Database Management System Seminar Assignment

By Lakshya Gupta
17BCS015

What is SQLIA?

SQL injection is known as a famous attacker application, that are capable to develop SQL based queries from input inserted by user. Attacker could get access of application database in case of successful attack. He could control it with help of SQL statements.

Types of SQL Injection Attacks:

1. Tautology Attack
   - SELECT * FROM users WHERE username=’blah’ or ‘a’=’a’ -- and password=’pass’

2. Piggy-Backed Queries
   - SELECT * FROM user_details WHERE userid = 'abcd' and password = ''; DROP TABLE xyz --

3. Union Query
   - SELECT * FROM user_details WHERE userid ='' UNION SELECT * FROM EMP_DETAILS -- ' and password = 'abcd'

4. Logically Incorrect Query
   - In this type of attack, the attacker sends an incorrect query and tries to gather information using the error messages it receives.

5. Inference
   - http://www.example.com/product.php?product_id=100 AND IF(version() like ‘5%’, sleep(15)

6. Stored Procedures
   - SELECT * FROM user_details WHERE userid = 'abcd' and password = ''; SHUTDOWN; -- '

Present SQL Detection Tools (Free):

1. SQLMAP: Open Source pen-testing tool that automates the process of detection and exploiting SQL injection flaws.

2. HAVIJ: It helps to detect SQL injection vulnerabilities on a webpage.

3. SQL Inject Me: it is a firefox add-on that helps to find SQL injection expolits on a website.

4. Backtrack5

SQL Detection Methods (In Web Applications):

1. Static Method : Also known as pre-generated detection of SQLIA. The developers follow certain guidelines and some commands like isString() , is used to detect if an SQLIA is heppening or not.

2. Dynamic Method Also known as post-generated detection of SQLIA. In this method, during run time a query is generated by the user , the program checks the input given by the user and then it allows the query to be passed if it is valid.

Proposed Techniques:

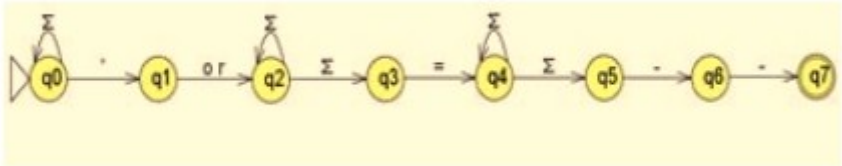
1. **Using Regular Expression And Finite Automata**

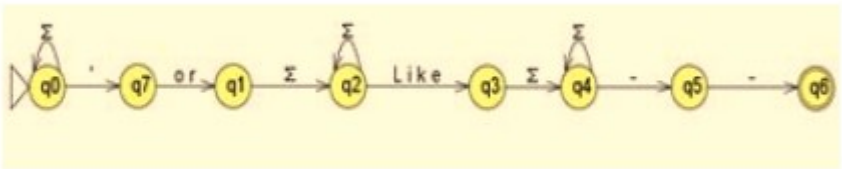
Problem:

Example	’ ’ب’ = ’ب’ –– it is always true
Place	Input box
Normal query	Select username, password from users where username = ’Mohammad’ and password =’P@ssw0rd’
Illegal reshaped query	Select username, password from users where username = ’Mohammad’ or ’ب’ = ’ب’ –– and password =’P@ssw0rd’
Description	The attacker uses tautology command in user input box by using Arabic character ’ب’ to login the system.

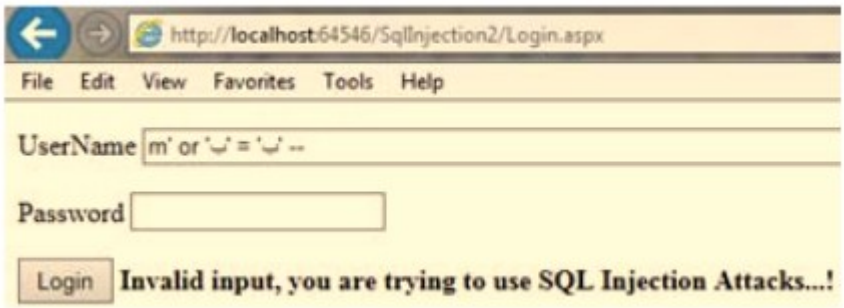
Example	http://www.site.org/index.aspx?id=888' or '공격' = '공격' -- it is always true
Place	URL
Normal query	Select ID, password from users where ID = 888
Illegal reshaped query	Select ID, password from users where ID = 888 or '공격' = '공격' --
Description	The attacker uses tautology command at the end of the URL by using Korean word '공격' which means "attack" in English to list all hidden ID's

