*International Journal of Research in Advent Technology, Vol.3, No.8, August 2015*
*E-ISSN: 2321-9637*

# Use of WCF Services and Combination of Methods for Preventing SQL Injection Attacks

Shreekant S.Wagh, Prof. Sanjivani Deokar
*Department of Computer Engineering, Lokmanya Tilak College of engineering, Mumbai University, Navi Mumbai, India*
*Email:Wagh.shreekant@gmail.com*

**Abstract** - SQL injection is a technique where malicious users can inject SQL commands into an SQL statement through user input. It is among the most common application layer attack techniques used normally. SQL Injection is among topmost attack mechanisms used by malicious user to steal data from organizations. This is one of the types of attack which takes advantage of improper coding to inject SQL commands into form through user input to allow them to gain access to the data. Use of WCF service for avoiding SQL injection attack will implement combination of adaptive methods and tokenization method. Usability of WCF service is allowed to implement code validation check and use it in any applications it also allows combination of any number of methods. Possibility of SQL Injection is when system takes data from online user hence for SQL Injection prevention we have to validate user input only when we are taking input data from user. One of the most known cause of SQL injection is incorrect way of coding and most of the time WCF services is having all code for validation of SQL Injection and methods implemented remotely, code will take data input from end user and pass to WCF service for validation and no need of knowledge about method implementation done in WCF Service and which allow code enhancement independently.

**Index Terms**- SQL Injection; WCF Service; SQLI; Vulnerability; Cross Site Scripting Attack; Injection;

## 1. INTRODUCTION

SQL Injection attack is ranked in Top10 vulnerability attack by According to the Open Web Application Security Project (OWASP). SQL Injection is a type of injection vulnerability in which the attacker tries to inject arbitrary pieces of malicious data into the input fields on web application, which, when processed by the code, intend that information to be executed as a piece of code by the back end database, thereby giving unwanted results which the application developer did not expect. The database back end server can be any relational database using SQL (My SQL, MSSQL and ORACLE etc.). The ability of the attacker to execute code (SQL statements) through vulnerable input parameters encourage him to directly interact with the back end database.

SQL injection is a technique used to take advantage of non-validated input vulnerabilities to pass SQL statements through a Web application for execution by a SQL database. Injector take favour of the fact that programmers often chain together SQL commands with user-provided input parameter, and can therefore inject SQL statement inside these input parameters. The result is that the attacker (injector) can execute arbitrary SQL queries and/or commands on the SQL database server through the Web application.

## 2. METHODS OF SQL INJECTION ATTACK

### *2.1. Tautologies :-*

In a tautology-based attack a code is injected using the conditional operator like AND, OR etc. so that the executing query always give result as TRUE. Injector can do attack using the URL: http://localhost/? Id =10‘‘OR ‘2’=’2. [1]The conditional OR statement will get appended to the SQL statement and the query will always returns TRUE. In this example it will give the following statement: SELECT EMPLOYEE info FROM EMPOYEEMMAST Table WHERE ID = '10' OR ‘2’=’2' .In this type of attack the injected code will always start with a string terminator (‘) followed by a conditional operator (AND or OR) [2].This logical operator will be append by a statement that always returns result as TRUE. So the signature in this attacks is the string terminator (‘), OR and AND a SQL statement that evaluates to TRUE.

### *2.2. Illegal/Logically Incorrect Queries:-*

An attacker gathers important information about the details database of a Web application by injecting illegal or logically incorrect SQL syntax which will make the application return default error pages that reveal vulnerable/injectable parameters to the attacker [4]. This database details empower attacker to next step of SQL injection attacks.

For example:

Sql: SELECT * FROM EMP_MASTER WHERE username = 'AB"' AND password ='BBB"

Result: "Incorrect syntax near 'AB'. Unclosed quotation mark after the character string " AND Password='BBB"."

### *2.3. UNION Query:-*

An attacker injects an UNION SELECT [3] in existing query to trick the application into returning data from a table different from the one that was expected. Below example is a common form using a *single quote* for this attack: Regular SQL statement + "single quote" + UNION SELECT+ <rest of injected query>.

***2.4. Piggy-backed Queries:-***

An attacker injects additional queries into the original query to extract data, for doing denial of service, insert or modify data or execute remote commands [3]. In this case, the injector does not aim to change the original intended query but to include new queries that piggy-back on the original query [2]. As a result, the database execution engine gets multiple SQL queries. Regular SQL query statement is executed without error and subsequent statements are executed to cause the attack [4]. Attacking person is using a common form a *query delimiter* (;) for this attack:

Normal SQL statement + ";" + INSERT (or DROP or DELETE or UPDATE) + <rest of injected query>

***2.5. Blind SQLI:-***

In this attacker executes SQL Injection attacks using the server responds with database server's error messages stating that the SQL Query's syntax is improper. Blind SQL injection[7] is similar as normal SQL Injection difference that when an attacker try to attempts to exploit an application rather than getting a useful error message they get an error page specified by the application programmer instead[4]. This will lower down probability of potential SQL Injection attack but not completely impossible. An attacker can steal data by asking a series of True and False questions through SQL statements.

