SECURITY OF SENSITIVE DATA IN XML OR FILE SYSTEM USING ENCODING THROUGH URL

[1] Kajal Shukla, [2] S. K. Singh

[1] M. Tech. Student, [2] Professor VIET, Dadri G. B. Nagar UP (India)

ABSTRACT

Database services have the web applications which are interactive targeted by an SQL Injection. User gives some data as a input and at last that coded input data is being used as to form SQL statement at runtime in these applications. A person who is a n attacker can be able to input a malicious or harmful query segment when user inputs any SQL statement during SQL Injection attacks, that is the result which could be used in many more different database request. Sensitive/Confidential information can be added or modified by an attacker to form attacks of SQL Injection. SQL Injection vulnerability could be used by an attacker as an IP scanner rudimentary. There are several paper published in literature having discussed that how to secure sensitive data in xml or file system , by checking SQL dynamic query commands.SQL Injection attacks However, for secure stored procedures in the database higher level layer / application layer a very less attention is given, which surely can be too suffered from attacks of SQL Injection .

Keywords*: *SQL query, SQL server, SQL Injection.

I. INTRODUCTION

One of the most demanding and challenging causes which can make impacts on the business and industry level in a Structured Query Language is that it can explore all of the sensitive information which is stored in our database, including most highly important information such as credit card details, usernames, addresses, passwords, names, phone, email id etc. To inject a Structured Query Language is the liability that when attacker gets the ability with SQL queries which can be passes to a database. The query which is passed through an attacker in to the database, an attacker can allows the query to database which is supporting element with database and our operating system. SQL Query which is able to accept the inputs from the attacker sides can harms our real web application. Attacker always try to insert harmful SQL query commands into a database so as on execution the query they can destroy or alter the database i.e. this technique is called code injection technique. So this attacker is also called attack vector for websites and this kind of attacker is used by any kind of SQL database.

According the study last year, Security Company **"**Imperva**"** find that the most web application attacks is done 4 times per month and other side retailers company is attacked by 2 times per month. That is not a good practice on the behalf of security.

II.TYPES OF SQL INJECTION

- Redirection and reshaping of a query
- Based on Error message
- Blind injection

2.1 Blind SQL Injection

- Formation of queries that results in Boolean values, and interprets output of HTML Pages is provided by Blind SQL injection technique IN database.
- Final result of SQL injection gives significant data theft and/or attacks in data modification.
- Essentially Blind attack playing 20 questions with the web server.

2.2 Focus on Blind SQL Injection

- This type of SQL Injection is as common as any other type of injection.
- An incorrect or wrong sense of security on the host is provided by the Blind holes.
- Requires a larger of time investment to properly execute manual penetration against.

Fig.1

2.3 Concepts of SQL Injection Attacks

- SQL injection attack is a process to find the query which is entered through the user for execution the command.
- SQL attackers create crafted manually input data so that SQL interpreter has to accept the query and give the permission to execute the commands and give his desired results.
- SQL Injection attack breaks the security of the database layer. When attacker breaks flaws through the SQL injection then attackers can drop, modify, create, and alter our sensitive database.

III. SECURITY IN SQL INJECTIONS

Web vulnerabilities minimum 20% of all that is being related to SQL Injection, called as the one of the most widespread type of catalog application security and as well as the subsequent most common software susceptibility which have the find and prevent capability .SQL Injection always must be on high priority for web developer and also for security basis. Generally a SQL Injection assault diminishes any web network application software which has not provided a proper validation or we can say coded by a user given input data. In the last phase that crafted input data is being used as an element of query over again whichever back-end database. Acquire an example, what time we generate any form it always asks for the ID that is called as identity. After that URL:"http://www.anywebsite.com/id/id.asp?id=anymanualdata" is created.

An invader, using the SQL Injection may perhaps go through any data or "1=1". At the particular time if the application software of the web network is not specified proper validation or incorrectly encoded the user given data that is directly send in the direction of database, and as well as input with the vulnerable query will reach there in reply that will depict every single one ids in the database ever since the condition is "1=1" is for all time true. The example given is an indispensable example however it illustrates the consequence of sanitizing client input data prior to using it in a SQL database query or SQL commands.

IV. LITERATURE REVIEW

Our web applications allow the visitors to enter or submit or retrieve database using any web browser through the internet. These kinds of data have to centralize therefore they be capable of storing data which is needed for websites. If any Suppliers, Employees, a host of stakeholders, customers etc. want to achieve specific content from database side then he can receive it

.Company statistics, User Details, financial, economical and payment information etc are stored in our database which is access through custom web applications. Our Database and Web applications allow us to run the production business frequently.

The Process which attempt to get ahead of commands or statements of SQL intended for implementation through the database over the web network application is called hacking technique in SQL Injection. If their attempts are right then our database allows hackers to view their desire information from the database and he can hampers our database, and be able to do the whole lot which he wants.

