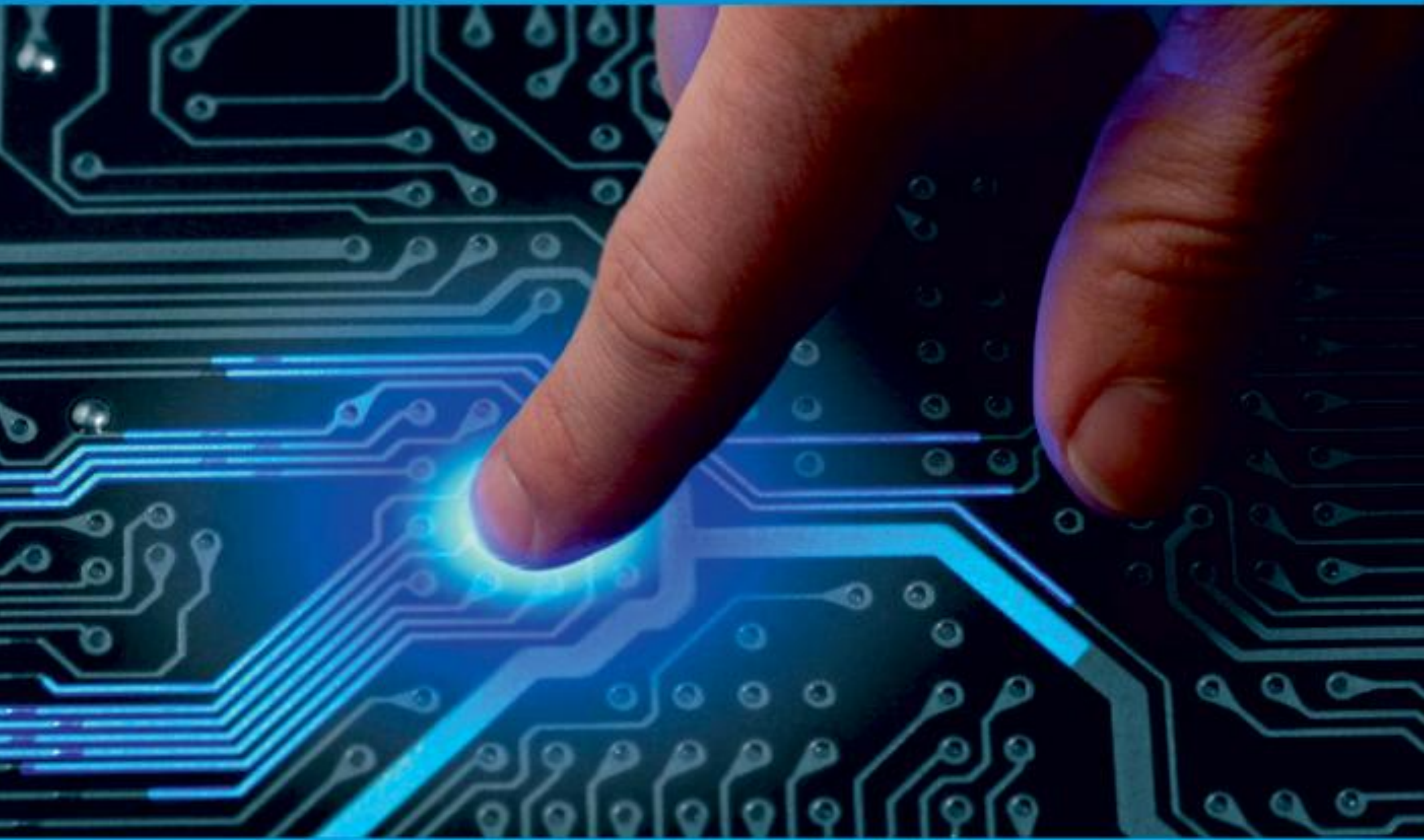
e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 10, Issue 5, May 2022

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 5, May 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1005115|

A Review on SQL Injection and Preventing Injection Attacks

PRAVEENA N [1], ASHWINI C[2]