***2.6. Alternate Encoding:-***

In this technique; attackers modify the injection query by using alternate encoding, such as Unicode, ASCII, hexadecimal etc [2]. because of this approach they can escape from developer's filter. Injector can use char (44) in place of single quote that is a bad character. Example: SELECT * FROM EMP_MASTER WHERE name= " AND pass=' '; exec (char (Ox73687574646j776e)).

In given example developer can use char() function and hexadecimal encoding now even if developer use filter for bad characters system will not identify Bad characters

**3. PROPOSED SYTEM**

**Use Of windows communication foundation (WCF) for avoiding SQL Injection:-**

Existing search sate that till now many of the technique is developed for SQL Injection detection and avoiding, and for different purpose different types of methods is used [5]. Hence depend on requirement we have to use different types of methods. By using WCF Services we can create and customize a common application according to business requirement and allow multiple methods or combination of methods on common platform also we can create WCF service which work independently and use in any application irrespective of its language or platform [2]. Application developer have no need to understand how logic is implemented in WCF service he/she only have to pass parameter (input data) to WCF services.

***System Description*: -** Although SQL injection [2] is risky but we need not to check it on each and every page. It is necessary only where system take input from user will submit data on form will be check by using WCF web service commonly developed for all application. WCF service [7] will have definition of one or more method and developer of system need not to worry about its implementation

-all validation of input data is at WCF service level which based on remote location.

-data input validation is done only when it is needed (when taken from user) otherwise pass data directly to server.

-data execution will be handled at database server.

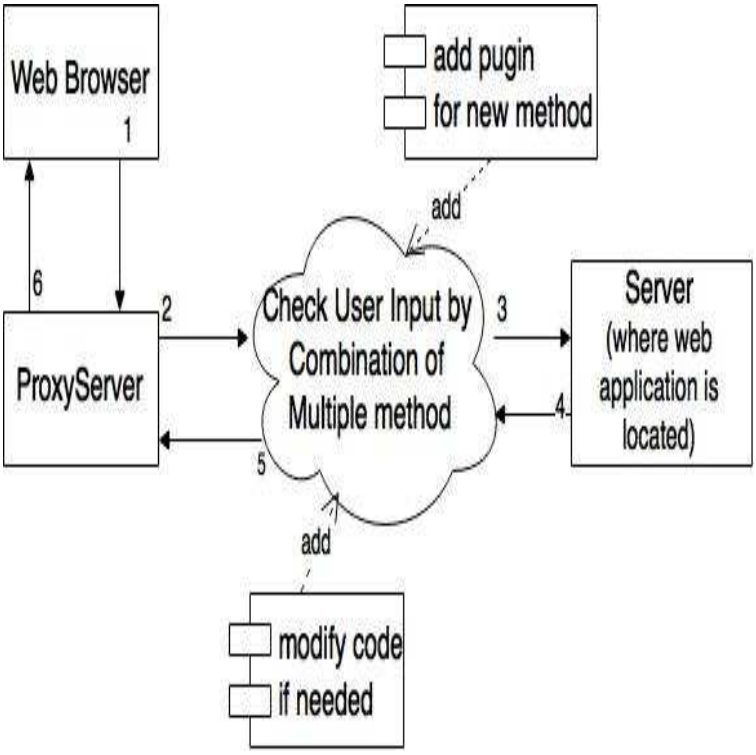


Fig.1. System Diagram

***Architecture*: -** Now days MVC architecture is most popular because of its flexibility and ability to customize application. MVVP [5], MVC architecture will be best suitable to use with WCF Service In which we will keep WCF Service as Separate component with the module so that if any changes occur we can change code of system independently and enhancement will done without affecting existing system or in less down time[4].

In future if programmer wants to do any modification or addition he/she can change code and deploy WCF service which is commonly using in all application.

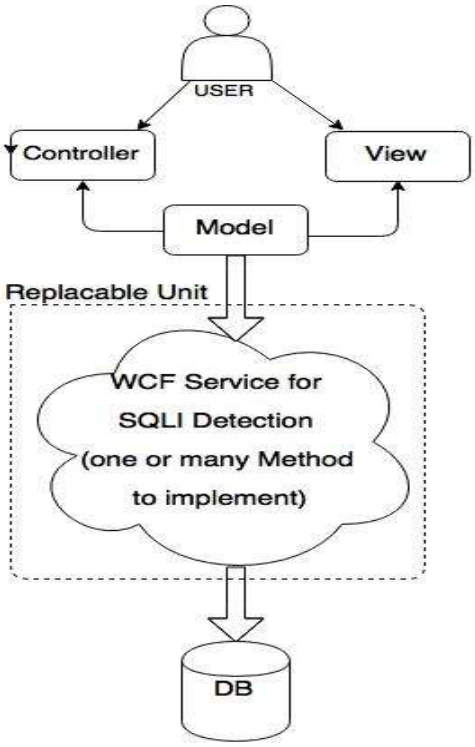


Fig 2. System Architecture

**4. WCF SERVICE WORKING**

WCF Service[6] is having all the methods which developer want to use and combination of methods can be use by developer according to need of application[4]. I have implemented combination of the Tokenization method [8] and adaptive prevention method of SQL Injection

1. Username and password will pass to WCF Service initially. In Tokenization phase WCF Service [2] will create token, username and password will be inspect and detect single code, space, double dash, and all string before single code, before space and before double dashes constitute a token
2. Array will be used to store token
3. First of all obtain length of injected query
    1. If their length is different, get the value as false.
    2. If the length same, get the true value and go the next layer.

