International Journal of Engineering Inventions
ISSN: 2278-7461, www.ijeijournal.com
Volume 1, Issue 8 (October2012) PP: 70-75

A STUDY OF SQL INJECTION IN BANKING TRANSACTION

Varun Tiwari[1], Kimmi Makhija[2], Jyoti Ratra[3]
[1]*Research Scholar NWMDI University South Africa.*
[2,3]*Asst. Prof. KIRAS, (Affiliated GGSIP University)*

Abstract:—In these days the use of the World Wide Web (WWW) is increasingly rapidly, so there are lot of problems comes from hacking. One of the most important fields of Hacking is Banking Sector. In case of Banking System there are lots of transactions daily, soproviding the security from the attackers is necessary**.** Now these days the major issues of the security in the Banking Transaction are SQL Injections, which are creating a serious issues regarding the attacks of Banking applications and acquiring the secret information's such as id and Password and accessing the bank databases through the SQL injections**.** My main motive to presenting this paper is describing the sql injection method, techniques and how to prevent this situation? These methods we can protect transaction from the attack by using SQL injections.These methods are used to denote the parameters that are used to attack by the SQL injections and analyzed by the transactions which cause illegal access. By these methods we can totally protect the applications without any hacking of the database and completely condemned the attacks and it will not generate any wrong transactions as a correct one. In an SQL injection attack, an attacker might insert a malicious SQL query as input to perform an unauthorized database operation. Using SQL injection attacks, an attacker can retrieve or modify confidential and sensitive information from the database.

Keywords:—SQLInjection, Security, Detection and Prevention.

I. INTRODUCTION

"An attack technique used to exploit web sites by altering backend SQL statements through manipulating application input." SQL injection is an attack in which malicious code is inserted In to strings that are later passed to an instance of SQL Server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives. Even parameterized data can be manipulated by a skilled and determined attacker. List of some SQL Injection attacker bank are

1. Forex Swiss Bank Vulnerable to SQL Injection [1]
2. HDFC Bank Database Hacked by zSecure team using SQL injection vulnerability [2]

[1] SQL Injection Vulnerability found in Dukascopy by zSecure Team. Dukascopy offers direct access to the Swiss Foreign Exchange Marketplace. This market provides the largest pool of ECN spot forex liquidity available for banks, hedge funds, other institutions and professional traders. To accommodate the existing banking relationships of its clients, Dukascopy offers full Prime Broker capability with give up facility, by utilizing an extensive network of banking partners. [5]

[2] ZSecure team is back in news again, this time they have discovered a critical SQL injection vulnerability in HDFC Bank's Web Portal. Using this critical flaw HDFC Bank's various databases can be accessed and dumped as well. This critical flaw really affects the customer relations of HDFC Bank's and this really questions the existing security in place within bank.HDFC Bank is the leading bank in India but they lack behind the basic security that needs to be implemented. ZSecure team claimed in their blog post that even after sending they complete details about the vulnerability and even after conducting the vulnerability assessment from the third party service provider they were not able to discover this critical flaw which existed in their web portal. This really raises a big question on their existing security policy. [6]

SQL Injection is a type of security exploit in which the attacker adds Structured Query Language (SQL) code to a Web form input box to gain access to an organization's resources or to make changes to data. Using this technique, hackers can determine the structure and location of key databases and can download the database or compromise the database server. "What makes this vulnerability so pervasive is that SQL Injection attacks can prey on all types of Web applications - even those as simple as a monthly loan payment calculator or a 'signup for our customer newsletter' form," .

"SQL Injection is successful only when the web application is not sufficiently secured,". "Unfortunately, the majority of websites and web applications are not secure. Thus, we are advising all organizations to use 'input validation' for any form to ensure that only the type of input that is expected is accepted." [1]

1.1 How Hacker Done

The hacker goes to coffee shop and connects to the same Wi-Fi network you are connected to. He runs a series of utilities to redirect other user's data through his machine. He runs a number of other utilities to sniff the data, act as an SSL Certificate Server and to be the Man-the-Middle. The following diagram shows a very simplified graphic of how your SSL Banking session should work under normal conditions, then how it would work during an attack:

