UTM UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

A Survey on SQL Injection Prevention Methods

Shahbaaz Mohammed Hayat Chaki & Mazura Mat Din
Faculty of Engineering, School of Computing,
Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia
Email: shahbaazchaki123@gmail.com, mazura@utm.my

Submitted: 6/01/2019. Revised edition: 24/03/2019. Accepted: 28/03/2019. Published online: 30/05/2019
DOI: https://doi.org/10.11113/ijic.v9n1.224

***Abstract*—Database plays a very important role in everyone's life including the organizations since everything today is connected via Internet and to manage so many data. There is a need of database which helps organizations to organize, sort and manage the data and to ensure that the data which a user is receiving and sending through the mean of database is secure since the database stores almost everything such as Banking details which includes user id, Password and so. Thus, it means that the data are really valuable and confidential to us and therefore security really matters for database. SQL Injection Attacks on the database are becoming common in this era where the hackers are trying to steal the valuable data of an individual through the mean of SQL Injection Attack by using malicious query on the application. This application reveals the individual data by an efficient and the best SQL Injection Prevention technique is required in order to protect the individual data from being stolen by the hackers. Therefore, this paper will be focusing on reviewing different types of SQL Injection prevention methods and SQL injection types. The initial finding of this paper can make comparison of different types of SQL Injection Prevention methods which will enable the Database Administrator to choose the best and the efficient SQL Injection Prevention Method for their organization. Consequently, Preventing of SQL Injection Attack from happening which would ultimately result in no data loss of an user.**

***Keywords*—Comparison, SQL Injection, SQL Injection Prevention, SQL, Prevention Methods**

I. INTRODUCTION

A database is sorted or gathered information to keep vast amount of data. Everything is stored in a Computer server or a system. Hence, databases are more complex and are frequently created utilizing formal structure/design and displaying strategies and modelling techniques called DBM, the database management system works in synergy with the database, applications and end clients to analyses data and provide the administrator with a direct access to the required database information when required. Essentially, the database entries with the DBMS and relevant applications form the database system. Likewise, it is not uncommon to use the word "database" to refer to the database system, DBMS or any database related application. The expression "database" is additionally used to freely allude to any of the DBMS, the database system or an application related with the database. Database management systems may also be arranged by considering how they bolster different database models. The 1980s was dominated by social databases, during which information was modelled as sections and lines in a table progression. Information queries and composition were, for the most part, also done using SQL. However, in the 2000s, there was the emergence of non-social databases that was dubbed NoSQL drawing on the fact that a rather diverse question dialects were adopted.

A database is like a heart of any organization. This is because without a database, the organization will not work or would be incomplete. Database stores valuable information of a customer or client or sales lead or it connects multiple database which contains multiple tables which connects to a bigger platform, making an organization complete. A good example of database in organization would be Facebook, Amazon, the Airline industry or banking system and there are millions of such website that need database every mile second of their work in order to run their business in an efficient and organized manner rather than doing everything manual which takes a lot of efforts and time and nearly seems impossible in this era where we fully depend on technology and a great example of this era technology would be database which helps us in day to day life. In a formal sense, the word "database" is used to refer the organization of a group of information that are connected in some way. Furthermore, the DBMS is actively used to access the information resource stored in such database. This is possible as the DBMS consists of connected computer systems that make it possible for users who need any information contained in the database to have access to them. However, in some cases there might also be restrictions that are designed to make curtail the ability to access some data.

Hence, in operation, DBMS ensures that users have a wide range of functions that make massive information to be entered, stored, and retrieved while allowing for efficient management and organization of the information. Owing to the existent relationship between the two "database" has commonly been associated with information ensuring that DBMS enthusiasts often use them interchangeably. In other circles unrelated to information technology, "database" is often used when discussing any group of connected information like a card index or program. In observe it's quite common to own multiple databases. The database that deals along with your order and client information can be fully freelance from you information that deals with human resource information. And in several organizations, you don't simply have multiple databases however they have multiple DBMS. Typically it's as a result of one DBMS is healthier at one thing than the opposite.

DBMS can fall under different categories, including:

- Hierarchical Database Systems
- Relational Database Management Systems
- NoSQL Database Systems
- Object-Oriented Database Systems
- Network Database Systems

II. SQL Injection

Nowadays, a database is used in every organization since it acts as the heart of all web applications and is used in the storage of all the information that are required by such applications, for example, credit card details, customer personal information, client orders, company information, and others. For that reason, databases have become attractive to the hackers as they are much interested in getting access to the information stored. Also, due to the rapid internet growth in the world today and the emergence of various new technologies has led to a wide application of internet applications on the Web environment. Most of the companies today make use of Web platforms in processing their information. This has brought about a number of database security issues that organizations have to fight today.

