Proposing an innovative TCS Key as a prevention measure from the SQL injection attacks and vulnerabilities

Dr. Amit Chaturvedi  
Assistant Prof., MCA Deptt.,  
Govt. Engineering College, Ajmer

Shailendra Bagdi  
M.Tech. Scholar  
Bhagwant University, Ajmer

Abstract: Due to the various Web server vulnerabilities and procedure of the inflexibility leads to a Web server script for attacks was increasing. These attacks mostly use ASP or PHP scripting injections. Website rapid expansion based on both ASP and PHP are also slowly becomes the mainstream. Attacks through SQL injection use the insert harmful character in their attack technology. The attacker using input data legitimacy detection not strictly or not detection characteristics, deliberately in a different way from client submit special code to manipulate data, thus collection procedures and server information, obtain the desired information. In this paper, we are proposing a new interface of the user login page and a methodology of generating the secret key. That will improves the security system of the websites have important or confidential information of the users like banks, reservation system, etc.

I. Introduction

With the spread of the Internet and the WEB’s rapid development, WEB applications not only improved the efficiency of work and enterprise strengthens the enterprise market competitiveness. Web platform have flexible, efficient, low cost and other information superiority has greatly improved the related department work efficiency, and promote the actual business thorough development, enhance the department and the outside world exchange, service and interaction. Our country's computer industry after more than ten years of development, the national industry production management system, are based on the Internet architecture, in the country's defense engineering, government office, financial systems, to network games, online banking, network transactions, is inseparable from the network. Today's Internet has become the indispensable part in life. How to effectively ensure network stability and safety operation is an important topic, also is the various network managers have a headache problem. SQL injection is a basic attack used either to gain unauthorized access to a database or to retrieve information directly from the database. They are used most often to attack databases and for extracting any confidential information such as Credit card information, Social Security numbers etc. Web applications are at highest risk to attack since often an attacker can exploit SQL injection vulnerabilities remotely without any proper database or application authentication. An application is vulnerable to SQL injection for only one reason – end user input string is not properly validated and is passed to a dynamic SQL statement without any such validation. If we are sanitizing the user input, then indirectly we are restricting them to not entering single quotes and double quotes in the input.

SQL injection is too much vulnerable that it can bypass many traditional security layers like Firewall, encryption, and traditional intrusion detection systems. SQL injection can not only be used for violating the security by seeing the private data of the people but also can be used for bypassing the authentication of user which is a big flaw in the web applications. Normally, web applications is a three tier architecture, the Application tier at the user side, Middle tier which converts the user queries into the SQL format, and the backend database server which stores the user data as well as the user’s authentication table Till now we have so many solutions to prevent SQLIA. Those are using bind variables, proper input validation, customized error messages, limiting database permissions. Any program or application may be vulnerable to SQL injection including stored procedures executed with a direct database connection. Write the stored procedure in one way, you can prevent SQL injection. Write it in another way, and you are still vulnerable to SQL injection.
II. Methods to Detect and Prevent SQLIAs

Many research authors explored a number of methods to detect and prevent SQLIAs; the most chosen techniques are static analysis, dynamic analysis, combined static and dynamic analysis, web framework, defensive programming and machine learning techniques. The method of static analysis are extreme were it analyzes the code for vulnerability by without actually executing the code. Software metrics and reverse engineering are some forms of static analysis. Model checking, data flow analysis, abstract interpretation and use of assertions in source code are the several techniques of static code analysis. The method of dynamic analysis can be performed automatically by the analysis of vulnerabilities during the execution of web applications which avoids thousands of tests by doing several times manually. Example: CANDID tool. Both the techniques have merits and demerits and therefore variations are identified from the efficacy. However the research study analyzed with various existing works and it has been proved dynamic analysis (penetration testing) tool is effective to test the web applications [2,5,9]. Penetration testing tools are easy to use and assure to provide security information systems to their users by fixing the security weaknesses before they get exposed. The major advantages of penetration (dynamic) testing are: (a) Not necessary to change the development lifecycle (b) Avoids static analysis challenges (c) No need for the source code, (d) Deployment-security.

The method of combined static and dynamic analysis can compensate the limitations of each method, which is considered as highly proficient against SQLIAs but it is very complicated. One of the best examples for such a method is AMNESIA tool. It uses static analysis to analyze the web-application code and automatically build a model of the legitimate queries that the application can generate. At runtime, the technique monitors all dynamically-generated queries and checks them for compliance with the statically-generated model. When the technique detects a query that violates the model, it classifies the query as an attack, prevents it from accessing the database, and logs the attack information. The web framework method is a filtering method of user input parameters. This method is proven to be in-effective while it is not able filter some special characters. The machine-learning method is the most commonly used method whereas the method results in high false positives and low detection rate. Example: WAVES tool [2, 6, 10, 12, 13].

