ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 7, Issue 6, June 2020

Analyzing the Strength of Organisation System Using Kali Linux

R.M.Meena Vijayalashmi[1], S.Nigar Fathima[2], J.Hezial[3], Dr.R.Ravi[4].

[1,2,3](Dept. of IT, UG Scholar, Francis Xavier Engineering College, Tirunelveli, India)

[4](Professor, Dept. of IT, Francis Xavier Engineering College, Tirunelveli, India)

Abstract: Web sites are dynamic, static, and most of the time it must be a combination of both. Web sites need security to secure their database. SQL injection attacks are interactive web applications. It provides database services. These applications take inputs from a user and to create an SQL query at run time. In an SQL injection attack, perform an unauthorized database operation by using malicious SQL query as input. An attacker can retrieve or modify confidential and sensitive information from the database using SQL injection attacks. It may jeopardize the confidentiality and security of Web sites which depends on the database. From SQL injection attacks this report presents a "code reengineering" that implicitly protects the applications which are written in PHP. It uses an original approach that merges static as well as dynamic analysis. In this report, we converting plain text inputs that are received from users into prepared statements with the help of technology for moving out SQL injection vulnerabilities from Java code.

Keywords: Dynamic, Static, Security, Database, SQL Injection, Vulnerabilities

.

I. INTRODUCTION

In recent years, widespread adoption of the internet has resulted in rapid advancement in information technologies. The internet is used by the general population for purposes such as financial transactions, educational endeavors, and countless other activities. The use of such as transferring a balance from a bank account always comes with a security risk. Today's web sites strive to keep their users' data confidential and after years of doing secure business online, these companies have become experts in information security. The database systems store non-critical data along with sensitive information behind these secure website. It allows information owners quick access while blocking break-in attempts from unauthorized users. From a database common break-in strategy is to try to access sensitive information by first generating a query that will cause the database parser to malfunction, which is applying this query to the desired database. The method of gaining access to private information is called SQL injection. Since databases are everywhere and are accessible from the internet, dealing with SQL injection has become more important than ever. the current database systems have little vulnerability. Computer Security Institute finds that every year about 50% of databases experience at least one security breach. The loss has been estimated to be over four million dollars.We need to have a good understanding of the kinds of communications that take place during a typical session between a user and a web application to get a better understanding of SQL injection.

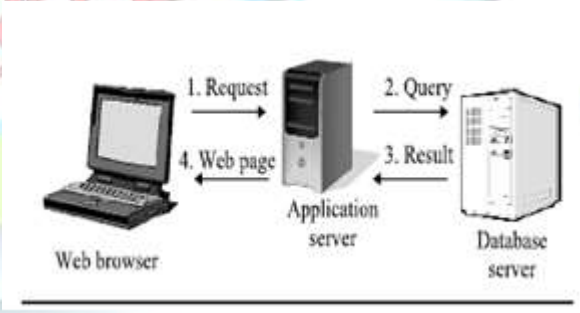

Fig 1: web application architecture

Source: Gary Wassermann Zhendong Su, sound and precise analysis of web applications for injection vulnerabilities, university of California, Davis-2007

A web application, based on the above model, takes text as input from users to give information from a database. Some web applications take that the input is legitimate and use it to build SQL queries to access a database. These web applications validate user queries after submitting them to retrieve data, they become more susceptible to SQL injection

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 7, Issue 6, June 2020

attacks. To produce SQL queries on the web application end attackers, using maliciously crafted input text which containing SQL instructions.

The accepted malicious query may break the security policies of the underlying database architecture with the process of a web application because the result of the query might cause the effect in database parser to malfunction and release sensitive information.

To build an automated fix generation method to prevent SQL injection vulnerability from plain text SQL statements is the goal of this project. A server will gather information about previously known vulnerabilities, SQL statements, generate a patch, and apply a patch by an automated method approach. The process can be completed by someone with no security expertise and secure legacy code, which will allow developers to fix the SQL injection vulnerability.

A. PROPOSED SYSTEM

As future work, we want to evaluate methods using different web-based application script with the public domain to achieve great accuracy in SQL injection prevention approaches. Integrate SQLiX with Nikto HTTP scanner, HTTP scanning proxies, and Metasploit will helps to detect other web vulnerabilities. Also, add a feature to dump the venerable database and database schema.

II. METHODOLOGY

A. SQL INJECTION DISCOVERY TECHNIQUE

An attacker doesn't need to visit the web pages using a browser to find if SQL injection is possible on the site. Generally, attackers build a web crawler to collect all URLs available on each web page of the site. A web crawler is also used to insert illegal characters into the query string of a URL and check for any error result sent by the server. If the server sends an error message, as a result, it is a strong positive indication that the illegal special meta character will pass as a part of the SQL query. The site is open to SQL Injection attack. For example, Microsoft Internet Information Server shows an ODBC error message if any meta character or an escaped single quote is passed to SQL Server. The Web crawler searches the response text for only the ODBC messages.

