Volume 4, Issue 3, March 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

SQL Injection Attacks on Web Applications

Chandershekhar Sharma*
Research scholar
Rajasthan Technical University
Kota, India

Dr. S.C. Jain
Professor in Computer Engineering Department
Rajasthan Technical University
Kota, India

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. In last 10 years we have observed the exponential growth in 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 technique for inserting a malicious code in user code. Results in leak of confidential information, adding or modifying data, performing denial of service, bypass authentication, table structure, network hacking and even corrupting or deleting the database .In this paper we will thoroughly discuss the various aspects of SQL injection.

Keywords: - SQL injection, Vulnerabilities, malicious code, Attack, Web applications, Database

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 the help of any browser on any operating system. To serve large number of user great volume of data is stored in web applications database [5] all around the world. Security threat in web application is becoming headache for the developers. Web applications which are not carefully designed(in terms of security) are prone to SQL injection attacks.SQL injection is a type of code injection in the web application in which attacker add malicious code in to the user code to get unauthorized and unlimited access. The attackers code in transmitted in such a way that it becomes a query.SQL injections are dangerous because it open the flood gate for hackers to do what they desires.SQL injections has the capability to influence the SQL query that passes to the back-end database [6] such as feedbacks, username and passwords. SQL injection exploits security vulnerabilities in the database. Attackers utilize these loopholes to submit malicious code. According to a survey by OWSAP [11] SQL injections (SQLIA) was considered as top 3rd attack in 2010 but in 2013 it is ranked at top in the list of vulnerabilities. The vulnerabilities in website database are results of inappropriate programming duo to which an attacker can exploit and access the confidential information.

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 used to interact with relational database [4].The query is the collection of statements which returns a single result set. Database is responsible for storing the information on the servers. The actual contents of database are stored as rows in the tables [5]. Whenever query makes a request for a particular set of information from database it first establishes 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 convince the database query to return additional rows that is not requested by the user. So we can conclude 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 has many interlinked components and each component plays a important role in proper 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 with the help of SQL commands. With the help of special crafted SQL commands attacker can access the user information.SQL injection is an attack on web-applications which have vulnerabilities. Actually these vulnerabilities are the weakness in the design of web application due to logic, syntax or semantics. The attackers can add a crafted query in form of SQL command which is executed by web application and exposed the back-end database. The attacks occur through user inputs without proper validation. The concept of SQL injection consists of 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.
* SQLIA helps the attacker to gain control over application database

IV. BASIC PRINCIPAL IN SQL INJECTION

SQLIA happens where loopholes are found in design of the applications. These loopholes can leak 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 add malicious code in form of sql injection then the attacker attempts to access the database unauthorized by manipulating conditions in sql query. This attack usually 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= Myra and password is =steps3

Instead of typing the actual username and password, if an 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 can access to the system because condition 1=1 is always true and --- indicates the comment statement. This show the weaknesses of the application i.e. the input variables are not properly checked during design phase 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=”Myra”

Input:

SQL> select Name from Customerinfo where Name =”Myra”;

Output:

Name

Myra

Output will display only desired result i.e. only one Name

But when some SQL injection statement is executed then the results will be according to the attacker’s intention

Input:

SQL>select name from customerinfo where name =”Myra” or ‘1’=’1’:

Output:

Name

Myra

Kartik

Mihika

It will provide the list of customers under the attribute name.

SQLIA will provide the unauthorized and unlimited access to the user database. That’s why it is considered as a top threat for the application. SQL injection attacks can be executed in two ways- manipulation of SQL statements and code injection.SQL manipulation involves modifying the sql statements through set operations. Code injection is when an attacker inserts new SQL statements or database commands into the SQL statement.

V. PROCESS FOR SQL INJECTION

In this process the attacker adds SQL statements through user input fields to access the backend resources. Lack of 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 via request

Step2: Create SQL statements

Step3: Submits this query to backend database.

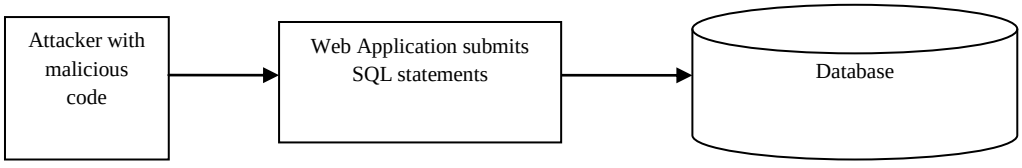

Fig. 1: Steps in Processing SQL Injection

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 consists of relational tables containing data. The data can be extracted from these tables by using some syntax i.e. by 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 can occurs with some modification in Commands.

VI. WAYS OF INJECTING CODE

The most common techniques [2] used for injecting Malicious SQL statements in an applications are:-

a) Through user input: In this case, attackers inject SQL Commands by providing suitably 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]
b) 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 embedding [12] it in the cookie.
c) 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 attack using forged header [2] in the server variables when these variables logged to the database.
d) Through order: The attacker can simply enter a malicious string and cause the modified code to be executed immediately (i.e. in a direct manner) or by some other activity (i.e. in an indirect manner). The attacker can also manipulate the implicit functions by changing the values.
e) Through database: The attacker injects 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

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: Previous application search criteria and other cached data.
* Storage type: Short-term Storage
Example: Information stored 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 manually removed.

VIII. ATTACKS PERFORMED BY SQL INJECTION

a) Bypass authentication: Allows an attacker to log on to an application without supplying a valid user name. For Example:

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

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.
b) Information disclosure: Attacker can directly or indirectly access 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 can access the information of employee having id 1234
c) 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 extracts 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 the user.
d) 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.

e) Remote command executions: Allows an attacker to access the host operating system by performing a command execution. In this the attacker can perform 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 seeking to break into corporate databases. The following are the main reasons [10]

a) 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**.**

b) Needs fewer Resources: The resources required for SQL injection attacks are just a computer and rest depends on the capabilities of the attacker. Rest is up to the attacker to which extent the attacker can peep in to our database.

C) Insecure Development Architecture:

The biggest reason for SQL injection is improper development planning and the use of insecure development architecture. Protecting yourself from these attacks lies in the architecture and design in which a portal is developed, and there are many techniques which can improve the security of a portal against SQL injections.

d) Trusting Input:

The lack of many of techniques comes down to developers and their organizations putting too much trust into user input. Trust without verification is one key reason why SQL injection is still so prevalent. Some application developers simply don't know any better; they inadvertently write applications that blindly accept any input without validation. If organizations want to decrease the risk of SQLIA then they need to sanitize input.

e) Non-belief at All Costs:

The Standard Query Language acts as a common 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 vulnerable to SQL injection. If organizations use legacy technologies or components that promote construction of ad-hoc queries, then they're likely to increase their risk of SQL injection attacks.

*g) Budget shortfalls***:**

The budgetary constraints are keeping the vulnerabilities alive. The cost of writing code and doing new code and ensuring the code is not exploitable. The cost of running the code again and again adds to cost in very competitive market**.**

X. CONSEQUENCES OF SQLIA

Injection attacks can have sewer effects on the database not only they can modify the data but can even hack one's network. Following are some consequences of SQLIA [9] [10]

a) Determining database schema: The attacker can have full 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.

b) 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.

c) Adding or modifying data: Sometimes the goal of attacker is to add or Change information in a database to mark the presence or to alert the user's data.

d) Performing 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 shut down the entire database also.

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

XI. 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 lack 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, "Anomaly based character Distribution Models in the 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
[12] www.w3.org/protocols-