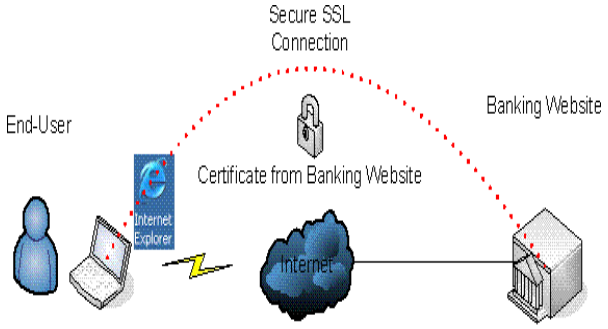


Fig 1. Secure SSL Connection

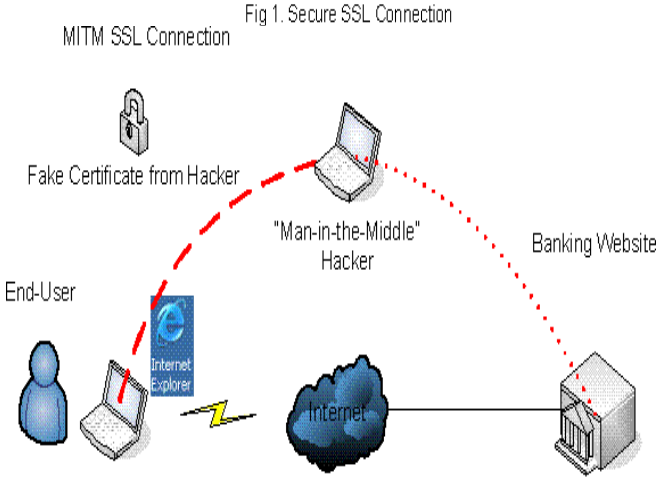


Fig 2. MITM InSecure SSL Connection

An important concept to grasp here is that a certificate is used to establish the secure SSL connection. This is a good thing, if you have a good certificate and are connecting directly to the website to which you intended to use. Then all your data is encrypted from your browser to the SSL website where the bank's website will use the information from the certificate it gave you to decrypt your data/credentials. If that is *truly* the case, then it is pretty darn hard for a hacker to decrypt the data/credentials being transmitted, even if he is able to sniff your data.

This is a bad thing if you have a "Fake" certificate being sent from the hacker, and you are actually connecting to his machine, not directly to the bank's website. In this case, your credentials are being transmitted between your browser and the hacker's machine. The hacker is able to grab that traffic, and, because he gave you the certificate to encrypt the data/credentials, he can use that same certificate to decrypt your data/credentials.

1.2 Approaching Methods

1. Input Validation
2. Static Query Statement
3. Least Privilege
4. Code Verification
5. Web Application Gateway
6. SQL Driver Proxy
7. MISC Method

Some other method to prevent attack of SQL Injection:

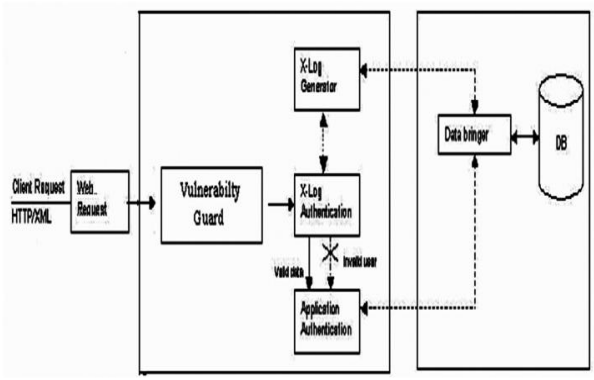
1. Trust no-one
2. Don't use dynamic SQL when it can be avoided
3. Update and patch
4. Firewall
5. Reduce your attack surface
6. Use appropriate privileges
7. Keep your secrets secret
8. Don't divulge more information than you need to

9. Don't forget the basics
10. Buy better software

1.3 Preventing Techniques

1.3.1 X – Log Authentication Technique

This approach addresses SQLIA's (Sql Injection Attack) with runtime monitoring. The key insights behind the approach are that (1) the sourcecode contains enough information to infer models of theexpected, legitimate SQL queries generated by theapplication, and (2) an SQLIA, by injecting additional SQLstatements into a query, would violate such a model.Proposed technique monitors the dynamically generatedqueries with the Data model which is generated by X-LogGenerator at runtime and checks them for compliance. Ifthe Data Comparison violates the model then it representpotential SQLIA' s and prevented from executing on thedatabase and reported. For each application, when the loginpage is redirected to our checking page, it was to detect andprevent attacks without stopping legitimate accesses.Moreover, our technique proved to be efficient, imposingonly a low overhead on the Web applications. This techniqueconsists of three filtration techniques to prevent SQLI'S.We summarize the steps and then describe them in moreConclusions. Detail in subsequent sections.



Vulnerability Guard: Vulnerability Guard detects theWildcard characters or Meta characters and prevents themalicious attacks.

X – Log Authentication: X-Log valuator validate from-Log Generator where the Sensitive data's are Stored fromthe Database, The user input fields compare with the dataexisted in X-Log generator if it is identical then the queryis allowed to proceed.

Stored Procedure: Testing the size and data type ofinput and enforce appropriate limit. Stored Procedures isused to validate user input and to perform server sidevalidation. The safety of stored procedures depends on theway in which they are coded and on the use of adequatedefensive coding practices. These Three input filtrations areused to improve the scalability, performance andoptimization.