III. SQL Injection Vulnerability Problems

SQL injection vulnerability results from the fact that most web application developers do not apply user input validation and they are not aware about the consequences of such practices [7]. This inappropriate programming practices enable the attackers to trick the system by executing malicious SQL commands to manipulate the backend database [7, 2]. One of the most important properties of SQL injection attack is that it is easy to be launched and difficult to be avoided. These factors make this kind of attack preferred by most cyber criminals, and it is getting more attention in the recent years [2]. Furthermore, the available scanning tools have limited features in shaping efficient attacking patterns which are required to detect hidden SQL injection vulnerability [7, 2]. Moreover, the available scanning tools use brute force techniques to extract data from the targeted websites. These tools do not show meaningful and detailed information about the detected vulnerability. Obtaining this critical detailed information would be very useful for web developers who are not aware about hacking techniques in helping them to fix the bugs, thus to eliminating these vulnerabilities. The lack of penetration testing scanning tools built with enhanced features that are able to conduct efficient penetration test is the main problem addressed in this study.

Therefore, this study aim is in developing a web scanning tool with enhanced features to detect SQL injection vulnerabilities of website databases using different types of attacking patterns, vectors and modes in shaping the attacks. The SQL injection vulnerability problems can be addressed by developing a new web scanning (MySQLlInjector) tool with enhanced features that is able to conduct efficient penetration test on PHP based websites to detect SQL injection vulnerabilities and getting the web developers to fix these vulnerabilities. The development of the tool is based on the following steps: (1) review the literature on current penetration testing tools for SQL injection attacks on databases for identifying the important types of attacking patterns, vectors and modes; (2) develop penetration testing styles for SQL injection attack on databases based on the different types of attacking patterns, vectors and modes based on information from the literature; (3) develop a web scanning tool (MySQLlInjector) to detect SQL injection vulnerabilities based on the identified styles; and (4) test and validate the My SQL Injector tools for penetration testing on different websites.
IV. Related Work

Many researchers have proposed various methods for detecting SQL injection. Developers design the code to find the keywords of the input parameters which presents malicious code. Stephen W. Boyd and Angelos D.Keromytis have proposed methods which concatenate the random keywords with legitimate query to avoid the injection. For this method, proxy server is needed and developer knowledge must be essential. In order to detect and prevent SQL Injection attacks, filtering and other detection methods are being researched. This section explains the related work[15]. Indrani Balasundaram proposes technique—Service-Oriented Authentication is to prevent SQL-Injection Attacks in database the deployment of this technique is by appending first level Service has the functionality of Tame-card detection and Prevention. Many techniques have been proposed prevent SQL injection Attacks for example, dynamic monitoring tools.

Various SQLIA detection techniques for the application layer have been proposed in literature, but none of them pay enough attention to SQLIA in stored procedures by providing enough security. Many existing techniques, such as filtering, information-flow analysis, penetration testing, and defensive coding, can detect and prevent subset of vulnerabilities that lead to SQLIAs. A number of techniques are in use for securing the web applications. The most common way is the authentication process through the username and password. One of the major problems in the authentication process is the input validation checking. Most of the papers are restricting the user by not entering single quotes and double quotes in the user fields. Here in this paper we will concentrate on encrypt user entered data and passed to the stored procedure so that SQL Injection will be prevented. Various techniques have been proposed for controlling SQL injection attacks, for example, Ke Wei [1] is using static analysis and runtime analysis in order to detect and prevent the SQL-Injections in stored procedures with the help of Sqliachecker() function that identify the user input by the current session id and build a finite state automata.

In AMNESIA (Analysis and Monitoring for Neutralizing SQL Injection attacks) (Halfond and Orso 2005) [11], the authors are using runtime checking of the query and declare it valid or malicious. AMNESIA checks query in different steps. In the first step it identifies the “hotspot”. Hotspots are application code which issues SQL query to database. Second, it forms a model for legitimate query in the form of NDFA (Non-Deterministic Finite Automata). Finally, as a request comes it checks the query with NDFA and declares it legitimate or malicious. M. Junjin [12] is using an approach for the detection of SQL injection vulnerabilities. The above mentioned author has used static, dynamic and automatic testing method for the detection of SQL Injection vulnerabilities. The approach traces user queries to vulnerable location. Although these techniques are effective, they cannot capture more general forms of SQLIAs that generate syntactically and type correct queries. Wassermann and Su combine static analysis with automated reasoning in [10] to detect tautologies in the dynamically generated SQL queries, but the other forms of SQLIAs would still succeed rendering the system vulnerable.

