Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 4, Issue 12, December -2017

Proposed Adaptive Algorithms to Prevent SQL Injections

Vamsi Mohan V[1], Dr. Sandeep Malik[2]

[1]*Department of Computer Science, School of Engineering and Technology, Raffles University, Neemrana, India.*
[2]*Dean, Yogananda College of Engineering & Technology, Jammu - 181205, India*

Abstract - *For the last two decades, SQL injection attacks are considered as common for Web applications. By growing of digitalization, the security attacks are increasing and most commonly SQL Injection Attacks (SQLIAs). Injection attacks considered in the first place (A1) as per the OWASP Top 10 Application Security Risks for the year 2017. To overcome these attacks, many researchers and scientists proposed various algorithms to adopt. However, software industry has not completely succeeded from the SQL injection attacks and vulnerabilities. In this paper, we examined various algorithms proposed by different researchers to prevent the SQL injections.*

Keywords:*SQLIA, Parse Tree Validation, Code Conversion, SQL query; Dynamic method; SQL-injection free algorithm; Runtime environment;*

I. INTRODUCTION

SQL injection attacks prey upon the fact that many modern web applications depend upon underlying databases to validate user requests and response. There are many researchers surveyed on various available SQL injection techniques and proposed algorithms. It is multifaceted for web intruders when we are talking about Security and privacy of database driven applications. Many open source and LGPL tools are available in the market that are proved not a single detection scanner provides the best result for various cyber-attacks.

II. LITERATURE REVIEW

Kanchana Natarajan, Sarala Subramani (2012), stated in their publication that one of the most dangerous cyber-attacks is the SQL-injection attack, which simply creates huge loss to commercial vendors. Research deliberates to provide SQL-injection free (SQL-IF) secure algorithm to detect and prevent SQL-injection attacks (SQLIAs). In their paper, they re-addressed several detection methods to conflict against the proposed SQL-IF secure algorithm. The generated algorithm has been integrated into the runtime environment while the implementation has been done through Java. The algorithm describes the method that how they follow the procedures for preventing SQL-injection attacks. They presented the SQL-IF secure algorithm and logic of the generated code. Comparison of similar types of attack along with different features is performed. The empirical results and its evaluation prove that the algorithm works efficiently to detect the SQLIAs.

Kanchana Natarajan, Sarala Subramani mentioned that SQL injections are more lucrative for attackers as they primarily focus to steal the bank account numbers, credit card numbers, etc. This kind of security issues on web applications are more susceptible, and can be handled by user authentications. Many forms of SQL injection attacks exist. However, most common takes the benefits of erroneously passed parameters, erroneous type handling, erroneous use of SQL statements, for e.g. (‘OR) 1 = --’). Industry familiar SQL injection attacks are tautologies, illegal/logically incorrect queries, UNION query, Piggy-backed queries, Stored Procedures, Blind SQL, Timing Attack, Alternate Encoding etc.

In their survey, Ashish John, Ajay Agarwal, Manish Bhardwaj (2015) on existing techniques against SQL Injection and analyzed the advantages and disadvantages of current techniques and proposed a novel and effective solution to avoid attacks on login phase. According to them the lack of adequate knowledge and understanding of software developers on security engineering and vulnerabilities leads to poor coding, e.g. by inappropriate programming, getting worse under deadline pressure and rush to production bugs. Thereare some readymade solutions may be available in the market today. But due to technology changes, new risks and challenges appearing day by day. To address such kind of vulnerabilities, different types of solutions must be combined together to be effective against attacks and the security of the system must be constantly monitored.

In their publication, Ashish John et al. said, the back-end database often contains confidential and sensitive information such security numbers, credit card number, financial data, medical data. Typically, when the user sends information, such as a username and password to login the system by interacting with the back-end database, relevant data will be returned to the browser. Some of the commonly performed web attacks are: Injection attacks (SQLIA), XSS Attack, Security Misconfiguration etc. According to OWASP (Open Web Application Security Project) Injection attack is at the first three places of the top 10 web attacks that are executed in the past 10 years. SQL injection is a method for exploiting the web applications that use client-supplied data in the SQL queries. SQL Injection refers to the technique of inserting SQL meta-characters and commands into Web-based input fields to manipulate the execution of the back-end SQL queries. The SQLIA occurs when an intruder changes the structure of the query by inserting any SQL commands.

In their paper, ***an adaptive algorithm to prevent SQL injection***, Ashish John et al. proposed a simple and effective method to detect SQL Injection Attacks, which uses the combination of Parse Tree Validation Technique and Code Conversion Method.
Parse Tree validation technique helps to parse the user input characters and filter the special characters from the input. On top of the parsing technique, Ashish John et al. implemented code conversion for the second layer of security.

Background history of related work on prevention of SQL injection

1. Parse Tree Validation Technique is based on comparing, at run time, the parse tree of the SQL statement before inclusion of user input with that resulting after inclusion of input.
2. Code Conversion Method is used to convert User input to code like ASCII, binary, hexa etc., and Searching the availability of converted input in Data table and returns valid Userid and Password.