For example Like feedback forms , Shopping carts , Search pages , product and support request forms, figure current websites , provide businesses and login pages etc pages are very necessary to commune with customers for keep our customer in touch. These kinds of pages of websites are very to use customer. These types are pages are suspicious for SQL hackers and foremost they attempts to try on these pages .We cannot hide this category of pages on website. If we do it then our client cannot be handling with us. So hacking the website is becoming very easy task for Hackers.

For Simple Example

To access the catalog database , normal user would input their username and password to come into their profile and access his personal details and change the contents which is allow by the administrative section i.e. authenticate user

are allowed to access our database. In other sides, our web network application which controls the authentication page will foremost communicate with our database through the specific planned commands as a result they be able to filter that he is authenticate user or not. In the case of valid user, database allows to access the contents.

In other sides, In case of SQL Injection, Specifically craft SQL commands with the intent of passing the login form difficulty is inputted by the hacker. In case of SQL Injection vulnerabilities, Hackers are eligible to converse with our database directly. Script languages which are Dynamic like JSP, PHP, and ASP.NET, CGI etc are the target technologies by the hacker. For publicity, our website wishes to be communal public so our safety mechanism will agree to to be communal public with our application (generally at beyond port 80/443).

SELECT count (*)

FROM person_list_table

WHERE
username=**'FIELD_USERNAME'** AND password=**'FIELD_PASSWORD"**

This SQL command is given instruction to the catalog database to compare User Id and secret code (password) filled by the current user to the combination that it has already stored in its database. Each and every web network application is hardly coded with specified SQL query so as to it will implement when executing functions and communicating with the database. If any data input of web network application is not accurately encoded, a hacker possibly will introduce extra vulnerable SQL queries which enlarge the area of SQL commands.

An attacker will therefore have a plain channel of communiqué to the web application database irrespective of the entire intrusion uncovering systems vulnerability and network based security equipments installed on the database layer.

V. SQL INJECTION IMPACT

When a hacker feels that a organism is ready to SQL Injection attacks, he is now able to insert SQL Commands to the n/w database an input from field. This is similarly like as to say attacker comes to make changes in our catalog and allow him to do insert or delete like DROP in to database. Execution of illogical SQL queries on the susceptible structure may be done by an attacker. This may break the reliability of your secure information. It depends on the back-end database, SQL injection vulnerabilities can be lead to varying levels of data/system access to the attacker. Manipulate in any existing queries, to UNION that is used to select related information from two tables use sub-selects arbitrary data, or append additional queries.

Some of the SQL Servers like Microsoft SQL Server contains stored and extended procedures for database server functions. In certain cases, it can be possible to read in or write out in files, and can execute shell commands on the underlying operating system. Data is being stolen through the various attacks at all the time. Hackers rarely get caught which are more expert.

Any attacker that can obtain access, it could spell disaster. A SQL injection attacks involves the modification of SQL statements that are used in a web application through the use of attacker-input data. Unfortunately the harm of SQL Injection is only found when the theft is discovered. Improper validation and improper construction and incorrect input of SQL statements in web applications can lead them theft to SQL injection attacks. Thus SQL injection is a potentially destructive and prevalent attack that the Open Web Application Security Project (OWASP) listed it as the number one threat to web applications.

VI. PROPOSED SOLUTION

SQL injection can helps to retrieve sensitive information like password or credit card details, to prevent SQL injection developer should has to take some measure steps like use session in place of query string to transfer value from one page to another. Store sensitive information like password or credit card to XML or file system which is not easily accessible. If using Query

String is necessary try using URL Encoding technique. Now a day's some DBMS like MS SQL server supports Regular expression validation which protect data insertion like " ' ". All DBMS doesn't support """handle it is very Necessary replace it with some other character.

Blindfolded SQL Injection techniques

(a) Boolean queries and WAIT FOR DELAY are used by Blind folded injection technique.
(b) By using commands such as BETWEEN, LIKE, IS NULL Comparison in queries is done.

IDS signature evasive SQL Injection techniques

(a) By using CONVERT & CAST commands by masking the attack payload.
(b) By using Null bytes to break the signature patterns.
(c) By using HEX encoding mixtures.
(d) By using SQL CHAR ( ) to convert ASCII values as numbers.

Example, when the attacker decided to go with a attack using: 1 = 1, at that time when it is entered as input box. The server recognizes 1 = 1 as a true statement and -- symbol is used for comment, everything after that is ignored making it possible to the attacker to access to the database. Through this SQL injection example page you can see precisely how this attack works on:

Welcome to SQL Injection Application

Logged in as: or '1=1--'AND Password='

Other sample pages:

BadProductList- Product List that is vulnerable to SQL Injection.

BetterProductList- Product List that is still vulnerable but that uses a lower privilege account to minimize

damage.

EncryptCnxString- Utility for encrypting any string: use it to encrypt cnxWindBest connection string in web config.

AddSecureUser- Add new users to Secure User table: Password will be hashed use it with BestLoginaspx.

