack Keyword Validation
Match each letter with special characters like
: ‘ “ # - ~
SQL keyword Validation (Check for the Below Keywords)
"char","nchar","varchar","nvarchar","fetch","insert","kill","open","select","alter","begin", "cas","create","cursor","declare","delete","drop","end","exec","execute","sys","sysobjects","syscolumns","table","update"
Cross Site Scripting Attack Validation
Check for the word “Script”
Does Letter Match?
Don't' allow SQL Query to pass
Allow SQL Query to pass
End

Figure 1: Architecture and Flow Diagram for Structured Query Language Based Injection and Cross Side Script Based Injection Comprising of Signature Validation using Payloads and Cross Attack Validations with Expressions and Specific Keywords based on Data Control, Data Manipulation and Data Definition Language Models.

IV. CONCLUSION AND FUTURE WORK

The above proposed scheme will form the attack in http or https (secured socket layered based sites) the payload defined under the rules will follow the work flow and in series will raise and intrusion as the accessibility to the web resource is granted using the null or join values the schema information of the databases mounted will be retrieved. Consequently, based on database schema the respective database objects example : tables, users, cursors and more will be accesses and mounted in terminal. Thereafter the data access or intrusion can be form and vital information can be accessed or clinched. For the future scope the architecture can be enhanced not intrude on web sites even to access the FTP and SMTP servers and can be integrated in cloud resources even to perform invasion and intrusion in cloud services and resources.

REFERENCES

1. Alssir, F. T., & Ahmed, M. (2012). Web Security Testing Approaches: Comparison Framework. In Proceedings of the 2011 2nd International Congress on Computer Applications and Computational Science (pp. 163-169). Springer Berlin Heidelberg.
2. Antunes & Vieira (2012). Defending against web application vulnerabilities. Computer, (2), 66-72.
3. Bau, J., Bursztein, E., Gupta, D., & Mitchell, J. (2010). State of the art: Automated black-box web application vulnerability testing. In Security and Privacy (SP), 2010 IEEE Symposium on (pp. 332-345). IEEE.
4. Chen, S. (2014). wavsep. Available: http://sectooladdict.blogspot.com/2014/02/wavsep-web-application-scanner.html. [Accessed 09 July 2015.]
5. Dessiatnikoff, A., Akrout, R., Alata, E., Kaaniche, M., & Nicomette, V. (2011). A clustering approach for web vulnerabilities detection. InDependable Computing (PRDC), 2011 IEEE 17th Pacific Rim International Symposium on (pp. 194-203). IEEE.
6. Dougherty, C. (2012).Practical Identification of SQL Injection Vulnerabilities. 2012. US-CERT-United States Computer Emergency Readiness Team. Citado na, 34. . [Accessed: 08th June 2015]
7. Doupe, A., Cova, M., & Vigna, G. (2010). Why Johnny can't pentest: An analysis of black-box web vulnerability scanners. In Detection of Intrusions and Malware, and Vulnerability Assessment (pp. 111-131). Springer Berlin Heidelberg. [Accessed: 10th June 2015]
8. Fonseca, J., Vieira, M., & Madeira, H. (2014). Evaluation of Web Security Mechanisms using Vulnerability & Attack Injection. Dependable and Secure Computing, IEEE Transactions on, 11(5), 440-453.
9. Granville, K . (2015).Nine Recent Cyber-attacks against Big Businesses. New York Times [online] Available from http://www.nytimes.com/interactive/2015/02/05/technology/recent-cyberattacks.html?_r=1. [Accessed 08 July 2015.]
10. Howard, M., LeBlanc, D., & Viega, J. (2010). 24 deadly sins of software security [electronic book]: Programming flaws and how to fix them. New York: McGraw-Hill.
11. Jovanovic, N., Kruegel, C., & Pixy, E. K. (2010). A Static Analysis Tool for Detecting Web Application Vulnerabilities (Short Paper). In Proceedings of the 2006 IEEE symposium on Security and Privacy, Washington, DC, IEEE Computer Society (pp. 258-263).
12. Kalman., G. (2014). Ten Most Common Web Security Vulnerabilities.[online] Available from: http://www.toptal.com/security/10-most-common-web-security-vulnerabilities [Accessed 08 July 2015.]
13. Kals, S., Kirda, E., Kruegel, C., & Jovanovic, N. (2014). A web vulnerability scanner. In Proceedings of the 15th international conference on World Wide Web (pp. 247-256). ACM.
14. Khoury, N., Zavarsky, P., Lindskog, D., & Ruhl, R. (2011). Testing and assessing web vulnerability scanners for persistent SQL injection attacks. In Proceedings of the First International Workshop on Security and Privacy Preserving in e-Societies (pp. 12-18). ACM.
15. McQuade, K. (2014). Open Source Web Vulnerability Scanners: The Cost Effective Choice?. In Proceedings of the Conference for Information Systems Applied Research ISSN (Vol. 2167, p. 1508). [Accessed: 18th June 2015]