PG Student, Department of MCA, University BDT College of Engineering, Davangere, Karnataka, India

Assistant Professor, Department of MCA, University BDT College of Engineering, Davangere, Karnataka, India

ABSTRACT: As the world is shifting in the direction of digitalization, SQL injection poses a protection hassle in which most of the systems manipulate the database and their requirements, which incorporate database servers to enable their clients to fetch data in the Structured Query language (SQL). The databases grow to be the focus of the interest to the hackers as they take advantage of vulnerabilities because of lack of input validation, allowing more than one query and SQL parameters to be used to introduce malicious code into the application that can immediately have an impact on an individual or enterprise financially and in terms ofreputation. This paper discusses the history of SQL injection attacks, some forms of injection attacks, and prevention measures to protect by the numerous application attacks and then the evaluation of different injection attacks and prevention techniques compared to the attacks and prevention methods.

KEYWORDS: SQL injection Attacks, SQL prevention.

I.INTRODUCTION

A SQL injection is a type of injection attack in an application where the attacker provides SQL code to a user input box of a web form to gain unauthorised access to a database that stores information about the users, admins, or company. An injection attack can harm the database in various ways, such as unauthorised manipulation of the database or retrieval of sensitive data. It can also be used to run system-level instructions that force the system to reject the application service. This issue is risky because it can cause data loss or misuse of data by parties who are not authorized, and as a result, the functionality and confidentiality of the application are destroyed.
To avoid this, web developers need to perform proper input sanitization and syntax evaluation to follow the security guidelines for the prevention of major loopholes during the programming phase. Many of the open source tools are being developed, however none of them will offer the excellent effects to save from all of the vulnerabilities. It is incredibly difficult for a security-minded developer to create an application that is free of all vulnerabilities, as some scanners fail to detect stored vulnerabilities, particularly when detecting

- cross-site scripting (XSS)
- cross-channel commands (XCS)
- Informationleaks

But none of them are very effective. OWASP (Open Web Application Security Project), a non-profit foundation focused on web application security. OWASP lists the top vulnerabilities where injection attacks have been gradedas "OWASP Top 10" the most severe vulnerability that a person, company or organization must prevent in their programs with each of their updates.

II.LITERATURE SURVEY

There are plentiful research papers which present various injection attacks and their prevention methods. The researchers started to find the solution to attacks by finding the application vulnerabilities in the development of various models. The models however cannot guarantee protection against other types of SQL injection attacks with completelydiverse attack pattern payload features, such as stored procedures and piggy-backed query attacks.

2.1. Background

Jeff Forristal, a well-known security specialist, originally documented injection in 1998 in an essay published in the magazine rain.forest.puppy, described how the Microsoft SQL server will use simple statements to retrieve sensitive data. However, SQL injection considered asanextremedifficulty in 2002 due to the terrorist assaults of September 11, 2001, which caused the concept of enhancing cyber safetyin the United States by the means ofdecreasing threads, viruses, and worms.A critical examination of vulnerabilities that could weaken the government or infrastructure reveals that SQL injection attacks can be highlighted as a particularly dangerous vulnerability for websites related to database software.

2.2. Attacks happened

Numerous instructional papers have dealt with SQL injection assaults.

- ➢ In 2007, Heartland Payment Systems was hacked through SQL injection attack which causes $130 million loss. The robbery of one hundred thirty million credit card numbers is one of all the largest record breaches of credit card fraud records in history.
- ➢ The 2008 Myspace records breach ranks as one of all the biggest assaults on a customer website. Cybercriminals stole emails, names, and partial passwords of almost 360 million human beings owed money.
- ➢ UK telecom TalkTalk cameunderhearthregion in 2015 which isvulnerabledanger that compromised customer’s information.
- ➢ An SQL injection example that issues the regular gamer, Epic Games had their forums hacked in 2016 and 8, 00, 000 humans owed cashwere leaked. SQL injections targetedat thewell-knownonline message board software database vBulletin, which has becomeinfamous for its vulnerability to SQL exploits. In general, SQL injection assaults are spreading through the gaming enterprise like wildfire.
- ➢ The 2017 Equifax records breach yielded private information such as names, social protection numbers, start dates, and addresses for 143 million consumers. Before the records breach occurred, a cyber-security studies organisation even warned Equifax they had been vulnerable to a SQLI attack, but the credit bureau took no action till it became too late.
- ➢ In 2018 Cisco prime license manager vulnerability was being detected which can gain access to system license plate manager.
- ➢ In 2019 aSQLinjection vulnerability was discovered in fortnite, which is an online game with over 350 million users, both are being patched the vulnerability.Countermeasures are also implemented by well-trained programmers with security in mind by adopting different models.

