ISSN:0975-9646

SQL Injection Prevention by Blocking Internet Protocol Address after Analysing Error Message of Database Server

Sanjay Mishra[#1], Subodh Mishra[*2], Vivek Sharma[#3]

[#]*TIT College, CSE Department, RGPV*
Bhopal, Madhya Pradesh, India

[*]*TIT College, RGPV*
Bhopal, Madhya Pradesh

***Abstract*— Web applications are used tremendously now-a-days as a means of dissemination for online services having financial consequences such as online payment of bills, e-banking, e-commerce etc. Social networking, emails, e-commerce companies, needs high security, confidentiality, integrity to availability to keep sensitive data. In order to protect these data from unauthorized access the system must be configure and programming in a manner that it's ensure the trustworthiness for storing and publishing of information. SQL Injection Attacks (SQLIAs) is technique in which a portion of malicious SQL query segment is injected as a user input which could result to obtain unrestricted access to confidential information from database. In this paper we will discuss different type of SQLIAs technique and error message given by database server when attackers apply illegal query for the same. And, proposed a novel technique to defend SQLIAs at different level after comparing runtime error message given by database server. If same Internet Protocol (IP) address is used for code injection numbers of time then it build a protective layer which reduce the data leakage and block the IP address.**

***Keywords*— Structure Query Language (SQL), Structure Query Language Injection Attacks (SQLIAs), Internet Protocol (IP) Address, Vulnerabilities, Web Application, Web Security.**

I. INTRODUCTION

The widespread adoption of the web applications are used for variety of purpose like simple thing as accessing of blogs to those having financial consequences and other banking related difficult tasks. As the usages of internet are increasing day by day, the cyber-crime is also increasing. Web application is normally cooperate with database. Hence security has become major concern to protect application against command injection attacks. In 2013 SQLIAs was rated as number one attack by Open Web Application Security Project (OWASP) [1]. In report it showed that most of the security attacks are caused by illegal query into an entry field for execution and can access the confidential data without any authorization. It has been seen that 97% were vulnerable to web attacks. One class of attacks that has been particularly dangerous is SQL injection [2]. There is no any requirement of hacking any network security protocol or breaking any encryption mechanism of the database server and network. SQL injection exploits the vulnerabilities in input validation and force database server to execute the query from backend and return the relevant information to attackers. Sometimes attackers tries to alter the database cause repudiation issue such as voiding transaction or changing balance can make business is in danger. Furthermore, attackers can exploits flows of websites, such as requiring database version, extracting and modification data, bypass the authentication, controlling systems, upgrading users privileges and shutting down database[3]. There are many methods and algorithms are proposed to prevent these injections but there are no any full proof solutions and unfortunately SQLIAs still exits.

II. BACKGROUND

SQL injections are using security vulnerabilities for exploiting the database server. An attacker can extract confidential data like financial, medical and personal information's by input drives of the application logic. The logic of application and malicious query can easily be injected by attackers in form of URL's, cookies after analysing the error message given by the database server. Whenever an illegal query is applied, the different database server response differently for the same query by which an attacker can easily identified the version and type of database that a web application using. This information can allow an attacker to craft a specific attack by which database schema information such as table name, column names, column data types, data value and other valuable information is extracted. Attackers can also alter the schema of database, locking or dropping database tables and shutdown the web application which tends to denying services to other users. During the development of web application developer generally rely on query building with string concatenation to construct SQL statements. When attacks apply queries based on varying condition set by the users, its cause vulnerabilities for SQL injections. Some developer used parameterized query or stored procedure but inappropriate uses of these such as absence of checks and absence or misuse of delimiters in query string cause SQLIAs.

