A Java Source-code SQL Injection Attack Detection Algorithm Based on Static Analysis

Wang Tian
Information Center
GuangDong Power Grid Corporation
GuangZhou,China
wangtian@gdxx.csg.cn

Wei Lihao
Information Center
GuangDong Power Grid Corporation
GuangZhou,China

Zou Hong
Information Center
GuangDong Power Grid Corporation
GuangZhou,China

***Abstract*—This paper researches the method of SQL injection attack detection and the principle of static analysis scanning, and presents a Java source-code SQL injection attack detection algorithm. The detection algorithm includes these steps: lexical analysis of source code, parsing of source code, constructing abstract syntax tree of source code, defining rules, abstract syntax tree traversal, tracking problems, detecting possible paths of SQL injection attack etc. Test results show the proposed detection algorithm in this paper performs perfectly and has higher recognition rate.**

Keywords- static analysis; SQL injection attack; abstract syntax tree.

I. INTRODUCTION

With the rapid development of computer technology and network technology, Web application software system has been wide used in all aspects of people's production and dairy life .It plays an important role in all the fields and gradually becomes an imperative and important component part of people's production and dairy life.

However, Web application software system development cycle is short and web programmers are lacking in rich experience and strong feeling of safe precaution, thus there are so many major safety risks and security loopholes in constructing website, which little by little makes lots of websites become objects of malicious users attack. Internet security is seriously jeopardized. SQL injection attack is one of the most common and serious risks among the security loopholes. It possesses characteristics of simplicity of operator, high performance of concealment, serious hazard and hard to being detected. When web application software system is attacked by SQL injection, catastrophic consequences will be brought about. Especially, if SQL injection attacks backend database systems of bank, stock, telecommunication mobile, government and e-commerce enterprises, it will generate immeasurable economic losses. What's more, Government website attacked or compiled by malicious users will cause a bad effect on society and will endanger social safety and stability being used by the wicked. How to detect SQL injection attack and how to take precautions against SQL injection attack are very useful and major significant great important.

This paper presents a SQL injection attack detection algorithm by analyzing Java source codes, setting up abstract syntax tree, defining rules and tree traversal. Web application program source code is carefully detected by this algorithm so that safety of Web application system is greatly improved.

II. ORDINARY SQL INJECTION ATTACK DETECTION ATTACK AND DEFENSE MEANS

Ordinary SQL injection attack main detection techniques are as follows:

(1) White box detection[1]: It is to check static web page in order to search for all paths where SQL injection attack may generate by static analysis and to inspect the quality of web page that being published before. For example, Fortify etc.

(2) Black box detection: It is to use a rule library to simulate hacker attacking application program and to pinpoint the problems by analyzing the results that the application program perform. For example, AppScan.

At present, there are main two means about SQL injection attack defense, such as platform level defense and code level defense. The common defense techniques are as follows:

(1) Server filtering[2]: It is to embed filtering modules web servers. Filtering modules find malicious codes and prevent web server from SQL injection attack, according to filtering rules analyzing Http input request. Its advantage lies in defending SQL injection attack.

(2) Restrict database permission: Database permission is limited in some application program, which may cut down various kinds of hackers attacking database.

(3) Parameterized query[3]:Some high-level programming languages offer database operations API substituting for using directly SQL to operate database and avoid SQL injection attack. This approach depends upon to a large extent high-level programming languages and has some limitations in application.

(4) Data filtering: It is to validate user input data and filter data by using white list as far as possible, or by using blacklist. For example, filtering dangerous single quote character etc.

(5) To a large extent，it is able to use storing process to defense SQL injection attack.

III. PRINCIPLE OF STATIC ANALYSIS SCANNING

Programming static analysis is not to execute the program and to find potential safety hazard in program with the algorithm automatically scanning[4]. It has the following advantages: quickly performing and high effectively etc. Source code with static analysis employs lexical analysis and parsing of compiling technology. The principle of static analysis is illustrated in Figure 1.

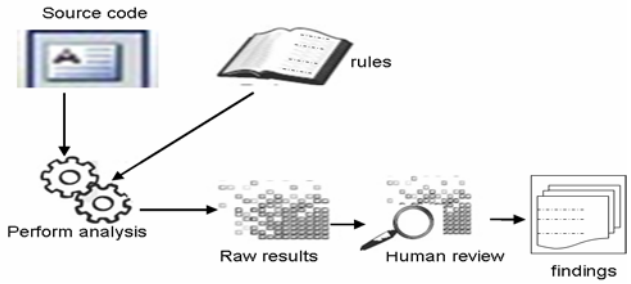

Figure1. The principle of source code with static analysis

IV. ALGORITHM DESCRIPTION

A. The proposed algorithm of Java source-code SQL injection attack detection

The grammar definition of version Java1.6 is pointed out. Based on the principle of static analysis, with lexical analysis of source code and parsing of source code, construct abstract syntax tree of source code according to program structure. Different nodes on the abstract syntax tree are marked by special signs in order to improve the efficiency of the proposed algorithm. The proposed SQL injection attack detection algorithm includes the following steps:

Step 1. Abstract syntax tree traversal . To search for all nodes named executeQuery , executeUpdate, execute or executeBatch under the branch node of METHOD_DECL and to store them in the Hash keyNode table.