Ashish John et al., proposed a method that consists of the best features of both parse tree validation technique and code conversion method. In this method, they parsed the user input and check whether it is vulnerable, if there is any chance of vulnerability present then code conversion will be applied over that input. In this way, they suggested to detect and prevent SQL Injection using a single code. Below is the algorithm for the proposed method:

III. PROPOSED ALGORITHM - I

For web pages that saves data to the database:
Algorithm: Proposed Algorithm

- Input the text.
- Apply Parse Tree Validation Technique on the text.
- Check if vulnerable (i.e., if tree size mismatch)
    a) If vulnerable, apply code conversion (say, ASCII to binary)
    b) Counter =1
    c) If not vulnerable
    d) Counter =0
- Save to database
- Exit

For web pages that only retrieves from the database:

- Check the value of counter
    a) If counter = 0
    b) If counter = 1
- Apply reverse code conversion (say, binary to ascii)
- Display the text
- Exit

From the value of counter, we can come to know whether the user input is converted or not.
While executing the algorithm proposed by Ashish John et al., we found many interesting results. Later reviewed the proposal of Kanchana Natarajan, Sarala Subramani on SQL Injection Free (IF) algorithm to prevent injection attacks.
Kanchana Natarajan, Sarala Subramani published in their journal that many researcher and authors explored different methods to detect and prevent SQLIAs; the most chosen techniques are static analysis, dynamic analysis, combined static and dynamic analysis, web framework, defensive programming and machine learning techniques. The method of static analysis is extreme were it analyzes the code for vulnerability by without executing the code. For example, Software metrics and reverse engineering are some forms of static analysis. Model checking, data flow analysis, abstract interpretation and use of assertions in source code are the several techniques of static code analysis. Dynamic analysis can be performed automatically by the analyzing the vulnerabilities during the execution of web applications which avoids thousands of tests by doing several times manually. Example: CANDID tool. Both the techniques have merits and demerits and therefore variations are identified from the efficacy. However, the research study analyzed with various existing methods and it has been proved dynamic analysis (penetration testing) tool is effective to test the web applications.
Penetration testing tools are easy to use and assure to provide security information systems to their users by fixing the security pitfalls before they get exposed to the outer world. Regarding the advantages of dynamic penetration testing, Kanchana Natarajan, Sarala Subramani mentioned (a) Not necessary to change the development lifecycle (b) Avoids static analysis challenges (c) No need for the source code, (d) Deployment-security.
The method of combined static and dynamic analysis can compensate the limitations of each method, which is considered as highly proficient against SQLIAs but it is very complicated. One of the best examples for such a method is AMNESIA (Analysis and Monitoring for Neutralizing SQL Injection Attacks) tool. It uses static analysis to analyze the web-application code and automatically build a model of the legitimate queries that the application can generate. At runtime, the technique monitors all dynamically-generated queries and checks for compliance with the statically-generated model. When the technique detects a query that violates the model, it classifies the query as an attack, prevents it from accessing the database, and logs the attack information. The web framework method is a filtering method of user

input parameters which is proven to be an in-effective while it is not able to filter some special characters. The machine-learning method is the most commonly used method whereas the method results in high false positives and low detection rate. Example: WAVES tool.

SQL-Injection Free (SQL-IF) Secure Algorithm

Kanchana Natarajan, Sarala Subramani proposed a new algorithm based on dynamic technique which violates the SQL injection attacks. This algorithm is concentrated to develop IF (Injection Free) attacks; where as a special type of test suite is developed to detect SQL injection attacks. The below shown Algorithm performs its function by assigning the Form Status (FS) as attack and free with collection of fields obtained from the collection of forms. The Form (fm') is obtained from the set of forms (Fm) whereas values of every field are obtained from the form (Fm'). Three well-defined functions are generated inside the method called CheckVulnerability (f') to check for any special characters, keywords and Boolean characters.

IV. PROPOSED ALGORITHM - II

Algorithm: Proposed SQL-IF Secure Algorithm

Algorithm SQLIAD (Form F)

1. Input: Fm denotes the collection of forms with collection of Fields.
2. Input: Enumerate Form_Status FS = {attack, free}
3. Input: Default Value to FS as free
4. Output: FS
5. for each fm' ∑ Fm do
6. for each f' ∑ fm' do
7. f' ← fields contains the values
8. if f' is not empty string then
9. FS ← output from the method CheckVulnerability (f')
10. if FS as attack
11. D ← Add the field f' in the collection
12. Reset the Http requests to issue warning;
13. return FS;

