International Journal For Technological Research In Engineering
Volume 6, Issue 4, December-2018 *ISSN (Online): 2347 - 4718*

NOVEL ANALYTICAL ALGORITHM SQL INJECTION DETECTION

Divya Kumari Saini[1], Shalini[2], Dr. Swapnil Singhal[3]
[1]M.Tech Scholar, [2,3]Assistant Professor, [1,2,3]CSE Dept.JIT, Jaipur, Rajasthan.

Abstract: All Websites and database applications are susceptible to the attack of SQL Injection , which results in the loss of the crucial data as well as sometimes results in the loss of the greater amount of money. This paper proposes Novel algorithm for SQL Injection Detection and can be integrated in application for the detection purpose..

I. INTRODUCTION

SQL injection vulnerabilities have been depicted as a standout amongst the most genuine dangers for Web applications [1]. Web applications that are helpless against SQL injection may enable an attacker to increase finish access to their hidden databases. Since these databases frequently contain delicate shopper or client information, the subsequent security infringement can incorporate wholesale fraud, loss of confidential information, and extortion. At times, attackers can even utilize a SQL injection powerlessness to take control of and degenerate the framework that has the Web application. Web applications that are helpless against SQL Injection Attacks (SQLIAs) [1] are across the board—an examination by Gartner Group on more than 300 Internet Web locales has demonstrated that the vast majority of them could be powerless against SQLIAs. Indeed, SQLIAs have effectively focused on prominent casualties, for example, Travelocity, FTD.com, and Guess Inc. SQL injection alludes to a class of code-injection attacks in which data given by the client is incorporated into a SQL inquiry such that piece of the client's information is dealt with as SQL code. By lever-maturing these vulnerabilities, an attacker can submit SQL orders straightforwardly to the database. These attacks are a genuine risk to any Web application that gets contribution from clients and joins it into SQL inquiries to a hidden database. Most Web applications utilized on the Internet or inside big business frameworks work along these lines and could in this manner be helpless against SQL injection [1]. SQL injection exposures have been imparted extraordinarily dangerous for the database. Key databases are completely accessible by attacker by infusing SQL inquiries that are recovered by web application. As customer information is every now and again kept in these databases, critical information is lost and the security break. Attackers can even utilize a SQL injection introduction is utilized by attackers for controlling and influencing the web application to structure worse [2]. A class of code-injection attacks is pointed by SQL Injection; customer gives the data which is joined into a SQL question such that piece of the customer's information to be known by SQL codes.SQL orders given by attacker straight away to the database, through these vulnerabilities. These attacks are unsafe to any Web application that gets data from customers and obliges it into SQL request to a key database [2]. SQL Injection is a type of web application security vulnerability in which an attacker is able to submit a database SQL command, which is executed by a web application, exposing the back-end database. SQL Injection attacks can occur when a web application utilizes user-supplied data without proper validation or encoding as part of a command or query.

The specially crafted user data tricks the application into executing unintended commands or changing data. SQL Injection allows an attacker to create, read, update, alter, or delete data stored in the back-end database. In its most common form, SQL Injection allows attackers to access sensitive information such as social security numbers, credit card number or other financial data. According to Veracode's State of Software Security Report SQL Injection is one of the most prevalent types of web application security vulnerability [2].

For the better understanding of SQLIA one must have the cognizance of web application architecture. Web applications are a set of web pages and programs which reside on a web server . The inputs provided by the user are sent to the server in the form of parameter string. These inputs are used to engender SQL query to retrieve information from the database. An authorized user can access it over the cyber world or over a public network and store the data in the database. A web application utilizes a web browser as an interface to extract the data from database server to accommodate the queries placed by the users. Every web application is predicated on 3–tier architecture consisting of three layers [48] .Each layer can run potentially on a different machine and each layer should be independent of other layers. The three layers are

Presentation Layer: Presentation layer contains presentation logic. It is the top most level of application and handles the interactions with users. Its main function is to receive input from the user and provide the result in a convenient way that user can facilely understand.