One of the most commonly known Web security issues today is SQL injection. This issue has posed many problems to databases despite the fact that a number of solutions have been proposed. SQL injections occur when a developer directly accepts user inputs that are placed in SQL statements without validating or filtering out some dangerous characters. To execute SQL injection, the attacker combines the compilation of SQL statements with a weak input validation. Such an input validation leads to a change in how the SQL query is structured. Such can make it easier for the intruder alter the SQL statements that has been passed to the database as the parameters to enable them for getting access to all the data. Following these, attackers can also modify and even delete the entire database. Any database is vulnerable to SQL injections in case the user input is not thoroughly filtered for some malicious characters or if it is not strongly typed [17]. In most scenarios, hackers and intruders carry out nefarious activities on web servers and database backend by modifying the contents of the webpage or steal vital information by taking advantage of flaws in the SQL components.

SQL injection attacks are highly risky to an organization and the consequences are severe. A successful SQL injection attack bypasses authentication and authorization so as to get a total control of the database, steal critical information, access users' credential information, change user passwords, carry out illegal transactions and to cause destructions to the entire database. To improve the standards of the operation security of the web environment, a significant study of the detection and prevention targeting SQL injection attacks should be done. Database administrators have to understand how these attacks are detected so that they can thoroughly filter User inputs from within the database well before permitting them. You can also understand these other ways in which such SQL injection attacks can be minimized and prevented. For this reason, researchers have been hugely concerned with and studied extensively on detecting and defending SQL injections. The widespread availability and ability and of firewalls and similar information shielding defenses for information stored by enterprises has heightened remote attacks seen on applications of networked enterprises. Over the past few years, such protective measures are however simply penetrated and defeated by basic attacks of script injection, ensuring that the SQLCIA – SQL command injection attack – can become virulent. Web applications that have an information server backend based on SQL, take inputs from users and forms queries in a dynamic manner remains a SQLCIA associate degree target [16]. Where this susceptible application exists, associate degree SQLCIA exploits deformed input that modifies the application's SQL query for unauthorized information access, while being able to manipulate, modify or extract sensitive data.

The Common Vulnerabilities (CVE) list at http:/cve.mitre.org, suggest a high prevalence and hierarchical nature of SQL injection attacks as CVE assumed second position on internet application attacks in 2006. The proportion of such attacks in the general range of attacks increased from 5.5% in 2004 to 14 July 2006. It is noteworthy that the SQLCIA that affected Card System Solutions in 2006, breaching MasterCard numbers in their thousands, is a prime example of the damaging effects of associate degree attacks on not only a company but the public at large. Taking code search of Google into consideration, experts have discovered a lot of programs and applications with sources that are prone to such vulnerabilities. Additionally, Sutton (2006) and a number of findings have reported that many web applications are extremely prone to SQL injection attacks with this form of attack consistently rising over the years [10].

There are two primary categories of research conducted on SQL injection attacks; the approach could either be aimed at attacking obstacles or identifying the apps vulnerabilities. In identifying vulnerabilities, approaches adopted are geared towards determining susceptible locations where SQL injection attacks are likely to occur in an web based application. It is usually important for an engineer to ensure inputs are submitted and validated to prevent the susceptibility of an application to SQL injection attacks by identifying potential loopholes providing a handy technique that can be employed in the static analysis of identifying the vulnerabilities that may be inherent in a website. This involves analyzing code and carrying out input validation checks prior including it in a question. Although these static analyses are incredible in detecting vulnerabilities, using them to address disadvantages of SQL injections is still studied due to their characterization of unvalidated inputs. For instance, they are not efficient in ensuring input validation, hence many programs with the potential of victimizing input validation are able to go past the checks unidentified and therefore remain prone to injection. Static victimization analysis progress provides helping to check and validate the correctness of functions. More in-depth analysis of this subject is discussed in the categories and hindrance techniques of attacks that ensure programs are retrofitted to prevent injection attacks. The techniques don't require much manual annotation, offering more ways of prevention against SQL injection attacks, instead of just sleuthing program vulnerabilities.