1.3.2 AMNESIA

AMNESIA (Analysis and Monitoring for Neutralizing SQL Injection Attacks) is the prototype tool that implements our techniqueto counter SQLIAs for Java-based web applications. AMNESIA is developed in Java and its implementation consists of three modules that leverage various existing technologies and libraries:

Analysis module: This module implements Steps 1 and 2 of our technique. Its input is a Java web application and it outputs list of hotspots and a SQL-query models for each hotspot. For the implementation of this module, we leveraged JSA [5]. The analysis module is able to analyze Java Servletsas well as JSP pages.

Instrumentation module: This module implements Step 3 of our technique. It inputs a Java web application and a list ofhotspots and instruments each hotspot with a call to the runtime monitor. We implemented this module using INSERT,a generic instrumentation and monitoring framework for Java developed at Georgia Tech [4].

Runtime-monitoring module: This module implements Step 4 ofour technique. It takes as input a query string and the IDof the hotspot that generated the query, retrieves the SQLquerymodel for that hotspot, and checks the query againstthe model. For this module, we also leveraged INSERT.

Figure 6 shows a high-level overview of AMNESIA. In thestatic phase, the Instrumentation Module and the Analysis Moduletake as input a web application and produce (1) an instrumentedversion

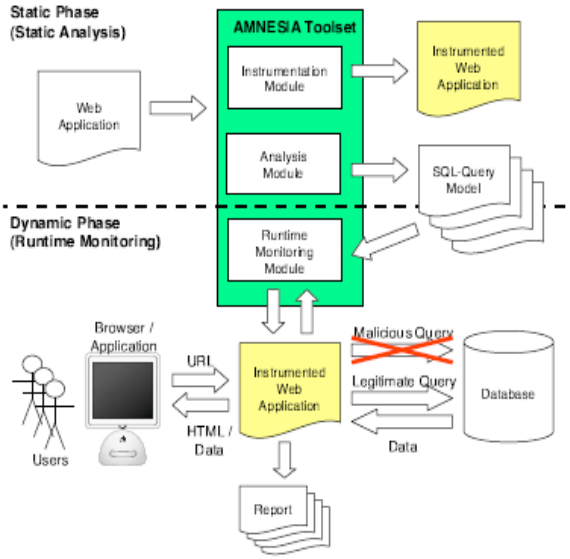


Figure 6: High-level overview of AMNESIA.

of the application, and (2) an SQL-query model for eachhotspot in the application. In the dynamic phase, the Runtime-Monitoring Module checks the dynamic queries while users interactwith the web application. If a query is identified as an attack, it is blocked and reported. [2]

Another Step of Amnesia Tool

Identify hotspots: Scan the application code to identifyhotspots—points in the application code that issue SQLqueries to the underlying database.

Build SQL-query models: For each hotspot, build a modelthat represents all of the possible SQL queries thatmay be generated at that hotspot. An SQL-querymodel is a non-deterministic finite-state automaton inwhich the transition labels consist of SQL tokens, de-limiters, and placeholders for string values.

Instrument Application: At each hotspot in the application, add calls to the runtime monitor.

Runtime monitoring: At runtime, check the dynamically-generated queries against the SQL-query model andreject and report queries that violate the model.

1.4 SECURITY EXAMPLE OF SQL INJECTION

Websites of many banks, credit unions, smaller online retailers, and many government agencies remain highly vulnerable to SQL injection attacks.

8th November 2010: (BBC News)

The Royal Navy's website has been hacked by a suspected Romanian hacker known as TinKode.

December 2009: (The New York Times)

An attacker breached a Rock You plaintext database containing the unencrypted usernames and passwords of about 32 million users using an SQL injection attackWebsites of many banks, credit unions, smaller online retailers, and many government agencies remain highly vulnerable to SQL injection attacks.

Some of the real world examples:

Most of the application will have a login screen and we construct dynamic SQL statement with the screen input as follows:

SELECT *FROM Users WHERE login = 'ravi' and password = 'dBa#1';

If the attacker modifies the user name supplied as ‘ravi' or 1=1; -- then the code will be as follows:

SELECT *FROM UsersWHERE login = 'ravi' or 1=1;--and password = 'dBa#1';

If the above statement executes, the attacker can gain access to the database as 1=1 is always true.

The attacker could log on as any user, given that they know the users name, using the following input:

Username: admin'--

If the user specifies the following:

Username: ‘drop table users--

Password:

The 'users' table will be deleted, denying access to the application for all users.