III.SQL INJECTION ATTACKS

Many applications that implement a web based login use a database to keep consumer credentials and carry out anSQL query to validate every login attempt.

Types of SQL injection attacks with an illustration

For the demonstration of some kinds of injection attacks,consider a banking site authenticates customer's username and password to login. If the login succeeds, details can be displayed. If not, the message is returned.

A. Tautologies

TautologySQL injection attack done through the use of "WHERE" clause usuallytrue for every query, which ends up in bypassing the authentication and retrieve details.

Retrieving clients when "uname" is known

Name: kaya' or '1'='1--

SQL: SELECT * FROM customers WHERE uname = 'kavya' or '1'='1' --

Result: All customers names are retrieved.

If both the username and password of clientare not known

username: ' or '' = '

password: ' or '' = '

SQL: Select * from students where username = '' or '' = '' and password = '' or '' = ''

Outcome: All clients are retrieved with their passwords.

Tautology-based SQL injection strategies are utilized by maximum hackers to skip the verification segment through simply adding "--" inline comment, which makes the relaxation of the SQL command as comment to perform the most result in return with the low variety of conditions.

B. Union Queries

This is a code injection and SQL manipulation attack where the UNION operator is used to merge two independent queries together like sending a SELECT query and UNIONS from the real SQL statement. The attacker needs to have anin-depth knowledge of the database schema and its tuples, somaximuminformation to be retrieved

Syntax:

NormalSQL statement + "semi-colon" + UNION SELECT <rest of injected query>.

When appearing an SQL injection UNION assault, one method which consists of injecting a chain for ORDER BY clauses and incrementing the preferred column index until a mistake occurs.For example, assuming the injection issue is a quoted string in WHERE clause of the actual query to submit:

‘ORDER BY 1 --
‘ORDER BY 2 --
‘ORDER BY 3 --

To extract a table from the current database

1' and 1=2 union select 1, group_concat (table_name), 3, 4 from information_schema.tables where table_schema = database () -- -

To extract column name from tablename which is selected

1' and 1=2 union select 1, group_concat (column_name), 3, 4 from information_schema.columns where table_schema = database () and table_name ='user'-- -

Finally extract sensitive information from table user

1' and 1=2 union select 1,group_concat(username,0x3a,password),3,4 from user-- -

Here, the injected query will show usernames and passwords from the users table on the webpage as a single set of results.

C. Logically incorrect queries

Logically incorrect Queries collect all potential information about the structure and the schema of the tables inside the database. This is a SQL manipulation attack where the error message produced by the database provides the attacker with an advantage of generating some error message from the databases that will provide the attacker the necessary information. This is considered as a preliminary, information-gathering step in SQL injection attacks. For example:

Examining the column name

username: 'ddpd"

password: aaaf

SQL: SELECT * FROM customers WHERE username = 'ddpd"' AND password =cndk’

Outcome:"Incorrect syntax near 'ddpd'. Unclosed quotation mark after the character string " AND Password='aaaf"."

Examining the number of columns in a table

username:customerid=A123' union select username,password from customers --

SQL: SELECT * FROM customers WHERE customerid = ' A123' ' union select username,username from customers--

Outcome: "All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists."