Character Lists involved the user input matching process with listed character in the database [6]. After receiving user input for username and password, the validity of string data will be checked with the listed character and tested with the following conditions:

C. If the string exists in listed character, get the value as false.

CI. If the string not exists in listed character, get the value as true and go the next layer.

If the string is weighed as illegal, get result as false else pass to the database

4. Input parameter is passed to database and passed to parametrize stored procedure. if any of cots contain in parameter it will be rejected by stored procedure to execute.

***Why to use WCF services?***

WCF supports more of the WS-* standards than Web Services. Web Service cannot really do streaming or support atomic transactions etc [7]. WCF supports multiple types of binding MSMQ, HTTP, TCP, WSHTTP etc. ASP.NET Web Services support for only HTTP.

Web Services [4] have no instance management in short you cannot have a singleton web service, or web service having session management. You can do state management in web services but it is not that much easy and simple.

WCF is similar to Web Services also incorporates all of the features of .NET Enterprise Services (COM+ Services Components) and Remoting.

It is flexible and extensible like Remoting and also it is compatible like Web Services [8] hence it is providing more features usability.

WCF is consist of lot of different component, you can create new component for authentication and security.

While using WCF, there is no need to do much change in code for implementing; enhancing the security and changing the type of binding small changes in the configuration file will make your requirements.

- WCF is faster than ASMX (Web Services).
- Load balancing & support scaling.
- WCF Service has integrated logging mechanism. Configuration file settings will provide this functionality [9]. Developer doesn't have to write the code again for it.
- WCF services having Interoperability for other languages like java, and more.
- WCF is interoperable with other services if compared with .Net Remoting, where the client and service have to be .Net.
- WCF services provide better reliability and security in compared to ASMX web services.

***Advantages:-***

1.More than one SQL injection defeating technique[5] is implemented in WCF service and hence developer don't have to write code each time he/she have to pass data inputs provided by user i. e.

I. Query Tokenization and Adaptive Method in Preventing SQL Injection [4].

II. Grammar- guided Validation of SQL Injection Sanitizers.

2. In future if any new method wants to implement new method developer can easily and independently in WCF Service.

3. System is cost effective because logic implementations is only on one place and execute one same server.

4. Flexible and work autonomously.

5. Work with any type of application HTTP, HTTPS, TCP, MSMQ and support old technology.

## 5. CONCLUSION

The proposed system is efficiently used for defeating SQL injection and allowed to enhance new method and technology platform. It also give supports to older technology and implementation. By using this system we can reduce cost because we are not incorporated with any new hardware implementation and same WCF Service can be work with any number of the application.

## 6. SCOPE FOR FURTHER DEVELOPMENT

System have WCF Services any one can easily create new algorithm according to use their need .and easily implement in existing system hence up-gradation of system will be possible.

## REFERENCES

[1]. William G.J. Halfond, T. Viegas and A. Orso, "A Classification of SQL Injection Attacks and Countermeasures," in Proc. of ISSSE06, 2006.

[2]. Dr. M. Amutha Prabakar, M.KarthiKeyan, Prof.K. Marimuthu, "An Efficient Technique for Preventing SQL Injection Attack Using Pattern Matching Algorithm," in Proc. of ICECCN, 2013.

[3]. Gao Jiao, Chang-Ming XU and Jing Maohua, "SQLIMW: a new mechanism against SQL-Injection," in Proc. of CSSS, 2012.

[4]. William G.J. Halfond and Alessandro Orso, "AMNESIA: Analysis and Monitoring for Neutralizing SQLInjection Attacks," in Proc. of ASE, 2005, p. 174–183.

[5]. Sruthi Bandhakavi, Bisht, P. Madhusudan, V.N. Venkatakrishnan, "CANDID: Preventing SQL Injection Attacks using Dynamic Candidate Evaluations," in Proc. Of
CCS'07, 2007.

[6]. S. W. Boyd and A. D. Keromytis, "SQLRand: Preventing SQL injection attacks," in Proc. of ACNS, 2004.

[7]. Ke Wei, M. Muthuprasanna and Suraj Kothari, "Preventing SQL Injection attacks in stored procedures," in proceedings of ASWEC, 2006.

[8]. Amit Kukreti. (2005) SQL Injection Attacks [9].http://www.codeproject.com/Articles/11020/SQL-injection-attacks.

[10]. http://www.acunetix.com/websitesecurity/sql-injection/