IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. VII (Mar-Apr. 2014), PP 01-04
www.iosrjournals.org

Multi-Tier Web Security on Web Applications from Sql Attacks

S.Suganya, D.Rajthilak, G.Gomathi
Dept. of Computer Science & Engg Annai Teresa College Of Engineering, Thirunavalur-607204, Ulundurpet TK,Villupuram DT.

Abstract: *This paper deals with the techniques that are used to prevent the web applications from web security vulnerabilities caused by hackers which leads to the misusage of data. Web services and applications have increased in both popularity and complexity over the past few years. Daily tasks, such as purchasing, banking, travel, and social networking, are all done via the web. Web services have always been the target of attacks. These attacks have recently become more diverse, as attention has shifted from attacking the front end to exploiting vulnerabilities of the web applications in order to corrupt the back-end database system. We present Double Guard, a system used to detect attacks in multitier web services. Our approach can create normality models of isolated user sessions that include both the web front-end and back-end network transactions. We analyze principles of SQL attacks, study a database protection system which is used between the Web application and the database. The role of a Web application and database in the database between the protection system for ordinary users and administrators were made by different users of protective measures to effectively guarantee the security of the database.*
Index Terms: *SQL injection, cross side scripting, security vulnerabilities, web applications.*

I. Introduction

Web is used in various daily need tasks such as banking, travel, and social networking. These services typically employ web server in the front end that consist of user interface logic and back end server that employ database server. Due to their regular use for personal, corporate data web dependent services have always been target for attacks.[1] SQL Injection is commonly used by hackers to steal and modify data from organizations. By taking advantage of improper parsing of inputs of the web forms, hackers could do the following steps: pass SQL commands through a web application to be executed by the backend database, query database directly, and gain access of the data held within the database. Dynamic script languages, such as ASP, ASP.NET, PHP, JSP and CGI, are vulnerable to this Attack[3]. XSS allows attack or to embed malicious content, such as JavaScript, VBScript, ActiveX, HTML or Flash, into a vulnerable dynamic page to fool the user and gather victim's data. XSS may occur anywhere a web application uses input from a user in the output it generates without validating it. The consequences of such attack may be as trivial as image or layout change, or as severe as disclosure of user's session cookies To detect attack using normal traffic in multi tiered web architecture Double Guard system can be used. Double Guard system can employ normal models for user sessions that use both web front end server (HTTP) and database back end server (SQL). In this system each user‟s web session can be assigned to dedicated container using light weight virtualization technique[1]. the backend queries can vary depend on value of parameters passed in the HTTP request and application logic. In some cases same applications primitive functionality can be triggered by many different web applications. So in these cases, the resulting casual mapping between web server and database request can range from one to many depending on value of variable passed in the web request.[1]

II. Related Work

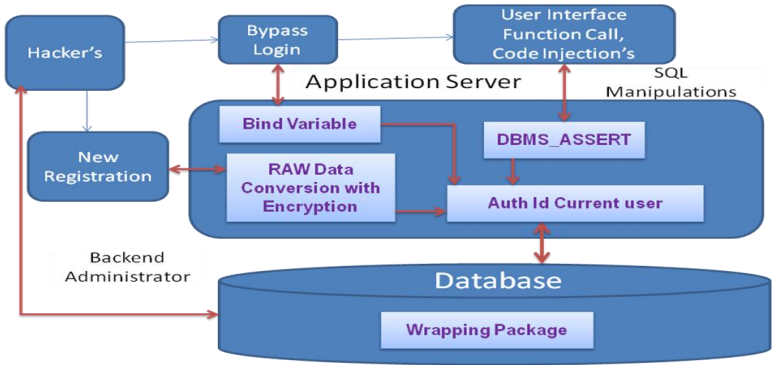
Many different web vulnerability scanners are available on the websites. Three different scanners, *X-Scan, Wikto, HP WebInspect,* have been tested, but *Acunetix Web Vulnerability Scanner,* was finally chosen to scan though UBC websites based on the following reasoning: 1) They generate report which contain the details of the scanning result 2) they scan more then just one type of vulnerabilities.[3] These
attacks are a serious threat to any Web application that receives input from users and incorporates it into SQL queries to an underlying database. Most Web applications used on the Internet or within enterprise systems work this way and could therefore be vulnerable to SQL injection.[5] One of the aspects that contribute to security problems seems to be related to how bad different programming languages are in terms of propensity for mistakes. A common security problems related to the easiness in programming with PHP and its features, but this affects many other programming languages. The choice of the type system (strong or weak) and the type checking (static or dynamic) of the programming language also affects the robustness of the software.[11].Web application vulnerabilities have been addressed by recent studies from several points of view, but without any code analysis To overcome the low level of detail of existing vulnerability databases, some researchers proposed

approaches based on the market, instead of on software engineering. The attacker's perspective has also been of some focus in the literature, but mainly through empirical data gathered by the authors highlighting social networking and what could be obtained from attacking specific vulnerabilities. Some studies analyzed the attacks from the victim's perspective, including the proposal of taxonomy to classify attacks based on their similarities and the analysis of attack traces from HoneyPots to separate the attack types. There is, however, a lack of knowledge about existing exploits and their correlation with the vulnerabilities. To improve software quality, developers need a deeper knowledge about the software faults that must be mitigated. The underlying idea is that knowing the root cause of software defects helps removing their source, therefore contributing to the quality improvement.[11].

