Securing Database using SQL Injection: A Review

Isha Shingari*
Priti Sharma**

Abstract

In one way or other we all are connected with internet. All web applications are dependent on the internet. Now a day's web applications play a vital role in everybody's life. Exponential growth could be observed in many user friendly web applications. Thousands of transactions are done daily through these applications, 80% out of which are vulnerable to malicious attacks according to the survey by the Open Web Application Security Projects (OWSAP) .SQL injections is the highest security threat for web applications .SQL injection is a mechanism for inserting a malicious code in user code. Results in adding or modifying data, leak of confidential information, bypass authentication, performing denial of service, network hacking , table structure, and deleting the database .In this paper we have discussed the various aspects of SQL injection.

Keywords: malicious code , Vulnerabilities ,SQL injection, Web applications, Attack, Database component; formatting; style; styling; insert (key words)

I. Introduction

In last decade the user of the internet has rapidly increased so as the web applications.web applications are the applications that can be accessed over the internet with any browser on any operating system.

To serve numerous users, great volume of data is stored in web applications database [5] all around the world. Security threat in web application is becoming a headache for the developers. Web applications that are not carefully designed(in security terms) are more prone to SQL injection attacks.SQL injection is a kind of code injection in the web application in which attacker add unauthentic code in to the user code to get unlimited and authentic access.

The attackers code is sent in such a way that it is interpreted as a query.SQL injections are very hazardous because it opens the flood gate for various hackers to do what they look for.SQL injections have the capability to affect the SQL query that sends to the backend database [6] such as username, passwords and feedbacks. SQL injection takes advantage of security vulnerabilities in the database. Attackers make use of these flaws to submit malicious code.

According to a survey conducted by OWSAP [11] SQL injections (SQLIA) was measured as top 3rd ranked attack in 2010 but in 2013 it is topped in list of vulnerabilities. The vulnerabilities in website database are outcomes of unsuitable programming duo to which an invader can exploit and access the not to be disclosed information.

Isha Shingari*
Department of Information Technology
Institute of Innovation in Technology & Management, New Delhi, India

Priti Sharma**
Department of Information Technology
Institute of Innovation in Technology & Management, New Delhi, India

II. SQL Query Basics

For the better understanding of SQL injection, Lets discuss some basic about SQL query. SQL is (structured query language) textual language interacted with relational database [4].The query is the collection of statements which a single result set is returned. Database is responsible for storing the information on the servers. The authentic contents of database are marked as rows in the tables [5].

Whenever a query is generated to make a request for a exact set of information from database it initiates the connection with database then performs a query. The query will return only specific row/rows against the request. So the attacker goal is to induce the database query to return the extra rows that were not requested by the user.

So it can be concluded that if an attacker convinces the database or the server to return any row/rows from the users table then they should be able to authenticate illegally.

III. SQL Injection Attacks

Web application consists of many interlinked components and each component plays a significant role in correct working of the web application. Browser sends the request to the web server and returns the desired result. The communication between web server and database is done by means of SQL commands.

With the help of SQL commands attacker can get the access to the user information.SQL injection is an attack on web applications which have vulnerabilities. Actually these vulnerabilities happen due to the weakness in the design of web application due to semantics, logic or syntax. The attackers can put in a crafted query in form of SQL command which is then executed by a web application and bare the back-end database.

The attacks are occurred through user inputs without proper validation. The idea of SQL injection covers following points: SQL injection occurs due to the vulnerabilities in design. The Integrity, Privacy and Security of user are at stack. *SQL manipulation and Code Injection are used for attacks.

* Attacker used special crafted input to attack the system. * SQLIA helps the attacker to get the control over application database.

IV. Basic Principal in SQL Injection SQLIA

It happens where flaws are found in design of the applications. These loopholes can seep out the confidential/sensitive data. For understanding the basic the principal of sql injection, consider the simple example of a login form. The registered user has to submit user name and password.

If an attacker wants to put in malicious code in form of sql injection then the attacker attempts to get the admittance to the database illegally by manipulating the conditions in sql query. This attack generally happens during the processing of the request by providing inputs by the user. For example: Normal

Query: SELECT * from table name WHERE user='' and password= '';

The above SQL statement shows two inputs which user has to provide. Say user name= Ria and password is =steps3 Instead of typing the actual username and password, if the attacker attempts illegally to access the database by adding malicious code (SQLIA). The attacker provides inputs 'any text' OR '1'='1' in user box and '——' in password box.