B. SQL PARSE TREE VALIDATION:

The Parse tree is the data structure built by the developer for the parsed representation of a statement. The grammar of the parse statement's language is required for parsing the statements. In this method, by parsing two statements and comparing their parse trees, we can check if the two queries are equal. When an attacker successfully injects SQL into a database query, the parse tree of the intended SQL query and the resulting SQL query generated after attacker input do not match.

C. *OTHER COMMERCIAL AND OPEN SOURCE SQL INJECTION SCANNERS:*

1) Acunetix web vulnerability scanner:

Acunetix is used to detect various types of web vulnerabilities as below.

1. SQL injection
2. Cross-site scripting
3. CGI scripting
4. Firewalls and SSL
5. URLredirection

2) SQLmap

SQLmap is an SQL injection scanner build in Python. This tool aims to detect SQL injection vulnerabilities and take advantage of these vulnerabilities on a web application. Sqlmap initially detects the loop whole in your site and then use a variety of option to perform extensive back-end database management, enumerate users, dump entire or specific DBMS, retrieve DBMS session user and database, read a specific file on the file system, etc. SQLmap is a bit faster than the acunetix web scanner but still slower than SQLiX, and it also makes very few URL injections into the database as compared to SQLiX. This tool also doesn't have a GUI interface.

D. SQLiX WEB VULNERABILITY TEST SITE

We built a website that is used to test SQLiX. This website provides basic information about SQL Injection attack. We created two partitions on the main web page, one partition provides the component available on-site, and another part is used to show the back end of the site. The main intension behind this structure is that the third user can easily see how the SQLiX tool injecting the different combinations of given URLs and trying to retrieve unauthorized information from the back end. We host this site on http:hostrator.com. To host first we need to register a domain name and upload the all front end file as well as server scripts. We also import the database schemas and data

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 7, Issue 6, June 2020

created on the localhost. Here is the link for the hosted website: http://jagdishhalde.hostrator.com/indexacu.php

III. EXPERIMENTAL RESULT

Fig 2: SQLiX Web vulnerability site with home page SQLiX source site

Fig. 1. SQLiX Web vulnerability site with home page and back end user info table

Fig 4: SQLiX Web vulnerability site showing basic SQL Injection attacks.

IV. CONCLUSION

Most of the web applications use an intermediate layer to accept a request from the user and retrieve sensitive information from the database. Most of the time they use a scripting language to build an intermediate layer. To breach the security of database hacker often uses SQL injection techniques. Generally, the attacker tries to confuse the intermediate layer technology by reshaping the SQL queries. Perhaps, the attacker will change the activities of the programmer for their benefits. Several methods are used to avoid SQL injection attack at the application level, but no feasible solution is available yet. This paper covered the most powerful techniques used for SQL injection prevention. From my research, it concludes that automated technique for preventing, detecting, and logging the SQL injection attack in 'stored procedure' is commonly used and they are concrete methods. A graph control method is also good for small database systems.

REFERENCES

[1]. C. Ebert, A. Dubey. (2019), "Convergence of enterprise IT and embedded systems", *IEEE Softw.*, vol. 36, no. 3, pp. 92-97,

[2]. ABDULLAH, Himli S. (2020). Evaluation of Open Source Web Application Vulnerability Scanners. Academic Journal of Nawroz University, [S.l.], v. 9, n. 1, p. 47-52,

[3]. Dutta N., Jadav N., Dutiya N., Joshi D. (2020) Using Honeypots for ICS Threats Evaluation. In: Pricop E., Fattahi J., Dutta N., Ibrahim M. (eds) Recent Developments on Industrial Control Systems Resilience. Studies in Systems, Decision and Control, vol 255. Springer, Cham.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 7, Issue 6, June 2020

[4]. C. V. Gonzalez and G. Jung, "Database SQL Injection Security Problem Handling with Examples," 2019 International Conference on Computational Science and Computational Intelligence (CSCI), *Las Vegas, NV, USA, 2019, pp. 145-149.*

[5]. Saha S., Das A., Kumar A., Biswas D., Saha S. (2020) Ethical Hacking: Redefining Security in Information System. In: Chakraborty M., Chakrabarti S., Balas V. (eds) Proceedings of International Ethical Hacking Conference 2019. eHaCON 2019. Advances in Intelligent Systems and Computing, vol 1065. Springer, Singapore.

[6]. Kyaw A.K., Tian Z., Cusack B. (2020) Design and Evaluation for Digital Forensic Ready Wireless Medical Systems. In: Garcia N., Pires I., Goleva R. (eds) IoT Technologies for HealthCare. HealthyIoT 2019. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 314. Springer, Cham

[7]. Clark, S.S., Fu, K.: Recent results in computer security for medical devices. In: Nikita, K.S., Lin, J.C., Fotiadis, D.I., Arredondo Waldmeyer, M.-T. (eds.) MobiHealth 2011. LNICST, vol. 83, pp. 111–118. Springer, Heidelberg (2012).