III. The Proposed Scheme

Our proposed scheme consist of three techniques to handle the disadvantages of existing system. Techniques like Background administrator ,Bypass login, SQL manipulation, Function call injection, Accessing secured information transfer of schema information which avoid unauthorized access to the application by using RAW Data Conversion Algorithm. So User cannot get secure information from database. We present Double Guard, a system used to detect attacks in multi tiered web services. Using the Bind variable concept we can avoid the Bypasses Login process. User cannot add sql statement to the existing sql statement by using DBMS_Asset Package. User cannot call oracle function or custom function. The system proposed to avoid both web Attacks and database Attacks to achieve more accurate.

A) ARCHITECTURE FOR PROPOSED SYSTEM



The architecture depicts the process of how the problem of bypass login, user interface function calls unknown registrations. These problems occur only at the front end tools as per the architecture the security system is enhanced at the back end in the data base and in the application server the database used here is oracle 10g which is used to store data in form of grids. The front end coding are generally using modified PL/SQL .

3.1) BACKGROUND ADMINISTRATOR

We secure our sensitive information and passwords from database attack using RAW Data Conversion and data encryption process. Cryptography is widely used in web applications. Imagine that a developer decided to write a simple cryptography algorithm to sign a user in from site A to site B automatically sing an MD5 hash function that comprises: *Hash {username: date} [13]*

MD5 should not be used, due to known collision attacks, but it is ok the use with at least 128 bit key. //generate random anti CSRF token

```
$csrfToken = md5(uniqid(rand(), TRUE));
//set the token as in the session data
$_SESSION['antiCsrf'] = $csrfToken;
//Transfer form with the hidden field
$form = '
<form name="transferForm" action="confirm.php" method="POST">
<div class="box">
<h1>BANK XYZ - Confirm Transfer</h1>
520
<p>
Do You want to confirm a transfer of <b>'. $_REQUEST['amount'] .' €</b> to
account: <b>'. $_REQUEST['account'] .'</b> ?
</p>
<label>
```

```
<input type="hidden" name="amount" value="' . $_REQUEST['amount'] .
"' />
<input type="hidden" name="account" value="' .
$_REQUEST['account'] . "' />
<input type="hidden" name="antiCsrf" value="' . $csrfToken . "' />
<input type="submit" class="button" value="Transfer Money" />
</label>
</div>
</form>';
```

In the last step are planned security controls and then, if is all ok, the transfer is done.[13]

3.2) ADMINISTRATOR BYPASSING USER ACCOUNT

SQLIA is hacking technique which the attacker adds SQL statements through a web application's input field or hidden parameter to access to resources. Lack of input validation in web applications causes hacker to be successful. Basically SQL process structured in three phases: i. An attack sends the malicious HTTP request to the web application. ii. Create the SQL Statements. iii. Submits the SQL statements to the back end database By this attack, intruders try to bypass database and application By this attack, intruders try to bypass database and application authentication mechanisms. Once it has been over passed, such mechanisms could allow the intruder to assume the rights and privileges associated with another application user. [6] .To avoid passing hard-coded search values as constants to Top-XQuery, users may use PASSING bind variable parameters as shown the example below: [12]

```
SELECT * FROM XMLTABLE (
'for $i in fn:collection("oradb:/SCOTT/EMP")
where $i/ROW[EMPNO = xs:decimal($empno)]
return $i'
PASSING :1
 as "empno")
```

3.3) FUNCTION CALL INJECTION

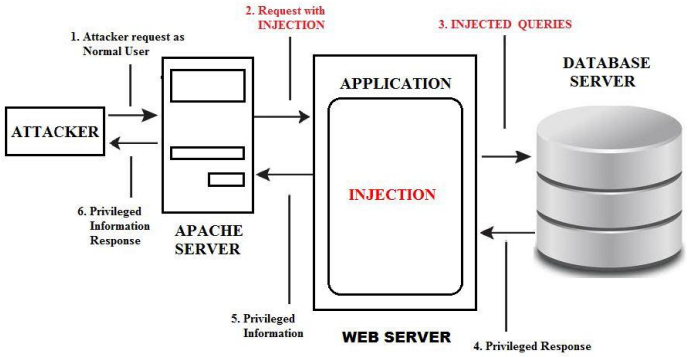
Piggy-backed Query Attack Intent: Extracting data, adding or modifying data, performing denial of service, executing remote commands ***Description:*** In the piggy-backed Query attacker tries to append additional queries to the original query string. On the successful attack the database receives and executes a query string that contains multiple distinct queries. In this method the first query is original whereas the subsequent queries are injected. This attack is very dangerous; attacker can use it to inject virtually any type of SQL command. For example, SELECT * FROM user WHERE id='admin' AND password='1234'; DROP TABLE user; --'; Here database treats above query string as two query separated by ";" and executes both. The second sub query is malicious query and it causes the database to drop the user table in the database.