Examining the number of columns in a table

username: customerid =A123' union select username, password, studentid fromcustomers--

SQL: SELECT * FROM customers WHERE customerid = 'A123' union select username, username, username from customers --

Outcome: "Login failed: Invalid credentials"

Examining table and/or column name

username: A123 ' group by (customerid)--

SQL: SELECT * FROM customers WHERE customerid = ' A123' group by (customerid) --
Outcome: "Column ' customers.username is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause."
Examining data type
Input (username): customerid = A123' compute sum (username)--
SQL: SELECT * FROM customers WHERE username = ' A123' customerid = A123' compute sum (username) --
Outcome: "Operand data type varchar is invalid for sum operator."
Examining data type
Input (username): customerid = A123' compute sum (customerid)--
SQL: SELECT * FROM customers WHERE username = ' A123' compute sum (customerid) --
Outcome:"Login failed: Invalid credentials"

D. Piggy-Backed Query

Piggy-backed queries represent the code injection attack, where the independent query was successfully inserted using the keywords. Here, after the first query is finished, the second query is executed like a normal query. The second query is hidden and masked as the first extended query and sent to the database.
Syntax:
NormalSQL statement + ";" + INSERT (or UPDATE, DELETE, DROP) <rest of injectable query>
For illustration:
To add a new customersaccount
'or 1=1; SET IDENTITY_INSERT customers ON INSERT INTO customers (customerID, Username, Password) VALUES (attacker customerid, 'attacker username','attacker password');--
Modify existing customergrades
'or 1=1; UPDATE Grades SET grade='attacker grade' WHERE customerid=attacker customerid;--
Delete existing customergrades
'or 1=1; DELETE FROM Grades WHERE course='attacker class' AND customerId=attacker customerid;--
This attack is convenientlyachievable if the database has given permission to execute a multiple queries at the same row, howeverthat isnot common ininserting information manipulation operations together with INSERT, UPDATE, and DELETE can compromise the integrity of the database.

E. Alternate Encodings

In this technique, attackers modify the injection query by using alternate encoding where Base64, ASCII, HEX, or Unicode are used as encoding methodologies to trick the intrusion detection systems/intrusion prevention systemby changing the communication of the injected SQL query. They can escape the developer's filter that scans input requests for special "bad characters".
For example:
Input: Select empno from employee where name=” and pin=0; exec (char (ox73687574646f776e))
The hexadecimal encoded character are taken as input which is used as char function that returns actual character. This encrypted string, shutdowns the database when the command is performed. The usual appearance of the query can be altered by using comment lines, which prevent IDSs and IPSs from detecting the anomaly. For example, between these /* */ nothing is considered executable by the MySQL server; but it changes the appearance of the attack. Therefore, IDS and IPS cannot detect it. Another practical example is ALT/*/ER will act like an ALTER function. The flexible nature of SQL provides the attacker with ample choices to change the form of his attack and evade the detection algorithms.

4. PREVENTING INJECTION ATTACKS

The prevention of SQL Injection vulnerabilities in anapplicationsmay be accomplished by developers with the aid of using parameterized database queries with bound, typed parameters and cautious use of parameterized storedproceduresin the database. This may bedone in a number of programming languages and extraequipmentwhich are able to detection the SQL injection attacks evolvedwith the aid of using many scholars.

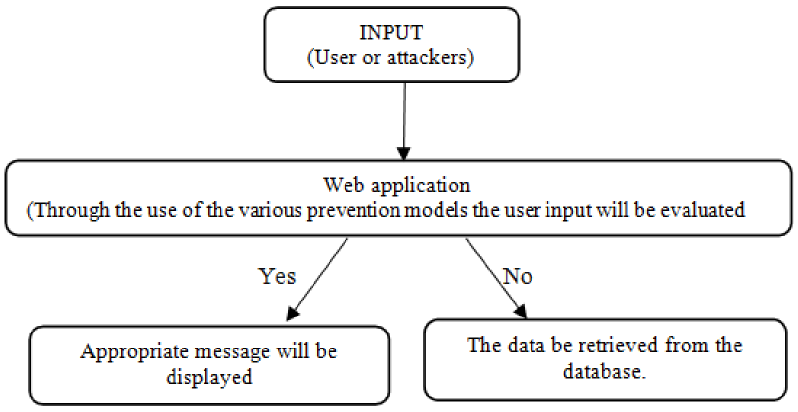

INPUT
(User or attackers)
Web application
(Through the use of the various prevention models the user input will be evaluated
Yes
No
Appropriate message will be displayed
The data be retrieved from the database.