CheckVulnerability (f'):

```
for each f' fields do
        if f' is a non-empty string then
                // to check for special characters in the input fields and parameters
        p = collection of compiled special characters like {([',&+=<><=>=])}
        for each tokens ff' ∑ f' then
                ff'' = compile ff' to make all the input tokens neutralize.
                        v = comparison of P and ff';
                if v is not true then return v;
                        // to check for keywords in the input fields and parameters
        k=collectionof keywords {union|select|intersect|insert|update|delete|drop|truncate}
                for each tokens ff' ∑ f' then
                ff'' = compile ff' to make all the input tokens neutralize.
                        v = compare ff'' with k;
        if v is not true then return v;
                // to check for Boolean characters in the input fields and parameters
        b = collection of Boolean characters {' or '|'or'|'AND'|'and'"}
                for each tokens ff' ∑ f' then
                ff'' = compile ff' to make all the input tokens neutralize.
                        v = compare b with ff';
        return v;
```

The detection method is elaborated by given Generic detection method1, CheckVulnerability (f') which checks for the special characters, keywords and Boolean characters in the input fields. It compiles a collection of special characters like {([',&+=<><=>=])} and a collection of keywords like union, select, intersect, insert, update, delete, drop, truncate and Boolean characters like 'or '|'or', 'AND',' and with the actual input parameters to make all the input values neutralized to the database. If there is any mismatch in the parametric values, it is directly send to the vulnerability data collector and resets the Http request to warnings. The algorithm detects the SQLIAs which can be applied for any real web-based applications wherever the user and the database interacts. The detection algorithm is implemented through Java platform which is feasible to secure applications against SQLIAs.

The proposed algorithm is substantial in scrutiny of its simple detection mechanism against SQL injection attacks. Testing of web applications for SQL injection attack is a significant step for ensuring its performance and quality. The proposed algorithm performs much faster and endowed with proficient solution to resolve against SQL injection attacks. Post testing the proposed algorithms, it results that the algorithm proposed by Ashish John, Ajay Agarwal, Manish Bhardwaj is more secure and reliable than the proposed SQL Injection Free (SQL-IF) algorithm. In the algorithm proposed by Ashish John et al., the Parse tree validation filters the special characters from the user input, which secures the credentials and restricts from the query execution. After filtering the characters, the counter method helps to flag the final response.

V. CONCLUSION

After analyzing and reviewing various algorithms on injection attacks, it is evident that there should be robust mechanism is needed to prevent the SQL injections. There are many algorithms which can with stand till certain levels of Injection attacks. However, sophisticated techniques need to be proposed to overcome the SQL vulnerabilities and injection attacks.

REFERENCES

[1] Abdul Bashah Mat Ali, Ala' Yaseen Ibrahim Shakhatrehb, Mohd Syazwan Abdullahc, Jasem Alostadd, "SQL-injection vulnerability scanning tool for automatic creation of SQL-injection attacks", Journal of Procedia Computer Science, Elsevier Ltd, 2010, pages: 453-458.
[2] Ashish John, Ajay Agarwal, Manish Bhardwaj (2015)."An adaptive algorithm to prevent SQL injection".American Journal of Networks and Communications 2015; 4(3-1): 12-15
[3] Dimitris Mitropoulos, Diomidis Spinellis, SDriver: "Location-specific signatures prevent SQL injection attacks", Journal of Computers & Security, Elsevier Ltd, 2009, pages: 121-129.
[4] Indrani Balasundaram, E. Ramaraj "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 © 2011 Published by Elsevier Ltd. Selection and/or peer-review under responsibility of ICCTSD 2011
[5] Inyong Lee, Soonki Jeong, Sangsoo Yeo, Jongsub Moon, "A novel method for SQL injection attack detection based on removing SQL query attribute values", Journal of Mathematical and Computer Modeling, Elsevier Ltd, 2011, pages: 1-11.
[6] Kanchana Natarajan, Sarala Subramani (2012). "Generation of SQL-injection free secure algorithm to detect and prevent SQL-injection attacks".ELSEVIER, SciVerse ScienceDirect. Procedia Technology 4 (2012) 790 – 796, C3IT-2012.
[7] Lijiu Zhang, Qing Gu, Shushen Peng, Xiang Chen, Haigang Zhao, Daoxu Chen, "D-WAV: A Web Application Vulnerabilities Detection Tool Using Characteristics of Web Forms", IEEE Fifth International Conference on Software Engineering Advances, pages: 501-507, 2010.
[8] MeiJunjin, "An approach for SQL injection vulnerability detection", IEEE Sixth International Conference on Information Technology: New Generations, pages: 1411-1414, 2009.
[9] Oppliger, R., "Internet security enters the Middle Ages", Computer, vol.28, no.10, pp.100,101, Oct 1995 doi: 10.1109/2.467613
[10] Stephen Thomas, Laurie Williams, Tao Xie, "On automated prepared statement generation to remove SQL injection vulnerabilities", Journal of Information and Software Technology, Elsevier Ltd, 2009, pages: 589-598.
[11] W.G.J. Halfond, A. Orso, "AMNESIA: analysis and monitoring for Neutralizing SQL-injection attacks", 20th IEEE/ACM International Conference on Automated Software Engineering, Long Beach, CA, USA, 2005, pp. 174–183.
[12] Z. Su, G. Wassermann, "The essence of command injection attacks in web applications", 33rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Charleston, SC, USA, 2006, pages: 372–382.