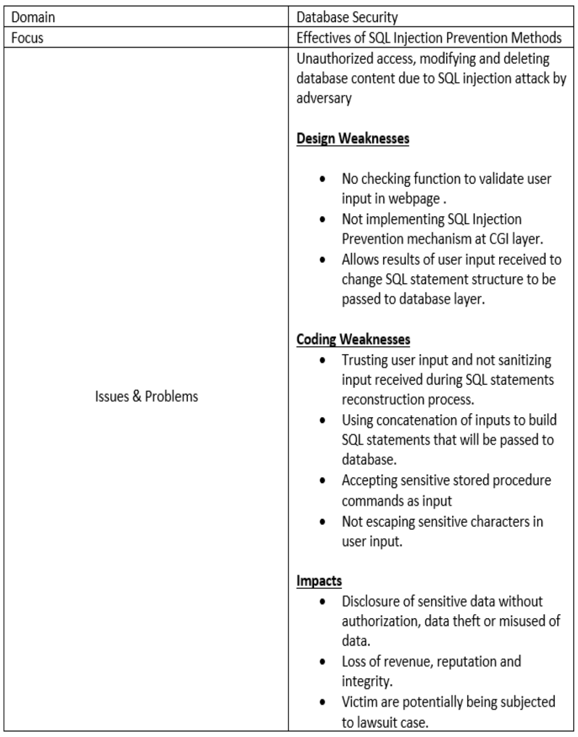
III. PROBLEM BACKGROUND

Many developers of the information system who suffered from SQL injection attacks did not take security elements into account during the initial design phases. Developers tend to assume that the system flow will always follows as designed and planned. This often allowed them to improperly handle that errors that at most of the time led to a successful SQL injection attack by the attacker. The attacker sends malicious SQL codes to web applications, Data sets and so on in the legitimate SQL statement that generates error messages depending on the statement that was used. The differences in results or error messages from the response would be used for SQL injection attacks.

There are many SQL Injection Prevention Method/Approaches that has been developed by many different authors in order to Prevention SQL Injection Attacks but unfortunately many of the database security expert spend their most of the time on deciding which approach is the best to prevent SQL Injection Attacks that results in Wasted of Money, Time and energy. Many of the database security experts deploy Intrusion Prevention System(IPS) or various types of firewall in order to prevent SQL Injection Attacks based on patterns that are known but this method is very much costly what most of the small/medium origination cannot afford.

Most of the SQL Injection Attacks happens because of bad security coding or sometimes company hire 3rd party developers to program and they tend to less care about the Database security that is in resulting to SQL Injection Attacks or the programmer tends to have no knowledge about SQL Injection Attacks Prevention Methods / Approaches. In Order to simplify things for programmers and Database Security experts related to SQL Injection Attacks Prevention Methods / Approaches, I will be doing the Paper on "A Survey on SQL Injection Prevention Methods" which would help many database security experts, Programmers etc. To identity on which SQL Injection Prevention method is more reliable and also suitable to safe guard the organization database thus saving the organization money, time and energy. Also this experiment will help on knowing the each SQL Injection Prevention Method on their effectiveness.

TABLE 1. Paper Details



Domain	Database Security
Focus	Effectives of SQL Injection Prevention Methods
Issues & Problems	Unauthorized access, modifying and deleting database content due to SQL injection attack by adversary **Design Weaknesses** • No checking function to validate user input in webpage . • Not implementing SQL Injection Prevention mechanism at CGI layer. • Allows results of user input received to change SQL statement structure to be passed to database layer. **Coding Weaknesses** • Trusting user input and not sanitizing input received during SQL statements reconstruction process. • Using concatenation of inputs to build SQL statements that will be passed to database. • Accepting sensitive stored procedure commands as input • Not escaping sensitive characters in user input. **Impacts** • Disclosure of sensitive data without authorization, data theft or misused of data. • Loss of revenue, reputation and integrity. • Victim are potentially being subjected to lawsuit case.

IV. SCOPE OF THE PAPER

This Paper will have the following scope:

- This Paper will be only focusing on SQL Injection Prevention Effectiveness under a controlled experimental approved environment.
- The experiments which would be conducted will have various SQL Injection Attacks on the data set.
- Attacks will be conducted on various real life scenario data set based on the approval from the data set owner/application owner/website owner for experiments purpose only.

V. AIM OF THE PAPER

This research will help Database Administrator on deciding which is the best MySQL prevention method that can be chosen in order to protect Database from SQL Injection Attacks. This will be achieved by doing comparison study based on different types of SQL Injection Prevention Method.

Furthermore, this study will also help researcher of the SQL Injection Prevention Methods on knowing the flaws that they have in their SQL Prevention method thus giving them valuable feedback to improve their SQL Injection Prevention Methods for future usage.