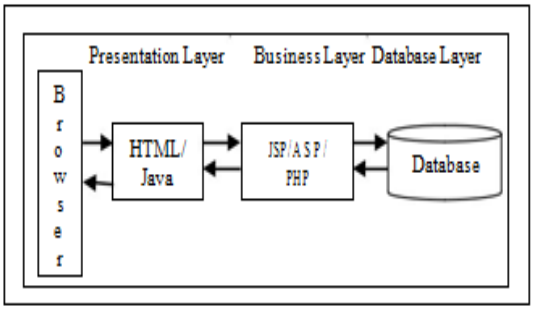

Fig 1. Architecture of web application

Business Layer: This layer is present in between presentation layer and database layer. It is a logic layer which consists of a set of rules for processing the information between two layers. It contains application process commands which retrieves the data from database and sends to presentation

layer for viewing the data. This tier can be programmed in any server scripting language like JSP, PHP, and ASP etc.
Database Layer: This is a physical storage layer for data persistence. It manages all access to database and file system. Information is stored and retrieved from database .It is then passed back to the presentation layer for processing and eventually back to the user. The main function of this layer is to provide access to authorized user and restrict the maleficent user. Working principle of architecture: The presentation layer receives the request from web browser, processes it and then passes the dynamic part to the business layer which processes server scripting languages. All requests for database access are passed to the database server. The result is then dispatched to web browser as web pages. This architecture is easy to maintain and all the components are reusable. For the faster and smooth functioning, all the layers are governed and managed by different groups of experts. Web designer looks after the presentation layer. Software engineer does the logic and database administrator manages the database servers.

II. RELATED STUDY

Archana Gupta, Dr. Surendra Kumar Yadav [1] SQL injection attacks are one of the highest dangers for applications composed for the Web. These attacks are dispatched through uncommonly made client information on web applications that utilization low level string operations to build SQL queries.SQL injection weakness permits an assailant to stream summons straightforwardly to a web application's hidden database and annihilate usefulness or privacy. In this paper we proposed a simplified algorithm which works on the basis features of the SQL Injection attacks and will successfully detects almost all types of the SQL Injection attacks. In the paper we have also presented the experiment results in order to acknowledge the proficiency of our algorithm. A new algorithm is presented to protect Web applications or even the desktop application against SQL injection Attacks. SQL Injection Attacks are a class of attacks that many of these systems are highly vulnerable to, and there is no known foolproof defense against such attacks. Some predefined methods and integrated approach of encryption method with secure hashing can be applied in the database to avoid attack on login phase. This combined method will be applied to a system where user‘s information is kept and the designing of this system will be done by using .Net.
Rhythm Dubey, Himanshu Gupta [2] Web applications operate even the smallest thing on the internet these days. If they take online banking system, e-grocery shopping, e-shopping, reservations etc. and their applications need to be served and reliable. All the information related to items and their transactions are stored in database. But this database is highly prone to SQL injection attacks these days and their attacks emerged as security threat to web applications and valuable information stored in vulnerable database. They will propose a technique which will be a combination of two security services for maintaining the confidentiality, integrity and authentic of data in more efficient way. The working of a proposed technique is a 2 stage process, the first phase checks the queries that the user wants to execute. This takes place at the proxy server and the second phase will check the login credentials of the user.
Vamshi Krishna Gudipati, Trinadh Venna, Soundarya Subburaj, Omar Abuzaghleh [3] A wide variety of data such as credit information, military data, human communication data, and countless types of data is shared over the far-flung computer networks. As the usage and reliability on computers increase, the threat to sensitive data likewise increases. The challenges with the cyber security when dealing with sensitive information is now a nightmare. To help understand the threats and the severity of exploits deployed, the paper provides proof of concepts for exploits carried out to compromise web applications and how the databases are exploited using the SQL injection methodologies. The SQL injection vulnerabilities in the web applications are surprisingly very vast and this is definitely is a huge security threat to personal data of people that is stored on web. In this paper, the methods used in information gathering, how the security is breached, and how payloads are used to exploit web applications are explained using the Kali Linux. In addition, an analysis is carried out on how the websites are comprised. Advanced methods on how to defend SQL injections are briefly justified. For the readers to understand better, a real time scenario of a penetration tester and a database server is set up with a few suppositions, and the commands that dodge the security characteristics and manipulate the databases are explicated.
Raja Prasad Karuparthi, Bing Zhou [4] This paper proposes an enhanced approach to dynamic query matching technique by imposing a sanitizer for quick and easy detection of attack. This paper presents an enhanced approach of DUD by proposing a SQLI sanitizer in the flow which enables a detection of attack at the initial level, by minimizing the utilization of time in processing. This enables the detection of unknown attacks which are weird to the common people. A combination of matching strategies can be used to improve the efficiency by imposing certain constraints on the user’s input query parameters. A combination of efficient matching techniques would derive an effective similarity in verifying with the known SQLI attacks.
William G.J. Halfond, Jeremy Viegas, and Alessandro Orso [5], notified that SQL injection attacks pose a serious security threat to Web applications: they allow attackers to obtain unrestricted access to the databases underlying the applications and to the potentially sensitive information these databases contain. Although researchers and practitioners have proposed various methods to address the SQL injection problem, current approaches either fail to address the full scope of the problem or have limitations that prevent their use and adoption. Many researchers and practitioners are familiar with only a subset of the wide range of techniques available to attackers who are trying to take advantage of SQL injection vulnerabilities. As a consequence, many solutions proposed in the literature address only some of the issues related to SQL injection. To address this problem, they present an extensive review of the different types of SQL injection attacks known to date. For each type of attack, they provide descriptions and examples of how attacks of