To illustrate how an SQLIA occurs, we introduce a simple example that we will use throughout the paper. The example is based on a Servlets, show.jsp, for which a possible implementation is shown in here: [4]

1.4.1 Example of Secure Login by Java Language

```
public class Show extends HttpServlet
{
publicResultSetgetUserInfo(String login, String password)
{
```

```
Connection conn = DriverManager.getConnection("MyDB");
Statement stmt = conn.createStatement();
String queryString = "";
queryString = "SELECT info FROM userTable WHERE ";
if ((! login.equals("")) && (! password.equals("")))
 {
queryString += "login='" + login +  AND pass='" + password + "'";
}
 Else
 {
queryString+="login='guest'";
}
ResultSettempSet = stmt.execute(queryString);
returntempSet;
}.
}
```

1.4.2 Example of Secure Login by Php Language

To prevent such an attack, a secure login technique is required. Here, in this article, we discuss the coding of a secure login script using PHP and Mysql.

Step I: Create a database and a table 'members' in it:

```
CREATE TABLE `members` (`username` varchar (20), `password` varchar (128))
```

Step II: Create a Login Form:

```
<formaction="process_login.php" method="post">
Username: <input type="text" name="username" />
 Password: <input type="password" name="password" />
<input type="submit" value="Login" /></form>
```

Connect to Mysql Server:

```
$host = 'local host'; // Host name Normally 'LocalHost'
$user = 'root'; // MySQL login username
$pass = ''; // MySQL login password
$database = 'test'; // Database name
$table = 'members'; // Members name
mysql_connect ($host, $user, $pass);
mysql_select_db($database);
```

Step III: Now, you need to provide mechanism to avoid SQL Injection. For this, escape special characters like ", ', \
We can escape special characters (prepend backslash) using mysql_real_escape_string or add slashes functions. In most cases PHP will this do automatically for you? But PHP will do so only if the magic_quotes_gpc setting is set to on in the php.ini file. If the setting is off, we use mysql_real_escape_string function to escape special characters. If you are using PHP version less that 4.3.0, you can use the add slashes function instead.

```
name = mysql_real_escape_string ($_POST['username']);
$password = md5 ($_POST ['password']);
$result = mysql_query ("SELECT * FROM $table WHERE username = '$username' AND password = '$password'");
```

Here, we use the MD5 (Message Digest 5) Algorithm, that generates the message digest for the password. So, while writing the script for registration page, care must be taken that the md5 of the password entered by the user must be stored in the database, instead of the actual text password.

Validating the login:

```
if(mysql_num_rows($result))
{
session_start();
  $_SESSION['username'] = htmlspecialchars($username);
}
else
{
  // Invalid username/password
echo '<p><strong>Error:</strong> Invalid username or password.</p>';
}

// Redirect
header('Location: http://www.example.com/loggedin.php');
exit;
```

You are done!! This code will help prevent the SQL injection problem. However, it must be noted that no script is 100% secure. So, it is advisable to provide multilevel security process, which makes the login more secure.

II. CONCLUSION

In this paper we mention there are lot of mechanism and technique used to prevent the SQL Injection in Banking Transaction. We have presented a survey reports on various types of SQL Injection attacks in Bank, Methods, and Prevention Techniques. At Last Conclusion the best and suitable techniques are Amnesia and X-Log Authentication. In this technique we use different type's language such as Java and Php.

REFERENCES

1. Research Book "SQLInjectionWhitePaper".
2. *AMNESIA: Analysis and Monitoring for Neutralizing SQLInjection Attacks* William G.J. Halfond and Alessandro Orso, College of Computing, Georgia Institute of Technology, *ASE'05,* November 7–11, 2005.
3. "Securing Web Application Code by Static Analysis and Runtime Protection", *In Proceedings of the 12th International World Wide Web Conference (WWW 04),* May 2004.
4. Steve Fried's Unixwiz.net Tech Tips "SQL Injection Attacks by Example".
5. Report on "Dukascopy: Forex Swiss Bank Vulnerable to SQL Injection".
6. Report on "HDFC Bank Database Hacked by zSecure team using SQL injection vulnerability".
7. Research Book "SQL Injection Attacks and Defense" author Justin Clarke.
8. "Preventing SQL Injection Attacks Using AMNESIA" William G.J. Halfond and Alessandro OrsoCollege of Computing Georgia Institute of Technology , *ICSE'06,* May 20–28, 2006, Shanghai, China.
9. "A Survey on SQL Injection: Vulnerabilities, Attacksand Prevent ion Techniques" *Diallo Abdoulaye Kindy and Al-Sakib Khan Pathan*Department of Computer Science, International Islamic University Malaysia, Malaysia.
10. M. Ruse, T. Sarkar and S. Basu. Analysis & Detection of SQLInjection Vulnerabilities via Automatic Test Case Generation ofPrograms. 10th Annual International Symposium on Applicationsand the Internet pp. 31 – 37 (2010)
11. Preventing SQL Injection Attacks in Stored Procedures Wei, M. Muthuprasanna, SurajKothari Dept. of Electrical and Computer Engineering Iowa State University Ames, IA – 50011.