III. SQL INJECTION ATTACKS

SQL injection attacks can be prevented only when we know about its implementation and its phenomenon. Types of SQL injection known to date and its description with attacks examples are discussed below:

A. Tautology

The tautology attack based on injecting code in which the conditional statements "WHERE" clause is evaluated always true. This attack is used to bypass the authentication pages and extracting data because the tautologies ignore all WHERE condition to return all records.
Example: An attackers submits 0' or '0'='0 and sometimes the most common tautology " or 1=1 –" that put whole condition true.
Select ACCOUNTS from USERS where LOGIN = 0' or '0'='0 and PASSWORD = 0' or '0'='0
This code inject transforms the entire WHERE clause into tautology.

B. Illegal/Logically Incorrect Query

This attack is used to gather information about database type and its schema which is consider as preliminary and treated as reconnaissance phase. Whenever, illegal or logically queries inject, an error messages is generated which can reveal vulnerable or injectable parameters by which an attacker can also perform database finger-printing, this is used to deduced the data types of certain column, name of tables etc.
Example: Select ACCOUNTS from USERS where LOGIN = "sanjay" and PASSWORD = "12345";
Error: Warning! mysql_fetch_array(): supplied argument is not a valid MySQL result. Password type not found.
Here error message expose the value of string password is not matching, so some type conversion will use for the same.

C. Union Query

The union query in SQL injection attack is used to exploit the vulnerabilities of database for extracting data when structure of database such as table name and columns names is known. This attack returns a dataset by the database after unioning the query with original SQL query.
Example:
http://www.victims.com/showproduct.php?id=304 union all select 1, 2 - -
Error: All queries in the SQL statement containing a UNION operator must have equal number of expressions in their target lists.
http://www.victims.com/showproduct.php?id=304 union all select 1, 2, 3 - -
No Error
Here, attacker selects his desire information in form three columns. Now, following attack retrieve the desired database name.
http://www.victims.com/showproduct.php?id=-304 select all 1, database_name, 3 - -
By using this query an unauthorized user extract the structure of database and data inside it.

D. Stored Procedures

This type of attacks is used for performing privilege escalation and executing remote command. The method is use by developer as there is misconception that by using the stored procedure SQL injection will be avoided but web application is vulnerable as normal applications. Attack such as denial of service, buffer overflow and for running the arbitrary code on server to escalate their privileges can easily be performed because stored procedure uses special scripting languages [4].

E. Piggy Backed Queries

In this attack, the attackers include new distinct queries into the original query and both the query instead of joining each other it run independently without any modifications in query [5]. This type of attack is only executed when database can run multiple queries in same line.
Example: select Account from USERS where USERNAME="Administrator" and PASSWORD="password"; drop table USERS - - and PIN="123456" - -
Here "- -" is used to ignore rest of the query.

F. Inferences

In this attack attacker inject a query which is being modified, but database is secured enough and do not provide any error message. Then attackers observe the changing behaviours of web application in the forms of function and response i.e. time taken to load the result page of the web applications. This attack can also use to extract the vulnerable parameters with additional information.
Example: http://www.victims.com/-- and If (version () like 2; sleep (20), "false") - -
It shows if database version having number 2 have 20 second delay in response while replaying and loading of page. By the help of this attack an attacker can know the type and version of database management system as "WAITFOR" command only be executed by Microsoft SQL Server or "SLEEP (n)" is used to making delay by Oracle and MySQL servers.

G. Alternate Encoding

In this attack an attackers bypass the filter by modifying the text string in Base64, Hexadecimal, ASCII, and Unicode such as: exec (char (112 97 115 115 119 111 114 100)) - - ; means password.
Suppose a system is configured by an Intrusion Detection System (IDS), with some predefined signature such as 0' or '0'='0 and if it's found then system drop the connection and show error.
Error message: SQL injection attempt is found. Server will block your IP in next attempt.
0' or '0'='0 SQL injection string which is encoding in other form by IDS.
Hexadecimal encoding of 0' or '0'='0 is *302019206f722020183020193d20193020.*
ASCII encoding of 0' or '0'='0 is *48 NULL 32 111 114 32 NULL 48 NULL 61 NULL 48 32.*
Unicode encoding of 0' or '0'='0 is *\u0030\u2019\u006f\u0072\u2018\u0030\u2019\u003d\u2019\u0030.*