VI. METHODOLOGY

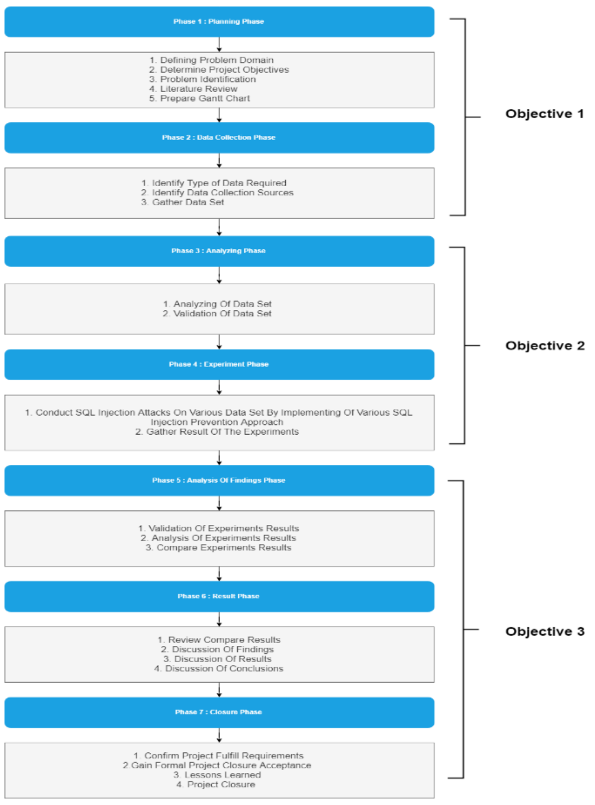

Phase 1 : Planning Phase
1. Defining Problem Domain
2. Determine Project Objectives
3. Problem Identification
4. Literature Review
5. Prepare Gantt Chart
Phase 2 : Data Collection Phase
1. Identify Type of Data Required
2. Identify Data Collection Sources
3. Gather Data Set
Objective 1
Phase 3 : Analyzing Phase
1. Analyzing Of Data Set
2. Validation Of Data Set
Phase 4 : Experiment Phase
1. Conduct SQL Injection Attacks On Various Data Set By Implementing Of Various SQL Injection Prevention Approach
2. Gather Result Of The Experiments
Objective 2
Phase 5 : Analysis Of Findings Phase
1. Validation Of Experiments Results
2. Analysis Of Experiments Results
3. Compare Experiments Results
Phase 6 : Result Phase
1. Review Compare Results
2. Discussion Of Findings
3. Discussion Of Results
4. Discussion Of Conclusions
Phase 7 : Closure Phase
1. Confirm Project Fulfill Requirements
2 Gain Formal Project Closure Acceptance
3. Lessons Learned
4. Project Closure
Objective 3

Fig. 1. Methodology

A. Planning Phase

In this phase, we discuss about the problem domain, Objectives and time constraints.

B. Data Collect Phase

In this phase of the project, we will be working on Identifying the Type Of Data set that the project would be required in order to conduct the experiments. Once the Data set type is being identify then I would be proceeding to identifying the data set source.

C. Analysis Phase

In this phase, we will validate the data that was collected to ensure that the data is a legit and can be used for our comparison.

D. Experiment Phase

In this phase, we will conduct various types of SQL injection attack using different types of SQL injection prevention methods to determine which is the best SQL injection prevention method. The SQL attack method and prevention method are written down further in this paper.

E. Analysis of Finding Phase

In this phase, we will analyses the result of the SQL injection attack which were conducted earlier using IBM SPSS software.

F. Result Phase

In this phase, we will conclude the results which were outputted by the IBM SPSS software.

G. Project Closure Phase

This phase will include full project report and closure approval from supervisor, Examiner and school of computing. The closure approval determines that the project report is as per the standards of the UTM which is required.

VII. COMMON TYPES OF SQL ATTACKS

Tautologies: This type of attack operates by injecting the code with one or more queries that are conditional SQL statements in a bid to render the SQL command execute like a true condition like (1=1) or (- -). Tautologies are often employed in bypassing web page authentications permitting access to the server of the database.

Piggy-backed Query: In this type of query, the attack on the database employs delimiters in queries such as ";" to inject more query statements than was present in the authentic query. The original query is the first, while subsequent ones are injected queries. This SQL attack is dangerous and can be used by an attacker to inject almost any SQL command.

Logically Incorrect: This attack uses the error messages that are returned following an incorrect request in the database. This is handy for attackers as error messages provide valuable information that can be used to determine how vulnerable application parameters or schema are:

- Union query: Also called injection statement attack, this breach occurs when a hacker adds more statement into the initial SQL statement. A union query attack can be carried out by including a UNION query and is also possible by adding a form "; < SQL statement > " statement into a vulnerable parameter.
- Stored Procedure: The attacker focuses on the stored processes in the database system in this technique. Procedures stored run directly through the database engine. It's an exploitable piece of code. The saved procedure gives the authorized or unauthorized customers true or false values. The attacker will write "; SHUTDOWN;-" for SQLIA with login.
- Alternate Encodings: This type of attack happens Once the injection query is changed by an attacker using alternative encoding, such as hexadecimal, ASCII and Unicode. This enables the attacker to escape the filter of the developer, who scans input queries for special known "bad character". If this

kind of attack combines with different attack techniques, it maybe be strong as it can target different layers in the application so that developers need to be acquainted with all of them to provide an efficient defensive coding to prevent alternative encoding.