PRODUCT LIST:

Product **Filter: 'UPDATE** Products SET Unit Price=0.0

Product Id	**Product Name**	**Quantity Per Unit**	**Unit Price**
1.	pen	10 boxes*20 bags	0.0000
2.	Alcohol	24-12 oz bottles	19.0000
3.	paper	36 boxes	21.3500
4.	Aniseed Syrup	12-550 ml bottle	0.0111
5.	Seasoning	48-6 oz jars	22.0000
6.	Jelly	12-8 oz jars	25.0000
7.	Uncle Bob's Organic Pears	12-1 ib pkgs.	30.0000
8	Cranberry Sauce	12-12 oz jars	40.0000
9.	Jam	18-500 g pkgs.	97.0000
10.	Pickle	12-200 ml jars	31.0000

WHERE Product=Set Filter

Fig.2

OUR ALGORITHM STEPS OF URL ENCODING ARE

string strCnx = ConfigurationSettings.AppSettings["cnxNWindBad"]; SqlConnection cnx = new SqlConnection(strCnx); cnx.Open();
string strQry = "SELECT Count(*) FROM Users WHERE UserName='" + txtUser.Text + "' AND Password='" + txtPassword.Text + "'";

int intRecs;
SqlCommand cmd = new SqlCommand(strQry, cnx); cmd.CommandType= CommandType.Text;
intRecs = (int) cmd.ExecuteScalar(); if (intRecs>0)

{

FormsAuthentication.RedirectFromLoginPage(txtUser.T ext, false);

}

else

{

lblMsg.Text = "Login attempt failed.";

}

cnx.Close();

//Prevention string strCnx = ConfigurationSettings.AppSettings["cnxNWindBetter"]; using(SqlConnection cnx = new SqlConnection(strCnx))

{

cnx.Open(); SqlCommand cmd = new SqlCommand("procVerifyUser", cnx); cmd.CommandType= CommandType.StoredProcedure; SqlParameter prm = new SqlParameter("@username",SqlDbType.VarChar,50); prm.Direction=ParameterDirection.Input; prm.Value = txtUser.Text; cmd.Parameters.Add(prm); prm = new

```
SqlParameter("@password",SqlDbType.VarChar,50);

prm.Direction=ParameterDirection.Input; prm.Value = txtPassword.Text; cmd.Parameters.Add(prm);

string strAccessLevel = (string) cmd.ExecuteScalar(); if (strAccessLevel.Length>0)

{
FormsAuthentication.RedirectFromLoginPage(txtUser.T ext, false);
}

else

{

lblMsg.Text = "Login attempt failed.";

}

}
```

VII. CONCLUSION

SQL attackers create crafted input data so that SQL interpreter has to accept the query and give the permission to execute the commands and give his desired results. SQL Injection attack breaks the security in the database layer and can alter, steal or destroy our database through using web application.

REFERENCES

(1) Ke Wei, M. Muthuprasanna, Suraj Kothari , Dept. of Electrical and Computer Engineering , Iowa State University Ames, IA – 50011 ,Email: {weike,muthu,kothari}@iastate.edu

(2) Cerrudo. Manipulating Microsoft sql server using sql injection.

(3) http://www.appsecinc.com/presentations/Mani pulating SQL Server Using SQL Injection.pdf, White Paper.

(4) William G.J. Halfond, Jeremy Viegas, and Alessandro Orso College of Computing Georgia Institute of

Technology {whalfond|jeremyv|orso}@cc.gatech.edu

(5) Z. Su and G. Wassermann. The Essence of Command Injection Attacks in Web Applications. In The 33rd Annual Symposium on Principles of Programming Languages (POPL 2006), Jan. 2006.

(6) F. Valeur, D. Mutz, and G. Vigna. A Learning-Based Approach to the Detection of SQL Attacks. In Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA), Vienna, Austria, July 2005.

(7) T. M. D. Network. Request.servervariables collection. Technical report, Microsoft Corporation, 2005. http://msdn.microsoft.com/library/default. asp?url=/library/en-us/iissdk/html/9768ecfe-8280-4407-b9c0-844f75508752.asp.

(8) José Fonseca CISUC - Polithecnic Institute of Guarda, Marco Vieira, Henrique Madeira DEI/CISUC - University of Coimbra. Testing and comparing web vulnerability scanning tools for SQL injection and XSS attacks. Retrieved July 10, 2007, from http://ieeexplore.ieee.org

(9) Yuji Kosuga, Kenji Kono, Miyuki Hanaoka Department of Information and Computer Science Keio University. Sania: Syntactic and Semantic Analysis for Automated Testing against SQL Injection. Retrieved November 12, 2007, from IEEE Computer Society. http://ieeexplore.ieee.org

(10) Benjamin Livshits and U´ lfar Erlingsson. Microsoft Research. Using Web Application Construction Frameworks to Protect Against Code Injection Attacks. Retrieved June 14, 2007, from http://ieeexplore.ieee.org