Securing Web Application against SQL Injection Attack: a Review

Ms. Mira K. Sadar,
Department of Comp. Sci.& Engg.(CSE)
SIPNA COET Amravati, India
mirasadar@gmail.com

Mr. Pritish A.Tijare,
Department of Comp. Sci. & Engg.(CSE)
Associate professor in CSE
Department of SIPNA COET
Amravati, India
pritishtijare@rediffmail.com

Mr. Swapnil N.Sawalkar
Department of Comp. Sci. & Engg.(CSE)
Assistant professor in CSE
department of SIPNA COET
Amravati, India
snsawalkar@rediffmail.com

Abstract**—**Nowadays use of web applications are rapidly increasing. We are using various applications to fulfill our daily needs such as online shopping, banking, ticket booking etc. Therefore our private data is available on the databases. Because of rapidly increasing use of web applications, the attacks on web applications are also increasing rapidly.SQL injection is one of the most serious attack on web application. It injects the SQL query and allows attackers to gain unrestricted access to databases underlying applications. In this paper we study about the SQL injection attack and its various types. In addition to these we have study that how SQL injection attacks on web applications can be protected or prevented using different types of predefined techniques and also discuss the integrated approach of encoding methodology with the combination of AES encryption and secure hashing is applied within the database to avoid attack on login section in detail. Lastly we discussed the consequences of SQL injection attacks.

Keywords**—***AES, attacker, query, SQL injection, web application.*

*****

I.INTRODUCTION

In today's world the employment of net become rising and also the ton of advances in technology have simplified our several daily work. The World Wide Web has evolved from a system that delivers static pages to a platform that supports distributed applications, referred to as web applications, , and has become one amongst the foremost rife technologies for data and service delivery. Multiple services are available via single click through various web applications; there is no need to stand in long queues at the banks or market to buy for the modern trends. As web applications are increasingly used to deliver essential services, they become a valuable target for security attacks. Many web applications interact with back-end database systems, which may store sensitive or confidential information such as related to faineance, health etc. Due to these attacks against them increases rapidly. Of those attacks, a serious role is control by SQL injection attacks (SQLIA). SQL injection attack is one amongst the intense threat to web application accustomed gain unauthorized access to database or to retrieve the confidential data present on the database. According to OWASP report released in 2012, SQL Injection attacks are top most risk/danger to Web Applications. [3] In SQL injection attack, attacker provides SQL code rather than the legitimate input in the input fields of the web form in order to vary the meaning of the original SQL query issued by the database at the backend. Once the attacker gains access to the database, it will alter any data.

SQL Injections are attacks by which an attacker changes the structure of the original SQL query by injecting SQL code in the input fields of the web form in order to obtain unauthorized access to the database. SQL injection attacks inject malicious queries by the attackers into the application projected queries to receive the required outputs from the database.SQL Injection allows attacker to create, read, update, modify, or delete data stored within the back-end database. The malicious user injects SQL Commands into SQL statements through the input of online webpage. The injected SQL Commands will compromise the safety of web application. SQLIA will cause nice impact on web applications and conjointly have an effect on the organization to that that web application is belong. so as to run the online application swimmingly over net or in the other network like LAN, WAN, there's basic have to be compelled to stop them with SQLIA.

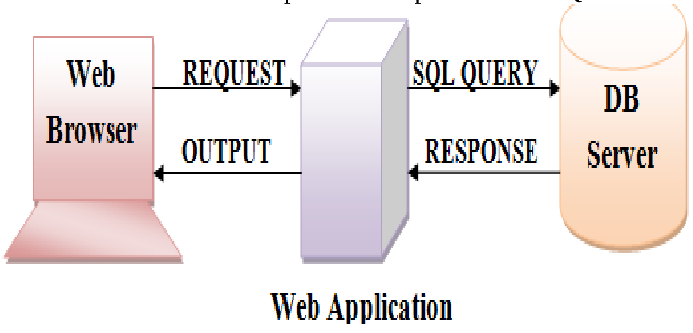

Fig.1 Web application structure

II. METHODS of SQL INJECTION ATTACK

There are diverse methods of attacks that depending on the goal of attacker are performed together or sequentially and its classification is given below:-

A. Tautologies:

The main purpose of tautology-based attack is to inject code in conditional statements so they're indefinitely evaluated as true. Using tautologies, the attacker needs to either bypass authentication or insert inject able parameters or extract information from the database. Whenever a conditional statement is injected with code so the result is true, then its analysis and result depends on the method that the query is evaluated within the application. The attacker mainly focuses on the where clause to inject the code. Transforming the conditional into a tautology causes all of the rows within the info table to be return in order that he will login with success while not having a legitimate username and password. It performs some action if a minimum of one record is projected.