IV. Related Work

Many existing techniques, such as input validation, filtering, information-flow analysis, penetration testing and defensive coding such as stored procedure can detect and prevent a subset of the vulnerabilities that lead to SQLIAs. The injected commands will be erroneously identified as the normal SQL command by the database server which will run them, and thus get the system to suffer damage or intrusion [6]. But after studying about all these technique it can be analyse that SQL injection can only be performed if there are flaws in coding in a single line and if some error message is generated by the database server. A Runtime monitor [7] is then developed to observe the path taken by critical variables and check them for compliance with the obtained legal paths. During runtime, if the path taken by the identified critical variables violates the legal paths obtained, this implies that the critical variables consist of the malicious input from the external user and the query formed is trying to access confidential information from the back-end database. This abnormal behaviour of the application, due to the critical variables, is identified by the runtime monitor and immediately notified to the administrator. But the approaches do not detect and prevent all other types of SQL injection Attack. We will discussed a part of their approach when runtime monitor notified to the administrator and at same point an attack pattern analyser we will introduced which store attack pattern with error message given by the database server in the prevention module with user information such as Internet Protocol (IP) address and logging time and logging time and when same pattern is recognized with same IP address then block users IP.

V. Proposed Solution

The proposed approach is consisting of prevention module which is converted with system that store user information such as user IP and logging time. The proposed system architecture consists of four systems that connected in a manner which prevents SQL injection at runtime. Generally writing secure code is best solution while making single mistake makes whole system vulnerable for SQLIAs.

A. Attacks Database

The web application will configure with database of attack pattern that consists of all patterns till that and some predefined keyword that treated as attack patterns. If the attack having matched with the pattern stored or even keyword matched the whole query injected will treated as attack and the Internet Protocol (IP) address of user is saved as unauthorized user

B. Prevention Module

Prevention module is intermediate between all the proposed phases and establishes communication with attacks database, stored system and verification system. If same attacks pattern is applied using same Internet Protocol (IP) address it automatically detect SQLIAs and generate an alarm for database server and display error message of blocking IP to user via verification system.

Error Message: “HTTP-500 Internal Server Error”. SQLIAs Identified! IP Blocked!

C. Stored System for Users

The system is stored the information of unauthorized user with their Internet Protocol (IP) address. If any attack is identified after comparing with predefined attack keywords or pattern of SQLIAs from attacks database. If any matched is found then it automatically stored the IP of user as unauthorised user and detects unauthorized access.

D. Stored System for Users

The system identified the unauthorised user after comparing with data saved in the store system for user information. If stored system authenticates the system validate user otherwise, display error message of unauthorised aces and warn user for applying SQLIAs.
Error Message: Your IP is saved as you are trying to access illegally! Next attempt block your access!

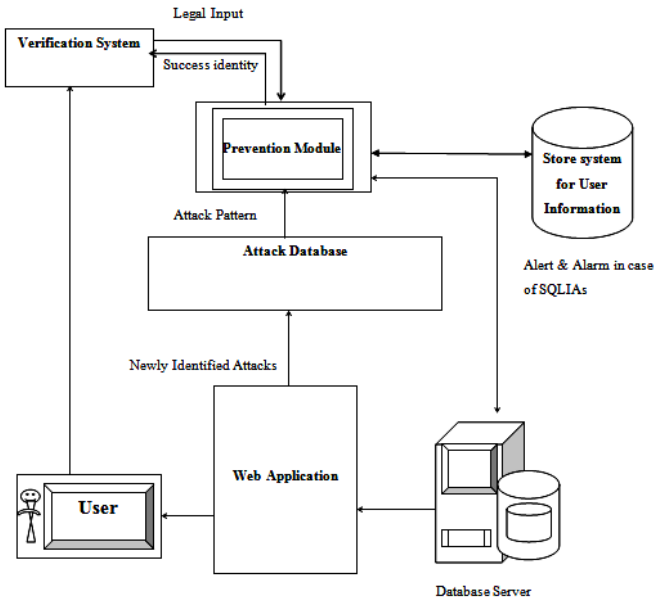

Fig. 1 A proposed Architecture for preventing SQL injection attacks after analysing Error message of database server