VIII. Risk associated with SQL injection

SQL injection is harmful and the associated risks give attackers an incentive to attack the database. The main consequences of these vulnerabilities are attacks on the following features [5]:

A) Authorization: The Critical and the valuable data that are stored in a vulnerable SQL database can be modified by a successful SQL Injection Attack.
B) Authentication: If the input fields within the authentication page are not properly controlled, it may be possible to log in as a normal user to a system without knowing the authenticated user.
C) Confidential: Databases usually consist of sensitive information such as personal data, credit card numbers and/or social numbers. The loss of confidentiality is therefore an awful problem with the vulnerability of SQL injection.
D) Integrity: Not only does an attacker read sensitive information through a successful SQLIA, but it is also possible to change or delete this private information.
E) Database Fingerprinting: The attacker can determine on the type of database which is being used in the backend in order to use the database-specific attacks that match weaknesses in a particular database management system (DBMS).

IX. Prevention method for SQL injection attacks

1. AMNESIA In, the author developed a technique called AMNESIA, which is used to analyses and monitor the neutralization of SQLIA. AMNESIA combines dynamic and static analysis to detect and prevent vulnerabilities of web applications during runtime. To generate different types of query statements, AMNESIA uses static analysis. AMNESIA interprets all queries in the dynamic phase before they are sent to the database and validates each query against the statically built models [4].

2. The authors [6] developed a WASP (Web Application SQL Injection Protector) is a kind of tool that can help in prevention and stopping more than 12,000 Attacks without having false alarm. The limitation of this tool depends on the approach which would use in the application that needs to be deployed.

3. In this the author [21] has developed a R-WASP (Real Time-Web Application SQL Injection Detector and Preventer) tool capable of effectively stopping all attacks and detecting SQLIAs in real time. The restriction of this tool requires more practices to reduce attacks on stored procedures effectively. In addition to detecting all classical SQLIA types, the authors developed an appropriate Real Time Web Application SQL Injection Protector (RT-WASP) tool to detect SQL injection attacks in stored procedures. The disadvantages of the RT-WASP tool, which does not detect the SQL Injection and XSS attack, are therefore intended by the authors to extend the RTWASP technique to include SQLI and XSS attacks in web applications.

4. In this the authors [3] used a prevention tool called SecuriFly for java. This tool is used to chase string instead of taint information character and try to remove query strings that were generated using tainted input. This approach cannot be used to stop an injection result from inserting SQL commands into numerical fields. The main limitation of this approach is that it is difficult to identify all the user input sources. In addition, this technique partially stops all classical SQLIA types due to the limitations of the underlying approach.

5. In this the author [8] presents a method for the automatic prevention of SQLIA called CANDID by dynamic candidate evaluations. This technique extracts the query structures dynamically from every SQL query location that the developer intends to use. CANDID therefore solves the problem of modifying the application manually in order to create the prepared statements. The disadvantage of this technique is that SQLIAs are partially stopped due to the limitations of the basic approach.

6. The author proposed a method which is to implemented PHP Data Object (PDO). PDO extension is a lightweight, consistent interface for accessing databases in PHP, and has become one of the among trends in developing dynamic web applications that connect to the database. The purpose behind using PDO in would be easy installation, Security, flexibility and faster execution when connected to the system database. PDO Parameterized Queries is able to prevent various types of SQL Injection attack [34]. Here, the authors talk about the use of SQLMap to protect the attack (SQLI + DNS). The SQLMap has the DNS Ex filtration feature and there are many command lines designed specifically to prevent and detect DNS. It is compatible with most versions of SQL database [23].

1. Validation of the parse tree to prevent attacks with SQL injection, The author adopted the parse tree framework technique. The technique is based on the comparison of the Tree Validation analysis of the SQL statement before the user input is included when the input is included to be visualized if it adapts to a model of that were expected in the queries [1].

2. Safe Query Objects Statically Typed Objects As Remotely Executable Queries: Statically type objects as remotely executable queries, The Author proposes this technique. They used an API to access the database. This API can perform time compile checks on query objects. This technology supports the strong type monitoring approach. Safe query objects support

query shipping so that they are efficiently executed between a database engine. Compile time metaprogramming is used to improve secure query objects with strategies that send queries to a relative execution database [2].

3. Use of query tokenization for detecting and preventing SQL injection attacks: The author has proposed a technique consisting of tokenizing each initial query and also the one with an injection When this can be done each token constitutes an array index and two arrays are formed at last Their work involves the implementation of a technique that identifies a single quote space or double dash A token is represented by all strings before a quote before a space or before double dashes All tokens sorted to form an array The tokenization for each initial query and thus the query with an injection is completed The arrays which are obtained are then compared as required and then an injection is detected if their lengths differ otherwise if their lengths differ this indicates that there is no injection at the moment [9].

4. Using SQL Pattern Matching Database System: This Author proposed this technique in order to prevent SQL Injection ,In this technique the Known SQL Patterns are stored in the database and every query is checked upon with the pattern through the database to know whether the query is a standard and non-harmful query or it is a malicious query which matches the SQL injection pattern in the database.