Example:

"SELECT * FROM login WHERE uname = 'mira' and password ='aaa' OR '1 '='1'

The code injected in the conditional (OR '1'='1') transforms the entire WHERE clause into a tautology the query evaluates to true for every row in the table and returns all of them [2].

B. Illegal/Logically Incorrect Queries

The main purpose of the Illegal/Logically Incorrect Queries based SQL Attacks is to assemble the information regarding the back end database of the web Application. When a query is wrong or illegitimate, a error message is returned from the database together with helpful debugging data. This error messages facilitate attacker to find vulnerable parameters within the application and consequently database of the application. Attacker injects junk input or SQL tokens in queries that may cause syntax error, type mismatches, or logical errors by purpose. The syntactic error helps to spot the inject able parameters. The logical errors typically reveal the names of the tables and columns that caused the error [5].

Example:

In this example attacker makes a type mismatch error by injecting the subsequent text into the pin input field:

1) Original URL:
http://www.arch.polimi.it/eventi/?id_nav=886

2) SQLInjection:
http:/`/www.arch.polimi.it/eventi/?id_nav=8864'

Error message showed:
SELECT name FROM Employee WHERE id =8864\' from

the message error we can find out name of table and fields:

name; Employee; id. By the gained information attacker can arrange more strict attacks [5].

C. Union Query

In this technique, attackers join injected query to the original query by the word UNION and then can receive data concerning other tables from the application. The output of this attack is that the database gives a dataset that is the union of the results of the initial query with the results of the injected query
Example:

SELECT Name, Address FROM Users WHERE Id=$id

By injecting the following-

Id value: $id=1 UNION ALL SELECT creditCardNumber,1

FROM CreditCarTable.

We will have the following query: -

SELECT Name, Address FROM Users WHERE Id=1

UNION ALL SELECT creditCardNumber, 1 FROM CreditCarTable

which will join the result of the original query with all the credit card users.[5]

D. Piggy-backed Queries

The attacker needs to execute remote commands or add or modify information. In this style of attack, the attacker doesn't will build changes in the original queries however inject further queries. this can be totally different from alternative types as a result of the attackers don't seem to be making an attempt to inject the original planned query; instead they're making an attempt to incorporate a new and distinct queries that 'piggy-back' on the initial query. Thus, the information receives multiple SQL queries. The primary is that the proposed query by the application that is performed as normal; the succeeding ones area unit the injected queries, that is performed in addition to the primary. If successful, the attackers will nearly insert any style of SQL command and have them execute with the original query. Vulnerability of this type of attack depends on the sort of database.

Example:
In this example, attacker inject " 0; drop table user" into the pin input field instead of logical value. Then the application would produce the query:

SELECT info FROM users WHERE login='doe' AND pin=0; drop table users

Because of ";" character, database accepts both queries and executes them. The second query is illegitimate and can drop users table from the database. It is clear that some databases do not want special separation character in multiple different queries, so for detecting this type of attack, scanning for a special character is not impressive solution [5].

E. Stored Procedure

This type of attack tries to execute stored procedures present in the database with malicious inputs. As stored procedure may well be coded by programmer, so, this part is as inject able as web application forms. Depend upon specific stored procedure on the database there are alternative ways to attack. As an SQL Injection Attack, 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 [5].

F. Inference:

By this sort of attack, intruders modification the behavior of a information or application. There are two standard attack techniques that are based on inference: blind injection and timing attacks.

1. Blind SQL Injection: In this sort of attack, helpful information for exploiting the backend database is collected by inferring from the replies of the page after questioning the server some true/false questions. However, when attacker attempts to use an application, instead of obtaining a helpful error message, they get a generic page mere by the developer instead. This makes exploiting a possible SQL Injection attack tougher however not possible. Associate attacker will still get access to sensitive information by asking a series of True and False queries through SQL statements.

SELECT accounts FROM users WHERE login= 'doe' and 1

=0 -- AND pass = AND pin=O SELECT accounts FROM

users WHERE login= 'doe' and 1 = 1 -- AND pass = ANDpin=O

If the application is secured, each query would be unsuccessful, because of input validation. However if there's no input validation, the attacker will strive the prospect. First the attacker submits the primary question and receives a error message because of "1 =0 " that the attacker doesn't perceive the error is for input validation or for logical error in query. Then the attacker submits the second query that perpetually true. If there's no login error message, then the attacker finds the login field at risk of injection.