Algorithm for Preventing SQLIAs by Analysing Error Message of Database Server

Step 1: Store the input value entered by user in database Server with Internet Protocol (IP) Address.
Step 2: If the string input is matched with database server then authenticate the user else identified input as attack.
Step 3: Attack database matched the input with predefined attacks patterns or keywords for SQLIAs.
Step 4: If only pattern is matched identified it as newly attacks and stored the IP of user as unauthorized in store system after saving the newly attack in database.
Step 5: Prevention Module store attack pattern as intrusion and also identify user as authentic via verification system.
Step 6: If SQLIAs with same Internet Protocol (IP) is injected then,
Step 7: Generate alert and alarm to database server is generated and error message of blocking IP is pop-up for same user.

VI. CONCLUSION

The prevention module prevents SQL injection by an unauthorized user after analysing their injected query. If Attacks Database detected the injecting query as SQLIAs, it automatically save the attack pattern and if it newly identify, one another pattern automatically inbuilt in database for further matching and after onward it's send the report to prevention module. After detection, the Prevention Module forwards all information of user such as login time and IP address for further comparison. And if same pattern of injection or some other SQL injection is identified by the same IP address the Verification Module Display the error message. In this paper we have presented a novel module for preventing the SQL injection after analysing the error message given by the database server followed by blocking the IP of the attacker. The module works on any types and version of the database server. The module considers two things for preventions of SQL injections.

- Best code practices with considering the SQL injections till date in attacks database.
- SQL Injection runtime prevention after analysing Error message given by server.

The mechanism are helpful in detecting different type of malicious activities in the network by combination of good configuration of web server and using the best code in configuring attacks database can increase the protection against SQL Injection attacks.

VII. FUTURE WORK

This method works well to protect our application against all type of SQL Injections detection and the prevention mechanism using Error Message with blocking Internet Protocol (IP) Address of attackers if particular message is detected. But there is still a lot of room for improvement in future is to adding more Error Message to the rule and also detecting proxy IP when attacker using VPN (Virtual Private Network). And also there may be scope of preventing SQLIAs at various level recommended in thesis. Those are our future follow-up research work.

REFERENCES

[1] OWASP Top Ten Project [EB/OL]. [2016-02-20]. https://www.owasp.org/index.php/Category:OWASP_Top_ten_prje ct.

[2] Mehdi Kiani, Andrew Clark, George Mohay: Evaluation of Anomaly Based Character Distribution Model in the Detection of SQL injection Attacks, Third International Conference on Availability, Reliability and Security Volume 00, Issue, 4-7 Page(s):47-55, IEEEE, March 2008.

[3] Meixing Le, Brent ByungHoon Kang: Double Guard-Detecting Intrusions in Multitier Web Applications", IEEE Transaction on Dependable and Secure Computing Vol. 9, No. 4, July/August 2012.

[4] E.M.Fayo: Advanced SQL Injection in Oracle databases, Technical report, Argeniss Information Security, Black Hat Beifings, Black Hat USA; 2005.

[5] A.S.Gadgikar: Preventing SQL Injection Attacks Using Negative Tainting Approach, DOI 10.1109/COMPSAC.2015.277.

[6] SQL injection[EB/OL]. http://en.wikipedia.org/wiki/SQL_injection. [2016-02-20]

[7] Ramya Dharam and Sajjan.G.Shiva: Runtime Monitor to Detect and Prevent Union Query based SQLIAs; DOI 10.1109/ITNG.2013.57, IEEE 2013.