***Union query: Attack Intent*:** Bypassing Authentication, extracting data.

Description: Union query injection is called as statement injection attack. In this attack attacker insert additional statement into the original SQL statement. This attack can be done by inserting either a UNION query or a statement of the form ";< SQL statement >" into vulnerable parameter. The output of this attack is that the database returns a dataset that is the union of the results of the original query with the results of the injected query. For example,

SELECT * FROM user WHERE id='1111' UNION SELECT * FROM member WHERE id='admin' --' AND password='1234';

Performing privilege escalation, performing denial of service, executing remote commands.In this technique, attacker focuses on the stored procedures which are present in the database system. Stored procedures run directly by the database engine. Stored procedure is nothing but a code and it can be vulnerable as program code. For authorized/unauthorized user the stored procedure returns true/false. As an SQLIA, intruder input "; SHUTDOWN; --" for username or password. Then the stored procedure generates the following query: For example, SELECT accounts FROM users WHERE login= '1111' AND pass='1234 '; SHUTDOWN;--; This type of attack works as piggyback attack. The first original query is executed and consequently the second query which is illegitimate is executed and causes database shut down. So, it is considerable that stored procedures are as vulnerable as web application code. To overcome this problem DBMS assert package is used When the user enters the user name and password it will be validated. If the username and password is correct then the next page will be redirected. Else the error page will be redirected. When the customer enters SQL queries in the destination field then an error message (validated by dbms_assert package) will be displayed.[7]



SQL injection block [1]

V. Conclusion

In this paper we propose a multi tier web security in which first the raw data is converted into an md5 key and stored in bind variable. Other SQL injections are handled by DBMS_ASSERT package hence a multi level of security is provided to web applications that deal with multiple processing like baking online purchase and so on. It is more effective in securing data from back end never the less worrying about the front end security and coding.

References

[1]. DoubleGuard: Detecting & Preventing Intrusions in Multitier Web Applications, Volume 2, No.2, February – March 2013 International Journal of Networks and Systems Available Online at http://warse.org/pdfs/2013/ijns02222013.pdf
[2]. Enhanced Security Model for SQL Injection attacks for web Database Deepti U Telang Prasanna Kumar H.R Research Scholar, Dept.of CSE, HOD, Department of CSE NMAMIT, Nitte India, NMAMIT, Nitte India(International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 6, June 2013).
[3]. Security Analysis of UBC Web Applications (November 2007) Jeffrey Qian, Je-Yu George Lee, William Ha and Phoebe Hsu, University of British Columbia
[4]. Cross-Site Scripting Attacks in Social Network APIs Yuqing Zhang , Xiali Wang University of Chinese Academy of Sciences Beijing, China, zhangyq@ucas.ac.cn
[5]. A Classification of SQL Injection Attacks and Countermeasures William G.J. Halfond, Jeremy Viegas, and Alessandro Orso College of Computing Georgia Institute of Technology {whalfond|jeremyv|orso}@cc.gatech.edu
[6]. A Survey on SQL Injection attacks, their Detection and Prevention Techniques V. ISSN:2319-7242 Volume 2 Issue 4 April, 2013 Page No. 886-905
[7]. www.jiarm.com Testing For Sql Attacks And System Implementation S.Rajasekar* *Doctoral Research Scholar, Dept. of Computer Science, CMJ University, India
[8]. IEEE Transactions On Dependable And Secure Computing, Vol. 1, No. 1, January-March 2004 Basic Concepts and Taxonomy of Dependable and Secure Computing Algirdas Avi_zienis, Fellow, IEEE, Jean-Claude Laprie, Brian Randell, and Carl Landwehr, Senior Member, IEEE
[9]. Acunetix,A Complete guide to securing a website
[10]. International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue
[11]. November 2013) "Intrusion Detection Using Double Guard In Multi-Tier Web Applications": A Survey Sagar Salunke1, Prof. Vani Hiremani2, Kamlesh Jetha3
[12]. Analysis of Field Data on Web Security Vulnerabilities José Fonseca, Nuno Seixas, Marco Vieira, Henrique Madeira
[13]. An Oracle White Paper January 2013 Oracle XML DB: Best Practices to Get Optimal Performance out of XML Queries
[14]. OWASP Testing Guide 2013: ALPHA OWASP Foundation 2013