X. DATASET

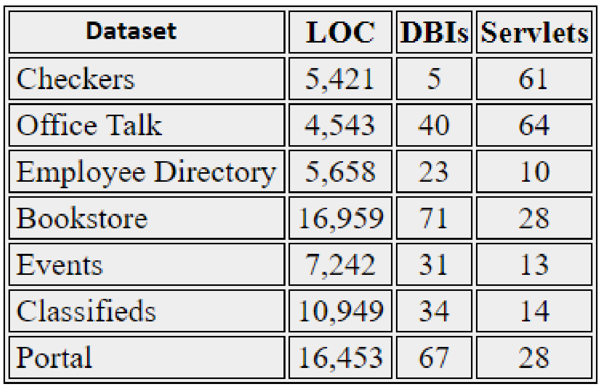
The paper will be using three dataset which are: Gotocode Dataset. Gotocode contains 5 different type of dataset those are:

1) Employee Directory Dataset
2) Bookstore Dataset
3) Events Dataset
4) Classifieds Dataset
5) Portal Dataset

Checkers Dataset: Checkers is a game application which is developed by students and their dataset is available to the public.

OfficeTalk Dataset: OfficeTalk is a purchase order management system which is developed by students and their dataset is available to the public.

TABLE 1.2. Dataset information



Dataset	LOC	DBIs	Servlets
Checkers	5,421	5	61
Office Talk	4,543	40	64
Employee Directory	5,658	23	10
Bookstore	16,959	71	28
Events	7,242	31	13
Classifieds	10,949	34	14
Portal	16,453	67	28

XI. COMPARISON TECHNIQUE

There are many ways to make comparison nonetheless here in this paper, we first selected the SQL Injection Attack technique method based on the results of the SQL injection Attack. The comparison will be done using IBM SPSS software based on the statically output through the SQL injection techniques method. There are two types of the SQL injection attack technique method that can be divided into two (2) parts which are Automatic Technique and Manual Technique which would be further explained below:

A. Automatic MySQL Injection Technique:

SQL Injection attacks can be done through automatic matter which means by using the software which automatically scans the web application for SQL Injection vulnerabilities based on the type of database selected and thus automatically showing the result after test is completed. Below are the software which will be used for the Automatic MySQL injection technique:

1. SQLMap.
2. Havij .
3. Safe3 SQL injector.

B. Manual MySQL Injection Technique

SQL injection is a security exploit in which the attacker adds malicious SQL code to a Web form input box to gain access. Manual testing for SQL injection is done through the browser in where numerous of malicious SQL query code as mentioned in section 2.7 are entered into web form input box in order to check whether the web application is vulnerable to that malicious SQL query code or not.

When testing whether there is a SQL injection vulnerability in a Web application system, the simplest test method is to attach an eternal (run as true) or a false (run as false) logic condition to the suspicious injection point, and view the execution result. If security detection has been added to the system, relevant prompt information will appear or the page will not be displayed properly. If the security test is not carried out, and the SQL command is joined directly to the database system, and the additional real-time injection of SQL command execution results are consistent with the results of no additional injection information, while the injection of SQL commands with a false value will cause the page to fail to display properly. Th employee's number is also used to query employee information. The test process is shown in Fig. 2:

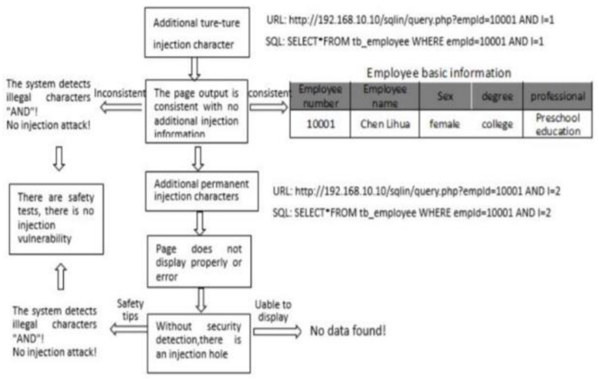

Additional ture-ture injection character
URL: http://192.168.10.10/sqlin/query.php?empld=10001 AND l=1
SQL: SELECT*FROM tb_employee WHERE empld=10001 AND l=1
The system detects illegal characters "AND"! No injection attack!
Inconsistent
The page output is consistent with no additional injection information
consistent
Employee basic information
Employee number
Employee name
Sex
degree
professional
10001
Chen Lihua
female
college
Preschool education
There are safety tests, there is no injection vulnerability
Additional permanent injection characters
URL: http://192.168.10.10/sqlin/query.php?empld=10001 AND l=2
SQL: SELECT*FROM tb_employee WHERE empld=10001 AND l=2
Page does not display properly or error
The system detects illegal characters "AND"! No injection attack!
Safety tips
Without security detection,there is an injection hole
Uable to display
No data found!