that type could be performed. They also present and analyze existing detection and prevention techniques against SQL injection attacks. For each technique, they discuss its strengths and weaknesses in addressing the entire range of SQL injection attacks.

III. PROPOSED WORK

In our proposed concept we have proposed an algorithm, which will be used for performing a check that the query fired by the user is an SQL Injection or not.

First the Query is provided as input in the form which we created for the Query Analysis

↓

Handling of the Nested Queries , firstly the components of the queries are separated by searching for the () parenthesis , in order to find the presence of any of the inner query and once the query presence is identified then the two or more than two queries are processed separately and have to undergo all the remaining steps of the algorithm

↓

In the First Check the Query is check for the DROP keyword as , to avoid SQL Injection which can delete the table structure

↓

In the Second check we check for the validity of the SQL statement , in order to check whether it is proper SQL statement i.e. begin with SELECT,INSERT etc..

↓

In the third check we will aviod the SQL Injection for the value '1'='1' , this type of injection can be given in various ways , so we implemented this in two sub section , firstly containing OR statement , where we split the query on the basis of OR keyword and then checked the parameters for similarity and if same then it SQL Injection Attack and second a simple Query which contains only statements like '1'='1' is handled after checking presence of = and checking parameters for equality. And also work for >,<,<> operators

↓

Then we have check for the queries with intension of knowing the tables in the databases.

↓

Finally we have check the queries with have no results , just fired in order to know the table structure.

Fig. 2 Flowchart for the Proposed Algorithm

IV. TESTING AND IMPLEMENTATION

I. Query Related to Tautologies using Relational Operators

Consider the following statement,

select * from employee where emp_id=’e001’ and 5>2

This query will retrieve the data from employee table.

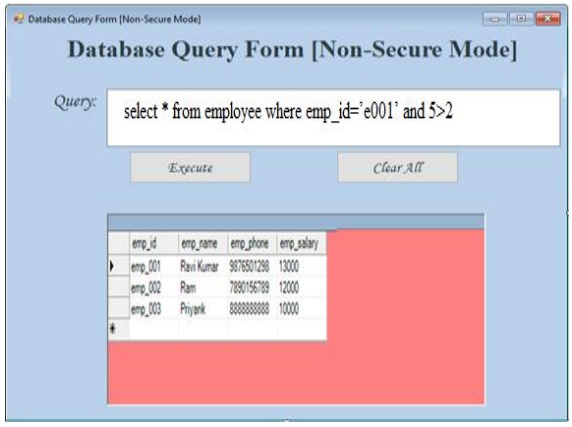

Fig 3 Non-Secure Demonstration for Case I

In the proposed concept, we have tautology related to relational operator, so will able to detect the queries related to it.

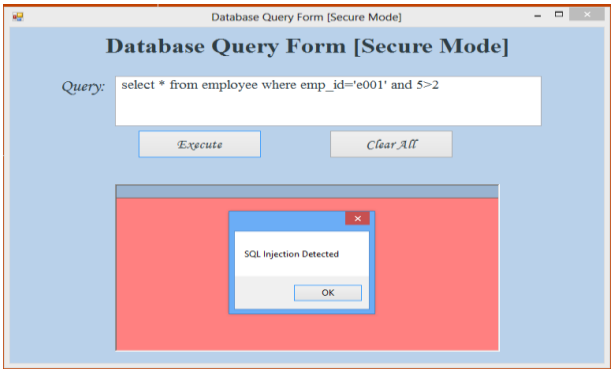

Fig 4 Secure Demonstration for Case I

II. Query Related to UNION and INSTERECTION

Consider the following statement, select * from employee UNION select * from register where 5>2 This query will retrieve the data from employee table and register table. The base implementation crash in execution of this query. In the proposed concept, we have tautology related to relational operator even in UNION, so will able to detect the queries related to it.