Step 2. To confirm whether each of all nodes in the Hash keyNode table is at- risk or not .

Step 2.1 Firstly, to gain the former expression of the keyNode and to obtain types of return value about the former expression.

Step 2.2 If the type of return value about the former expression.is java.sql.Statement, this node can be validated. Then go to next step; Othewise, go to Setp 2 and analyze the next node.

Setp 3.Obtain the first parameter as the path of node about confirmed nodes, analyze and track data flow of nodes on the path:

Setp 3.1. Analyze and track nodes that are variables expression on the path

Setp 3.2. Analyze and track nodes that are methodological expression on the path.

Setp3.3. End the condition of tracking:

(1) If all the tracking variables are constant, there is not SQL injection attack on the path and go to Step 2.

(2) Track the methodological nodes that are API defined ,and go to Step 4.

Setp 4. Record and store paths of QL injection attack and go to Setp2,analyze the next the node. The algorithm stops when all nodes are detected in the keyNode table.

B. Test experiments

In order to evaluate performance of the proposed algorithm, the following Java source code is used to be tested. The beginning API and end API of SQL injection attack analyzed are seen in table 1. Those are rules of Java source code attack detection. Constructing abstract syntax tree according to source code is shown in Figure 2. By implementing the proposed algorithm, in the light of detection rules, risks in the program can be found and the paths of SQL injection attack may be reported.

TABLE I. RULES OF JAVA SOURCE CODE ATTACK DETECTION

API	Source Code
Begin API	javax.servlet.ServletRequest javax.servlet.http.HttpServletRequest java.lang.String getParameter(java.lang.String) java.lang.String[] getParameterValues(java.lang.String) java.util.Map getParameterMap() ……
End API	java.sql.Statement java.sql.ResultSet executeQuery(java.lang.String) int executeUpdate(java.lang.String) int executeUpdate(java.lang.String, int) int executeUpdate(java.lang.String, int[]) int executeUpdate(java.lang.String, java.lang.String[]) boolean execute(java.lang.String) boolean execute(java.lang.String, int) boolean execute(java.lang.String, int[]) boolean execute(java.lang.String, java.lang.String[]) void addBatch(java.lang.String)and int[] executeBatch() ……

```
import java.sql.Statement;
import java.sql.ResultSet;
import javax.servlet.http.HttpServletRequest;
public class Test
{
Statement statement = new Statement();
public void testMethod(HttpServletRequest request);
{
        StringBuffer sqlStatement = new StringBuffer("select * from employee where userid=");
```

```
        String id = request.getParameter("userid");
        if (id != null)
          sqlStatement.append(id);
        else
          sqlStatement.append("");
        ResultSet results =
statement.executeQuery(sqlStatement.toString());
      }
  }
```

Figure2. Abstract syntax tree according to source code

In addition, 100 thousand lines of source code are tested by the proposed algorithm. Results demonstrate that the proposed algorithm performs effectively, performing time is five second, recognition rate is ninety percent and misstatement rate is ten percent.

V. CONCLUSIONS AND FUTURE WORK

SQL injection attack has characteristics of simplicity of operator, high performance of concealment, serious hazard and hard to being detected. It has become most serious hazard to web application system. This paper presents a Java source-code SQL injection attack deletion algorithm. The proposed algorithm includes these steps such as lexical analysis of source codes, parsing of source codes, constructing abstract syntax tree of source code, defining rules, abstract syntax tree traversal, etc. Finally, Use experiments to evaluate the performance of the algorithm. Test results show the proposed algorithm performs perfectly. Nevertheless, because of various means of SQL injection attack and technological updating of SQL injection attack, hence language rules must be studied further so as to greatly raise recognition rate and reduce misstatement rate.

REFERENCES

[1] William G J, Viegas H J, Orso A. A Classification of SQL Injection Attacks and Countermeasures[C]//Proc. of International Symposium on Secure Software Engineering. Arlington, USA: IEEE Press, 2006.

[2] Su Zhendong, Wassermann G. The Essence of Command Injection Attacks in Web Applications[C]//Proc. of Annual Symposium on Principles of Programming Languages. Charleston, USA: [s. n.],2006.

[3] Stuttard D, Pinto M. The Web Application Hacker's Hand book: Discovering and Exploiting Security Flaws [M]. Beijing: People's Telecon Publishing House, 2009.

[4] Zhang zhuo, SQL injection attack techniques and countermeasures analysis [D].Shanghai Jiao tong university.2007, 50-51