Query with injection SELECT * from table name WHERE user='abc' OR'1'='1' and password= '——';

Now the attacker has the access to system because condition 1=1 is always true and — indicates the comment statement. This shows the loopholes of the application i.e. the input variables are not properly checked during design stage of the application. Consider another example with a normal statement which will return customer information according to customer names. Where Relation name is Customerinfo, Desired Attribute is Name and condition is, Name="Ria"

Input: SQL> select Name from Customerinfo where Name ="Ria"; Output: Name Ria Output will display only desired result i.e. only one Name But when a few SQL injection statements are executed then the results will be according to the attacker's intention Input: SQL>select name from customerinfo where name ="Ria" or '1'='1': Output: Name Ria Kartik Mihika It will provide the list of customers under the attribute name. SQLIA will provide the unlimited and unauthorized access to the user database. That's why it is considered as a top threat for the application.

SQL code injection attacks can be executed in two ways- code injection and manipulation of SQL statements. Code injection involves the insertion of new SQL statements or database commands into the SQL statement. SQL manipulation modifies the sql statements through set operations.

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

V. Process for SQL Injection

In this process the attacker adds SQL statements through user input fields to get the access to the backend resources. Lack of proper sanitization of inputs becomes the cause attacker to be successful. The process of SQLIA involves three steps [9]

Step1: Attackers send malicious code to web application in the form of request

Step2: Creation of SQL statements The above figure shows the client send the request (HTTP request) to the web server and web server send the request to the database.

The database has relational tables containing data. The data could be extracted from these tables by using SQL commands. As processing of queries and result of these queries will be communicated between web server and the database server through DBMS .So we can observe that SQL injection attacks occurs with some modification in Commands.

VI. Ways of Injecting Code

The most frequent techniques [2] used for injecting Malicious SQL statements in an applications are:- Through user input: In this case, attackers inject SQL Commands by providing rightfully crafted user input. The attacker targets that web application in which user provides information and then request is processed. Like HTTP GET or POST Requests [8] [12] Through cookies: Cookies stores information on client machine generated by web application.

If a Web application uses the cookie's contents to build SQL queries, an attacker can easily attack by putting [12] it in the cookie. Through server variables: Server variables such as HTTP, network headers etc. are used for knowing the logging usage and identifying browser trends. The attacker can assault using forged header [2] in the server variables when these variables are logged to the database.

Through order: The attacker can just enter a malicious string and manipulate code to be executed instantly (i.e. in a direct manner) or by some other activity (i.e. in an indirect manner). The attacker can also modify the implicit functions by making some changes in the values. Through database: The attacker introduces the attacks by SQL manipulation. The attacker modifies the existing SQL statement [9].

This SQL manipulation attack can be through Inference, Basic Union Queries, and Piggy-Backed Queries &Tautology

VII. Injected Code Storage Areas

```
                                                    where
SELECT          Emp_id from
                  Employee
username=' 'or '1'='1' and password=' ';
```

Injected code may be stored using many methods. The method of storage will decide the category of attack. The following classification shows the storage type with related example. *Storage type: Temporary Example: Earlier application search criteria and other cached data.

Storage type: Short-term Storage Example: Information put in daily/weekly logs that are reviewed irregularly or over written/purged frequently Storage Type: Long-term Storage Example: Data permanently stored in backend systems that must be physically removed.

VIII. Attacks Performed by SQL Injection

Bypass authentication: Allows an attacker to log on to an application without supplying a genuine user name.

For Example: SELECT Emp_id from Employee username=' 'or '1'='1' and password=' '; _where _ _ As the condition 1=1 is always true and —— is used for comments, so comments will be ignored and the attacker can enter the system without proper authentication. Information disclosure: Attacker can directly or indirectly get in touch of sensitive information from a database.

In this type, the attacker uses union queries which contain set operators. For example: SELECT Salary_info from Employee where user name='abc ' and password=' ' ; UNION SELECT Salary _info from Employee where Emp_id= '1234' ; The first part of query gives null values but has the right to the information of employee having id 1234 Unauthorized knowledge of database: In this type the attacker injects a query which causes a syntax or logical error in to the database.

On the basis of the resulting error message the attacker pulls out the information regarding the details of database being used. For example: SELECT Emp_id from Employee where user name="xyz" and password=!@#$%^&* The query is incorrect. An error message is displayed due to improper syntax. The attacker uses this message to extract information about the database in being used by user.

Compromised Availability of Data: Attacker deletes or removes information with a harming intention. This can be implemented when an additional query is added with main query. For example: SELECT Salary_info from Employee where user name='abc 'and password=' ' ; DROP table user: First query is null so the system executes the second query and will delete the table from the database. Remote command executions: It allows an attacker to host operating system accessible by performing a command execution.

In this the attacker performs remote execution of procedure by injecting queries.[7] For example: SELECT Emp_ Salary from Employee where username=" ; SHUTDOWN; and password="; In above query only SHUTDOWN operation will be performed which shutdown the database.

IX. Why SQL Injection Still Works

After all of these years, SQL injection vulnerabilities still stand as an old reliable for attackers looking for to break into corporate databases. The following are the main reasons [10] Reluctant Behavior: Even after knowing that that SQLIA is a threat, the designers are still reluctant in reducing the attractiveness of database. The data in the SQL database must be encrypted and the encryption key should be somewhere else.