2. Timing Attacks

A timing attack lets an attacker pile up information from a database by observing timing attack delays within the database's responses. This system by victimization if-then statement cause the SQL engine to execute an extended running query or a time delay statement reckoning on the logic injected. This attack is analogous to blind injection and attacker will then live the time the page takes to load to work out if the injected statement is true. This system uses if-then statement for injecting queries. WAITFOR is a keyword on the branches, that causes the information to delay its response by a such as time. For instance, within the following query:

declare @ varchar(8000) select @ = db_nameO if (ascii(substring(@, 1, 1)) & (power(2, 0))) > 0 waitfor delay '0:0:5'

Database can pause for 5 seconds if the primary little bit of the primary computer memory unit of the name of this information is one. Then code is then injected to get a delay in latent period once the condition is true. Also, attacker will raise a series of alternative questions about this character. As these examples show, the data is extracted from the database employing a vulnerable parameter [4].

G. Alternate Encodings: The attacker uses Alternate Encodings like, ASCII, Unicode, EBCDIC and positional representation system to inject code in order that it will bypass the validations on the input, if any.

Example –

Original Query- „Select * from login where username = „a123‟ and pwd=‟xxx‟‟

Injected Query- „Select * from login where username = „ ‟; exec(char(0x73687574646f776e)) --‟ and pwd=‟not required‟”

The value passed to the char() perform is that the hexadecimal coding for SHUTDOWN. Therefore because the on top of injection uses hexadecimal coding rather than actual characters, it'll bypass the input validations and can cause the SHUTDOWN command to be execute [10].

III. VARIOUS TECHNIQUES TO DETECT AND PREVENT AGAINST SQL INJECTION ATTACK:

There are various techniques exist to detect and prevent the SQL Injection attacks. Following are the various approaches used to detect the SQL injection attacks.

- SQLrand was projected that uses instruction set randomization of SQL statements to visualize SQL injection attacks. It uses a proxy to append the key to SQL keywords. A derandomizing proxy then converts the randomized queries into acceptable SQL queries for the information. The key's not illustrious to the attacker, therefore the code injected by attacker is treated as indefinable keywords and expressions that cause runtime exceptions, and also the question isn't sent to the database. The disadvantage of this method is its advanced configuration and also the security of the key. If the key is exposed, the attacker will formulate queries for a successful attack.[12]
- Halfond and Orso in developed AMNESIA that may be a model-based technique that mixes static and dynamic analysis. The tool first identifies hotspots wherever SQL queries are issued to database engines. At every hotspot, a query model is developed by exploitation Non-Deterministic Finite Automata (NDFA). The hotspot is instrumented with monitor code, that matches the dynamically generated question against the query model. [7]
- Angelo Ciampa in projected associate degree approach and a tool- named V1p3R for web-application penetration testing. The operating of this approach is predicated on pattern matching of error message associate degreed on outputs made by the applying beneath testing; it depends upon an protrusible knowledge-base consisting of an oversized set of templates.
- Raveshi offers a method, dynamic query structure validation, that mechanically (and dynamically) mines programmer-intended query structures at every SQL query location, therefore providing a strong answer to the retrofitting downside. [11]

- Sonam Panda Approach - In [1], propose a method wherever predefined strategies are used and hybrid cryptography method is applied within the information to remain far away from attack on login part. This applied hybrid cryptography methodology may be a integration of Advanced cryptography customary (AES) and Rabin cryptosystem.

IV. TECHNIQUE TO SECURE AGAINST WEB APPLICATIONS

In this paper we present the replacement technique for preventing Database against SQL injection attack exploitation using integrated approach. Throughout this approach, in an exceedingly Login Table, two columns are created by DBA. One for username and different is used for password. This technique needs two additional columns. One for the secure hash value of username and one for the secure hash value of password. The secure hash values of username and word are designed and keep in Login Table once the user's account is initial time created with the web application. Whenever user wishes to login to account his/her identity is verified via username, word and secure hash values. These secure hash values generated at runtime exploitation stored procedure once user desires to login into the database.

In database there are conjointly separate tables for AES secret encryption and combined approach of secure hashing on encryption. Different techniques and strategies square measure being developed that are wont to defend the database. By applying encryption technique, database attacks will be prevented. Secret writing of data essentially helps to vary the information into a type that's not readable [1]. While not the proper key, this format can't be deciphered even if attacker hacks the data. Application of secret writing in login section makes it troublesome for unauthorized users to access the data. In this paper, some predefined strategies square measure mentioned and mixture of encryption technique with secure hashing is applied within the database to avoid attack on login section. This applied integrated approach could be an encryption technique that is a combination of advanced encryption Standard (AES) and Secure Hashing technique [5].