Figure 1: How the attacks are processed generally by different prevention models.

I. Tautology checker

Tautology checker is a method proposed by G. Wassermann and Z. Su that detects vulnerabilities withoutgenuinely executing the code howeverwith the aid of using analysing the written code to save from tautology attackshowever is not usefultowardsdifferentattacks.

II. Whitelisting/Blacklisting

The term "whitelist" refers to permitting only those words or characters in an input fieldthat are predefined by the developer. For example, usernames generally do not have special characters, so restricting special characters in the username field won't be accepted. Any other entry or input will be rejected. The opposite of that is blacklisting, which refers to now no longerpermittingthe ones characters or expressionswhich might bedescribed in a blacklist by the developer.

Any entry that is stated in the blacklist will be automatically removed or will generate an error. SQL syntax consisting of UNION, AND, OR and others may be positioned into a blacklist so that no longer be utilized by an attacker to get entry to data. Whitelisting is more effective than blacklisting as an attacker uses complicated encoding schemes that may not be present in a blacklist.

III. SQLrand approach

Stephen W. Boyd and Angelo D. Keromytis proposed this methodin which the proxy server among the webserver and the database is used for de-ciphering the queries that aresent from client to database server. This method has two essential tasks. One is to de-randomize the SQL queries and thensendSQL queries with the same old set to the database server for computation. Another task is to handle the error message, which is generated by the database server On behalf of a query that allows hackers get recordsapproximatelyabout database tables and schemas.

IV. SQLchecktechnique

The SQL Check proposed by Zhendong Su and Gary Wassermann to test the real-time enter of queries have beencarried out with described ones through the developer and a secret key wasimplemented for the consumerenter delamination.

V. Parameterized queries

This technique, designed to make SQL more efficient, consists of a predefined structure of SQL syntax that includes placeholders. The parameters that can be provided to the application by the user are not immediately sent to the database but first placed in the predefined parameter and if the query generates no errors, then the

SQL query is sent to the database for execution. This prevents an injected SQL query from being executed directly in the database.
As an example,
SELECT * FROM pbook WHERE uname =? AND pass =? ;
Where "?"is placeholder. This querycan befinishedvia way of means of server after whichdispatched to database.

VI. WASP

Halfond and his colleaguesintroduced WASP to work on the basis of dynamic tainting and also syntax evaluation, which is syntax sensitive. The types of tainting are as follows:

- Positive tainting differs from negative tainting due to the factit's primarily constructed on thefinding, marking, and tracing of trusted data rather than untrusted data. By addressing issuesdue to incompleteness, it trulyessentialtaskthroughout the implementation of protectiontechniquesbasedmostly on dynamic tainting with in theidentification of relevantrecords to be marked. The purpose for that isit's farcommonguidance for builders to constructSQLinstructionsthrough the approachof mixing hard-coded strings that consist ofSQLkeywords or operators with user-provided numeric or string literals. This approach uses whitelisting and the general principle of fail-safe defaults as outlined by Satzer and Schroeder.
- Accurate Taint Propagation includesobserving taint resultsrelated to the recordsas it is used and manipulated at runtime. When tainting is used for security-related applications, it is especially important for the propagation to be accurate. Inaccurate propagation can compromise the effectiveness of the technique by associating incorrect markers with the data, which can lead to mishandling of the approach.
- Character-level tainting tracks taint information at the character level rather than at the string level because to build SQL queries, strings are constantly broken into substrings, manipulated, combined, and achieve the encapsulation offered by object-oriented languages, andparticularlythroughway of Java, in which all string manipulations are finishedusing a small set of methods. Another approach is to trace tainted data at the bit level to manipulate as character values using bitwise operators, which are complex to implement.
- Syntactic-Aware Evaluation permits using untrusted input data in an SQL query as long as the query does not cause an SQL injection attacks. The approach first makes use of an SQL parser to disturb the string. It then iterates through the tokens and tests whether or not no longer tokens (substrings) other than literals incorporate most effective depended on records.