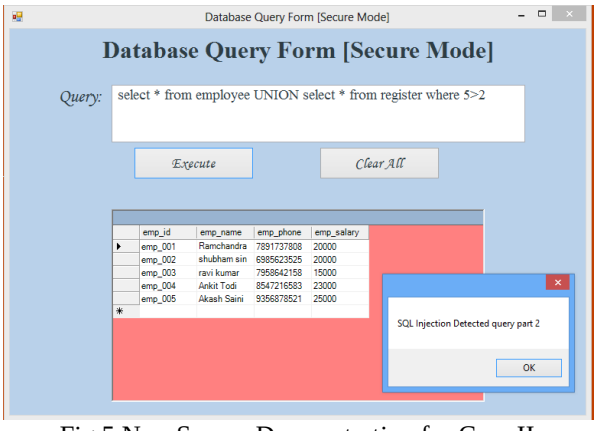

Fig 5 Non-Secure Demonstration for Case II

V. CONCLUSION

The result obtained via the proposed algorithm are quite impressive and the lot work is to be done in this field and continuous research is required in this field.

REFERENCES

[1] Archana Gupta, Dr. Surendra Kumar Yadav, "An Approach for Preventing SQL Injection Attack on Web Application", International Journal of Computer Science and Mobile Computing(IJCSMC), vol. 5, Issue 6, pp. 01-10, June 2016

[2] Rhythm Dubey, Himanshu Gupta, "SQL Filtering: An Effective Technique to Prevent SQL Injection Attack", IEEE 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), pp. 312-317, 7-9 September 2016

[3] Vamshi Krishna Gudipati, Trinadh Venna, Soundarya Subburaj and Omar Abuzaghleh, "Advanced Automated SQL Injection Attacks and Defensive Mechanisms", IEEE Annual Connecticut Conference on Industrial Electronics, Technology & Automation (CT-IETA), pp. 1-6, 14-15 October 2016

[4] Raja Prasad Karuparthi, Bing Zhou, "Enhanced Approach to Detection of SQL Injection Attack", IEEE International Conference on Machine Learning and Applications, pp. 466-469, 18-20 December 2016

[5] William G.J. Halfond, Jeremy Viegas and Alessandro Orso, "A Classification of SQL Injection Attacks and Countermeasures", IEEE, 2006

[6] Chandershekhar Sharma, Dr. S. C. Jain, "SQL Injection Attacks on Web Applications", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), vol. 4, Issue 3, pp. 1268-1272, March 2014

[7] Bojken Shehu, Aleksander Xhuvani, "A Literature Review and Comparative Analyses on SQL Injection: Vulnerabilities, Attacks and Their Prevention and Detection Techniques", International Journal of Computer Science Issues(IJCSI), vol. 11, Issue 4, no. 1, pp. 28-37, July 2014

[8] Ashish John, "SQL Injection Prevention by adaptive algorithm", IOSR Journal of Computer Engineering, vol. 17, pp. 19-24, January 2015.

[9] Shubham Mukherjee, Pritam Sen, Sudeshna Bora and Chittaranjan Pradhan, "SQL Injection: A Sample Review", IEEE Computing, Communication and Networking Technologies (ICCCNT), pp. 1-7, 13-15 July 2015

[10] Pankajdeep Kaur, Kanwal Preet Kour, "SQL Injection: Study and Augmentation", IEEE International Conference on Signal Processing, Computing and Control (ISPCC), pp. 102-107, 24-26 September 2015

[11] Tejinderdeep Singh Kalsi, Navjot Kaur, "Detection and Prevention of SQL Injection Attacks using Novel Method in Web Applications", International Journal of Advances in Engineering and Technology(IJAET), vol. 6, Issue 4, pp. 11-15, December 2015

[12] Subhranil, Som Sapna Sinha and Ritu Kataria, "Study on SQL Injection Attacks: Mode, Detection and Prevention", International Journal of Engineering Applied Sciences and Technology (IJEAST)", vol. 1, Issue 8, pp. 23-29, 2016