In [3], I. Balasundaram and E.Ramaraj have proposed a technique to prevent SQL-Injection attacks by using encryption but not in stored procedures. The use of stored procedures alone does not protect one against SQLIAs as is commonly assumed by most developers, but appropriate use of parameters along with stored procedures is necessary to achieve a minimal defense against such attacks [8] [11]. Most of existing techniques, such as filtering, information flow analysis, penetration testing, and defensive coding, can detect and prevent a subset of the vulnerabilities that lead to SQLIAs. In this section, we list the most relevant techniques- William G.J.Halfond et al.’s Scheme- [6]- This approach works by combining static analysis and runtime monitoring. In its static part, technique uses program analysis to automatically build a model of the legitimate queries that could be generated by the application. In its dynamic part, technique monitors the dynamically generated queries at runtime and checks them for compliance with the statically generated model. Queries that violate the model represent potential SQLIAs and are thus prevented.
from executing on the database and reported. SAFELI – [3] proposes a Static Analysis Framework in order to detect SQL Injection Vulnerabilities. SAFELI framework aims at identifying the SQL Injection attacks during the compile-time. This static analysis tool has two main advantages. Firstly, it does a White-box Static Analysis and secondly, it uses a Hybrid-Constraint Solver. For the White box Static Analysis, the proposed approach considers the byte-code and deals mainly with strings. For the Hybrid-Constraint Solver, the method implements an efficient string analysis tool which is able to deal with Boolean, integer and string variables.

V. Proposed System

D.R.Rani, B.S.Kumar, L.T.R.Rao, V.T.S. Jagdish, M.Pradeep proposed “Web Security by Preventing SQL Injection Using Encryption in Stored Procedures”. They propose an SQL-Injection Attack prevention technique that works by combining encryption of user entered data within the stored procedure. The basis of such a technique is that effect of the malicious code can be avoided by using encryption algorithm.

After submitting the user registration form, a unique secret key will be generated corresponding to the given username and password. A stored procedure will be called by passing the following parameters:

- Username
- Password and
- Secret key

They refer the basic system Model:

Secret key will be generated by using the combination of the two fields, username and password. First three letters of username & password will combined in random order and then encoded to get a sequence of bits which is the secret key. This will be done dynamically without storing the secret key every time user enters username and password.

**Generation of Secret key**

![Flow Chart of Secret Key Generation](image)

We find a research gap in this key generation method. Instead of using two fields (username, password) for key generation, we should use three fields for key generation (USERNAME, PASSWORD, CURRENT_DATE). This process will generate the new key every day when user works and this will again, it will be changed with the change in date. This may reduce the chance of attacks on the database. This novel approach for key generation is presented in the figure 3.

![First time login screen every day](image)
In figure 3(a), the field Secret Key and Submit Button is shown disabled because when user first time login. It has to first Generate Secret Key by entering the authenticated Username and Password with the log. This key will be send to the user’s authenticated email ID and for the rest of the day this key will be valid. This is the first part of our proposed system. Now, how will system generate this key is totally hidden from the user. Now, we propose a new key generation methodology as given in the figure 4.

![Figure 3(b): login screen after getting a new secret key](image)

In the figure 3(a), the field Secret Key and Submit Button is shown disabled because when user first time login. It has to first Generate Secret Key by entering the authenticated Username and Password with the log. This key will be send to the user’s authenticated email ID and for the rest of the day this key will be valid. This is the first part of our proposed system. Now, how will system generate this key is totally hidden from the user. Now, we propose a new key generation methodology as given in the figure 4.

**Figure 4: Process for creating Secret Key**

As illustrated in figure 4, the user enters username and password everyday on first time accessing the database or may for using the website. The process itself does the whole process of generating the 10-character secret Key [TCS Key] by taking first and last character of the username and password given by the user and including the system date as six digits (ddmmyy). Then the function will combine these 10 alphanumeric string in random order to generate the final string. Then generate a hash string from this 10 alphanumeric string to generate the final string i.e. the 10-Character Secret Key [TCS Key].

This proposed system from our observation is an innovative and more appropriate solution for the prevention of the system from SQL Injection Attacks. Because this secret key will be generated everyday and attacker may the chase the system from the previous secret Key.

**VI. Conclusion**

SQL Injection is one of serious security threat issues for the organizations and businesses operating on the web. Security of data is very important for every organization. SQL injection is a common technique hackers employ to attack underlying databases. The attack alters the SQL queries and behavior of the system for the benefits of Hacker and always try to theft the user login information. If any how he guess or get the correct information, then stealing the information from the database is very easy.

In out proposed system, we have presented an innovative idea of creating the secret key that will be changed and send to the user everyday when first time he logged to the website daily. By using this methodology, the system will be deny the attackers attempt to login with all the previously steal login information. Hence this will improve the security system of the websites.

**References :**


[16]. Abhishek Kumar Baranwal, Approaches to detect SQL Injection and XSS in web applications,EECE 571b,Term Survey paper, April 2012.