Fig. 2. Test process of SQL Injection (Manually)

XII. DISCUSSION

The SQL attack prevention method is implemented in order to prevent and protect the confidential data from hackers and unauthorized party. However, SQL attack prevention method is not easily achieved just by implementing of any method without doing comparison between the available methods proposed by the various different authors. Every SQL injection prevention method has their own advantages and disactivates, one method maybe able to provide what the organization required but might lack in the other requirements. this Paper will be focusing on comparing different types of SQL injection prevention methods as written in section VI which are best to be implemented. The main goal of this paper is review various types of SQL injection attacks, SQL injection prevention methods and also to show the readers on the comparisons can be done for different types of SQL prevention methods. In order for them to select the best SQL prevention method for their organization.

XIII. CONCLUSION

This chapter presents the literature review on an Comparison on the efficacy of the SQL Prevention Methods. Also it includes the risk associated with SQL injection and various SQL Injection Prevention Techniques. In addition, in this chapter include an insight on the requirement that shall be applied in the following chapter three, which includes the process, methodology, dataset used.

XIV. SUMMARY

The paper talks about various types of common SQL injection attacks that organization face, the risk associated to this types of SQL injection attacks and the types of SQL prevention methods which can help organization to safeguard against SQL attacks.

REFERENCES

[1] Buehrer, G., B. W. Weide, and P. A. G. Sivilotti. (2005). Using Parse Tree Validation to Prevent SQL Injection Attacks. *Proceedings of the 5th International Workshop on Software Engineering and Middleware. ACM: Lisbon, Portugal*, 106-113.

[2] Cook, W. R. and S. Rai. (2005). Safe Query Objects: Statically Typed Objects as Remotely Executable Queries. *Proceedings of the 27th International Conference on Software engineering. ACM: St. Louis, MO, USA,* 97-106.

[3] Michael Martin, B. L., Monica S. Lam. (2005). Finding Application Errors and Security Flaws Using PQL: A Program Query Language.

[4] William, G. J. Halfond, A. O. (2005). Preventing SQL Injection Attacks Using AMNESIA.

[5] Halfond, W. G. J. and A. Orso. (2007). Detection and Prevention of SQL Injection Attacks. *Malware Detection*, 85-+.

[6] Halfond, W., A. Orso, and P. Manolios. (2008). WASP: Protecting Web Applications Using Positive Tainting and Syntax-Aware Evaluation. *IEEE Transactions on Software Engineering*, 34(1), 65-81.

[7] Wang, H. F., W. Y. Chen, and S. L. Song. (2009). Design of Jinan City Flood Prevention and Warning Decision-Making Support System based on SQL Server and GIS. *First International Workshop on Database Technology and Applications, Proceedings*, 488-492.

[8] Bisht, P., P. Madhusudan, and V. N. Venkatakrishnan. (2010). CANDID: Dynamic Candidate Evaluations for Automatic Prevention of SQL Injection Attacks. *ACM Transactions on Information and System Security*, 13(2).

[9] Ntagwabira, L. and S. L. Kang. (2010). Use of Query Tokenization to Detect and Prevent SQL Injection Attacks. *2010 3rd International Conference on Computer Science and Information Technology.*

[10] Tajpour, A., *et al.* (2010). SQL Injection Detection and Prevention Tools Assessment. *Proceedings of 2010 3rd IEEE International Conference on Computer Science and Information Technology*, 9(ICCSIT 2010), 518-522.

[11] Tajpour, A. and M. J. Z. Shooshtari. (2010). Evaluation of SQL Injection Detection and Prevention Techniques. 2010 Second International Conference on Computational Intelligence. *Communication Systems and Networks (CICSYN),* 216-221.

[12] Narayanan, S. N., A. R. Pais, and R. Mohandas. (2011). Detection and Prevention of SQL Injection Attacks Using Semantic Equivalence. *Computer Networks and Intelligent Computing*, 157, 103-112.

[13] Selvamani, K. and A. Kannan. (2011). A Novel Approach for Prevention of SQL Injection Attacks Using Cryptography and Access Control Policies. *Advances in Power Electronics and Instrumentation Engineering*, 148, 26-+.

[14] Zhang, X. Z. and X. J. Zhang. (2011). Discussion on the Detection and Prevention of SQL Injection. *Applications of Engineering Materials,* Pts 1-4, 287-290, 3047-3050.

[15] Balasundaram, I. and E. Ramaraj. (2012). An Efficient Technique for Detection and Prevention of SQL Injection Attack using ASCII Based String Matching. *International Conference on Communication Technology and System Design 2011*, 30, 183-190.

[16] Cherry, D. (2012). *Securing SQL Server: Protecting Your Database from Attackers*. 2nd ed. Waltham, MA: Syngress/Elsevier. xxii, 381.

[17] Clarke, J. (2012). *SQL Injection Attacks and Defense.* Waltham, MA: Elsevier. xvviii, 547.