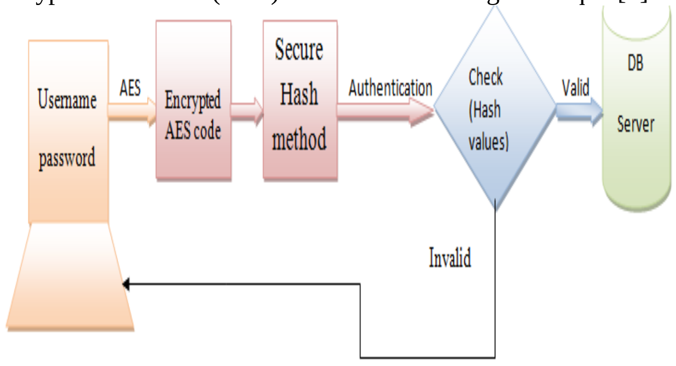

Fig. 2 AES and secure hash technique

V. CONCEQUENCES OF SQL INJECTION ATTACK

With SQL injections, criminals will take complete control of the database allowing them to govern the database to do something they need. They could insert a command to induce access to all or any account details in a system, including user names, and retrieve passwords from the written record. Databases may well be shut down. Files may well be uploaded, corrupting, or deleted. Fraudsters may interact in online theft by dynamic the worth of a product or service, so the price is negligible or free. Bastard names may be inserted, or master card details may be stolen. Taken master card data could even be inserted into a system to scam it at a later date. A database and every one of its contents may even be deleted. To help shield against these attacks, organization should check and filter user input and limit the length of input strings, as most attacks depend upon query strings. [6]

CONCLUSION

The SQL - Injection Attacks area unit hugely dangerous in association to alternative varieties of Web-based attacks, for the reason that here the top result's knowledge manipulation.SQL injection holes may be simply exploited by a method known as SQL Injection Attacks. The web-application code dead contains a policy that permits identifying lawful and malicious queries. During this paper, completely different types of SQL injection attacks furthermore as predefined bar strategies are mentioned. Then the hybrid approach of cryptography is used which has AES encryption and Secure hashing technique. The explanation behind applying 2 layer of cryptography & hashing is that it'll provide additional security in information. SQL question is generated and encrypted by exploitation AES technique that is additional hashed by a secure hashing technique and as we all know hashed codes could be a method cryptography technique therefore it's not possible to rewrite it.

REFERENCES

[1] Sonam Panda, 1 Ramani S2, "Protection of Web Application against Sql Injection Attacks", International Journal of Modern Engineering Research (IJMER) Vol.3, Issue.1, Jan-Feb. 2013 pp- 166-168 ISSN: 2249- 6645

[2] W. G. Halfond, J. Viegas, and A. Orso , "A Classification of SQL Injection Attacks and Countermeasures," in Proc. The International Symposium on Secure Software Engineering 2006.

[3] The Open Web Application Security Project, OWAP TOP 10 Projects. [Online]. Available: http://www.owasp.org/

[4] Mayank Namdev *, Fehreen Hasan, Gaurav Shrivastav, " Review of SQL Injection Attack and Proposed Method for Detection and Prevention of SQLIA" , International Journal of Advanced Research in Computer Science and Software Engineering,Volume 2, Issue 7, July 2012

[5] Neha Mishra1, Sunita Gond2, "Defenses To Protect Against SQL Injection Attacks", International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 10, October 2013

[6] The blackhat's toolbox: SQL injections Steve Moyle, founder and chief technology officer, Secerno

[7] William G.J. Halfond and Alessandro Orso,Preventing SQL Injection Attacks Using AMNESIA,ICSE'06, May 20–28, 2006

[8] 1Khaleel Ahmad*, 2Jayant Shekhar and 3K.P. Yadav, “A Potential Solution to Mitigate SQL Injection Attack”, VSRD-TNTJ, Vol. I (3), 2010, 145-152
[9] 1Dr. T. N. Sharma, 2 Amitesh Kumar, 3 Sumitra Singar, 4Vishakha Singhal, “A Propose Model for Prevention of Attack SQL Injection”,IJCST Vol. 4, Iss ue Spl - 2, April - June 2013
[10] Neha Singh,Ravindra Kumar Purwar,”SQL INJECTIONS – A HAZARD TO WEB APPLICATIONS”, International Journal Of Advanced Research in Computer Science and Software Engineering,Volume 2, Issue 6, June 2012
[11] Ms. Zeinab Raveshi,Mrs. Sonali R. Idate, “Efficient Method to Secure Web applications and Databases against SQL Injection Attacks”, International Journal of Advanced Research in Computer Science and Engineering Research Paper,Volume 3, Issue 5, May 2013.
[12] S. W. Boyd and A. D. Keromytis. SQLrand: Preventing SQL Injection Attacks. In Proceedings of the 2nd Applied Cryptography and Network Security Conference, pages 292–302, June 2004.