Needs fewer Resources: The resources are required for SQL injection attacks. They are just a computer and rest depends on the capabilities of the attacker. Rest is up to the attacker to which extent the attacker could peep in to our database. Insecure Development Architecture: The biggest reason behind SQL injection is improper development planning and the usage of insecure development architecture.

Protecting yourself from such attacks lies in design and architecture in which a portal is developed, and there are many techniques which can improve the security of a portal against SQL injections. Trusting Input: The shortage of many of techniques comes down to developers and their organizations putting too much trust into user input.

Trust without the verification is one of the key reasons why SQL injection is still so prevalent. Some application developers just don't know any better; they inadvertently write applications that blindly accept any input without validation. If organizations want to reduce the risk of SQLIA then they are required to sanitize input.

Non-belief at All Costs: The Standard Query Language acts as a widespread language that works across database platforms. But that quest to keep application code and application data on a single database server is a double-edged sword because stored procedures or prepared statements are often specific to a database platform.

f) Code Samples Outdated: Most code samples from which programmers take their first SQL programs are susceptible to SQL injection. If organizations use legacy technologies or components that promote construction of ad-hoc queries, then they're likely to boost their risk of SQL injection attacks. g) Budget shortfalls: The monetary constraints are keeping the vulnerabilities active. The cost of writing code and doing new code and ensuring the code is not exploitable.

The cost of running the code repeatedly adds to cost in very competitive market. X CONSEQUENCES OF SQLIA Injection attacks can have sewer effects on database not only they can modify the data but can even hack one's network. Following are some consequences of SQLIA [9] [10] determining database schema: The attacker can have complete access of the database by knowing database schema.

The attack can be very specific by knowing as table names, column names, and column data types. Extracting data: when the attacker has full access on database the sensitive information can be access which is highly desirable by the attacker. Most of the SQLIA attacks are done for this reason.

Adding or modifying data: Sometimes the objective of attacker is to add or Change information in a

database to mark the presence or to alert the user's data. Executing the denial of service. When attacker is involved in locking or dropping database tables in the database of web application results in denying service to other user. The attacker can terminate the entire database also.

Bypassing authentications: In this category the objective of the attack is to bypass database and application authentication mechanisms. Bypassing such methods can allow the attacker to presume the privileges and rights associated with another application user.

X. Conclusion

Information is most important asset and achieving the security of the data stored on web is the top priority in this competitive world.

The attacks exploits security vulnerability occurs in DB of an application by injecting some code. Vulnerabilities are becoming the opportunities for the attackers. The absence of good mechanism for accessing the application at design level is exposed. We need a complete methodology for evaluating the performance of the source code in the existing system.

References

1. M. Dornseif - Common Failures in Internet Applications, May 2005
2. William G.J. Hal fond, Jeremy Vie gas, and Alessandro Orso: A Classification of SQL Injection Attacks and Countermeasures 2006 IEEE.
3. D. A. Kindy and A. K. Pathan: A Survey on SQL Injection: Vulnerabilities, Attacks, and Prevention Techniques 2011 IEEE
4. Atefeh Tajpour, Suhaimi Ibrahim, Maslin Masrom: SQL Injection Detection and Prevention Techniques, International Journal of Advancements in Computing August 2011.
5. Geoffrey Vaughan- Understanding SQL injection attacks inside and out- University of Ontario Institute of Technology, Canada- 2012.
6. Pushkar Y.Jane, M.S.Chaudhari- SQLIA: Detection And Prevention Techniques: A Survey IOSR Journal of Computer Engineering September 2012
7. Asha N, M.Varun Kumar, Vaidhyanathan.G "Preventing SQL Injection Attacks". The Third International Journal of Computer Applications volume52-no-13, 2012
8. Chad Dougherty- Practical Identification of SQL Injection Vulnerabilities, Carnegie Mellon University. Produced for US-CERT© 2012
9. V. Nithya, R.Regan, J.vijayaraghavan-A Survey on SQL Injection attacks, their Detection and Prevention Techniques- International Journal of Engineering and Computer Science April, 2013
10. Ericka Chickowski, Contributing Writer Dark Reading May, 2013 [11] OWSAP –The open web application security project (OWASP) available at www.owasp.org/index.php/mainpage last access Jan 2014
11. www.w3.org/protocols- last access January 2014.
12. Amit Banchhor, Tushar Vaidya" Sql injection: A Survey paper" International Journal of Advanced Technology in Engineering and Science Volume No 03, Special Issue No. 01, May 2015
13. Yash Tiwari, Mallika Tiwari" A Study of SQL of Injections Techniques and their Prevention Methods International Journal of Computer Applications (0975–8887)Volume 114–No. 17, March 2015