V. A COMPARATIVE ANALYSIS OF SQL INJECTION ATTACKS AND PREVENTION METHODS

Many researchers have explored severalapproaches to discover and prevent various SQLinjection and prevention methods where most chosen techniques are static, dynamic or combination of both. In static analysis, the vulnerabilities are discovered through code inspection, whereas in dynamic analysis can be performed automatically by the analysis of vulnerabilities during the execution of a web application, which avoids thousands of tests by doing several tests manually.

Techniques	Tautologies	Logically Improper Queries	Union Queries	Piggy-Backed Query	Alternate Encoding	Detection	Prevention
Whitelisting/Blacklisting	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SQL rand approach	Yes	No	Yes	Yes	No	Yes	Yes
SQL check approach	Yes	Yes	Yes	Yes	Yes	Yes	No
WASP (Tainting)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Tautology checker	Yes	No	No	No	No	Yes	Yes

Table 1: A comparison of SQL injection attacks and prevention methods for different attack types.

In the above table, anassessment is being made to the evaluation of stoppingSQL injection assaultsthrough the prevention techniques.A SQL injection is one of the threats to the database system where a person has the motive to sneak into the web application to change, modify, and retrieve the sensitive data for the sake oftracking the enterprisewebsite hostingvulnerable application. There are diversevarieties ofassaults that arise, consisting of tautology, that'susing logical operations to outline a statementthat'scontinually true. Then the logically incorrect query will retrieve the information by the use of syntactical and logical incorrect queries so that an error will occur to know the various information in the database. The union attacks are a sort of logically incorrect injection attack. The alternate coding attack uses the decimal ASCII (American standard code for information interchange) and Unicode to escape from developers filters, which are very dangerous to escape from if they are combined with any other attacks.

VI. CONCLUSION

This paper presents,the various real-existence attacks of the attacker are being defined and then some of the SQL injection attacks are being described with their process. An attacker can use the loopholes in the application to get unauthorised access to the database by using correctly crafted queries attack, the attacker can alter or make any modifications to the database, and howeverthis will be detected and prevented through the various methods which are being used by many scholars or cyber security experts. The whole analysis of the injection and prevention methods is being described by using automated tools. Different test cases should be used to test applications regressively. If a vulnerability is found, then it could be fixedthrough the usage of prevention measures. Different counter measures paintings for extraordinaryvarieties ofSQL injection attacks.

REFERENCES

1. P. Kumar, R. K. Pateriya, "A Survey on SQL Injection Attacks, Detection and Prevention Techniques", IEEE International Conference on Computing Communications & Networking Technologies, July 2012, pp. 1-5. 2.
2. Zainab S. Alwan, Manal F. Younis. Detection and Prevention of SQL Injection Attack: A Survey. IJCSMC, Vol. 6, Issue. 8, August 2017.
3. Jai Puneet Singh. “Analysis of SQL Injection Detection Techniques”.
4. Pritam Sen, Shubham Mukherjee, Chittaranjan Pradhan, Sudeshna Bora. “SQL Injection: A Sample Review”, sixth ICCCNT 2015, Denton, U.S.A.
5. Puspendra Kumar, R.K. Pateriya. A Survey on SQL Injection Attacks, Detection and Prevention Techniques, ICCCNT’12, 2012, Coimbatore, India.
6. MD. Mazharul Islam Miraz, Aditya Rai, Swati, Harpreet Kaur, Deshbandhu Das.“SQL Injection: Classification and Prevention”, 2d ICIEM, 2021. 7.
7. W.G.J.Halfond, A. Orso, P.Manolios, WASP: Protecting Web Applications Using Positive Tainting and Syntax-Aware Evaluation, IEEE Transactions on software Engineering, 1, Feb 2008.
8. Vidushi, Prof. S. Niranjan. A Review on SQL Injection, IJERT, NCETEMS-2015 Conference.
9. https://brightsec.com/blo/SQL-injection-attack/
10. Thomas Hyslip. “SQL Injection: The Longest Running Sequel in Programming History”, June 2017.