[18] Kumar, P. and R. K. Pateriya. (2012). A Survey on SQL Injection Attacks, Detection and Prevention Techniques. 2012 *Third International Conference on Computing Communication & Networking Technologies (ICCCNT).*

[19] Shafie, E. and A. Cau. (2012). A Framework for the Detection and Prevention of SQL Injection Attacks. *Proceedings of the 11th European Conference on Information Warfare and Security*, 329-336.

[20] Kar, D. and S. Panigrahi. (2013). Prevention of SQL Injection Attack Using Query Transformation and Hashing. *Proceedings of the 2013 3rd IEEE International Advance Computing Conference (IACC),* 1317-1323.

[21] Munqath, H. Alattar, S. P. M. (2013). R-WASP: Real Time-Web Application SQL Injection Detector and Preventer.

[22] Sadeghian, A., M. Zamani, and A. Abd Manaf. (2013). A Taxonomy of SQL Injection Detection and Prevention Techniques. *2013 International Conference on Informatics and Creative Multimedia (ICICM*), 53-56.

[23] Stampar, M. (2013). Data Retrieval over DNS in SQL Injection Attacks.

[24] Djanali, S., *et al*. (2014). SQL Injection Detection and Prevention System with Raspberry Pi Honeypot Cluster for Trapping Attacker. *2014 1st International Symposium on Technology Management and Emerging Technologies (ISTMET 2014),* 163-166.

[25] Doshi, J. C., M. Christian, and B. H. Trived, (2014). SQL FILTER-SQL Injection Prevention and Logging Using Dynamic Network Filter. *Security in Computing and Communications,* 467, 400-406.

[26] Tajpour, A., *et al.* (2014). SQL Injection Detection and Prevention Tools Assessment. *Proceedings of the 2014 International Joint Conference on Neural Networks (IJCNN),* 730-731.

[27] Wang, J., X. S. Cheng, and L. S. Ren. (2014). Construction of Knowledge Base for Prevention and Control of Cucumber's Diseases and Insect Pests Based on SQL Server 2005. *2014 4th International Conference on Education and Education Management (EEM 2014),* Pt 6, 68, 197-201.

[28] Hanmanthu, B., B. R. Ram, and P. Niranjan. (2015). SQL Injection Attack Prevention Based on Decision Tree Classification. *Proceedings of 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO).*

[29] Masri, W. and S. Sleiman. (2015). SQLPIL: SQL Injection Prevention by Input Labeling. *Security and Communication Networks*, 8(15), 2545-2560.

[30] Qian, L., *et al.* (2015). Research of SQL Injection Attack and Prevention Technology. *Proceedings of 2015 International Conference on Estimation, Detection and Information Fusion ICEDIF 2015*, 303-306.

[31] Ben-Ghorbel-Talbi, M., F. Lesueur, and G. Perrin. (2016). Information Flow Control on a Multi-paradigm Web Application for SQL Injection Prevention. Foundations and Practice of Security (FPS 2015), 9482, 277-285.

[32] Chen, P., *et al.* (2016). Research and Implementation of SQL Injection Prevention Method Based on ISR. 2016 2nd IEEE International Conference on Computer and Communications (ICCC), 1153-1156.

[33] Kamtuo, K. and C. Soomlek. (2016). Machine Learning for SQL Injection Prevention on Server-Side Scripting. *2016 20th International Computer Science and Engineering Conference (ICSEC).*

[34] Sendiang, M., A. Polii, and J. Mappadang. (2016). Minimization of SQL Injection in Scheduling Application Development. *2016 International Conference on Knowledge Creation and Intelligent Computing (KCIC).*

[35] Singh, N., *et al*. (2016). SQL Injection: Types, Methodology, Attack Queries and Prevention. *Proceedings of the 10th Indiacom - 2016 3rd International Conference on Computing for Sustainable Global Development,* 2872-2876.

[36] Voitovych, O. P., O. S. Yuvkovetskyi, and L. M. Kupershtein. (2016). SQL Injection Prevention System. *2016 International Conference Radio Electronics & Info Communications (UkrMiCo).*

[37] Ghafarian, A. (2017). A Hybrid Method for Detection and Prevention of SQL Injection Attacks. *2017 Computing Conference,* 833-838.

[38] Hu, H. B. (2017). Research on the Technology of Detecting the SQL Injection Attack and Non-Intrusive Prevention in WEB System. *Materials Science, Energy Technology, and Power Engineering I,* 1839.

[39] Nadeem, R. M., *et al.* (2017). Detection and Prevention of SQL Injection Attack by Dynamic Analyzer and Testing Model. *International Journal of Advanced Computer Science and Applications,* 8(8), 209-214.

[40] Armiati, S. and R. M. Awangga. (2018). SQL Collaborative Learning Framework Based on SOA. *International Conference on Mechanical, Electronics, Computer, and Industrial Technology,* 1007.

.