Solved Using Regular Expression And Finite Automata

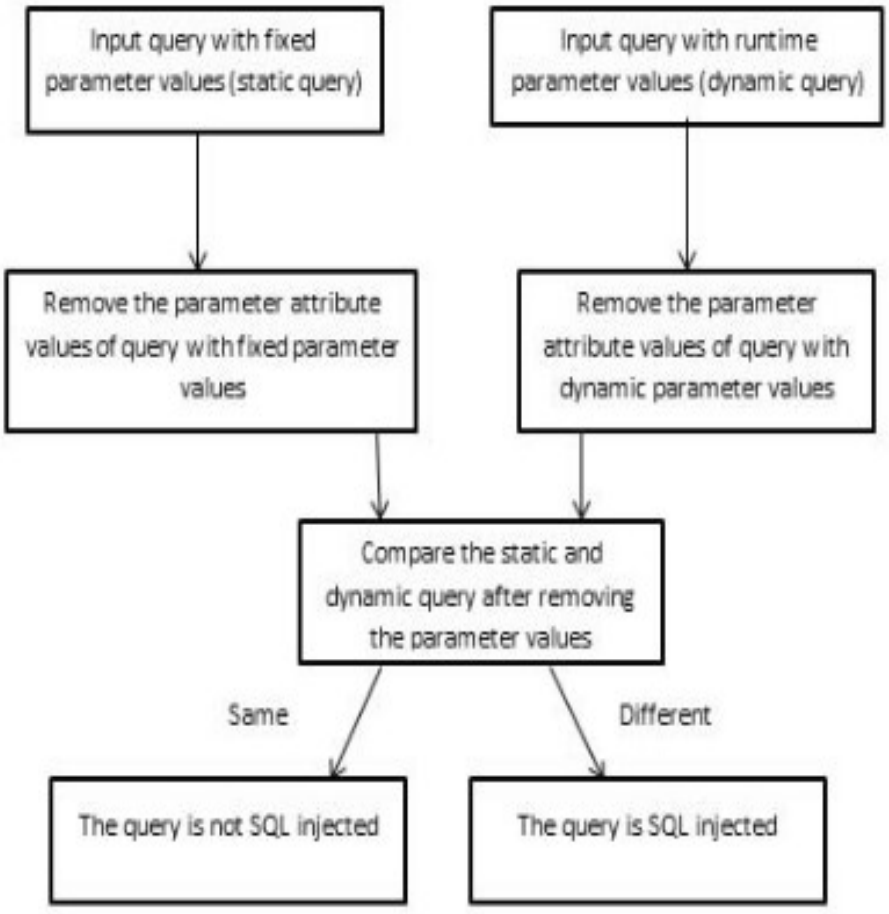




Implementation



2. By Removing The Parameters



Example:

Consider a query:

FXQ: SELECT * FROM STUDENT WHERE RNO='$rollno' AND NAME='$name'; ..... (1)

A function delete () is used to delete the parameter values [1]. The parameter values in fixed SQL queries i.e. static and the SQL queries generated at dynamic time are deleted.

By applying the function delete () to the query FXQ. The result is:

DFXQ=delete (FXQ) = SELECT * FROM STUDENT WHERE RNO= ' ' AND NAME= ' '; …. (2)

At the runtime the query can be in the form,

RTQ1= SELECT * FROM STUDENT WHERE RNO='1001' AND NAME='AJAY'; ….. (3)

By applying the function delete () to the query RTQ1. The result is:

DFXQ1= delete (RTQ1) = SELECT * FROM STUDENT WHERE RNO=' ' AND NAME=' '; …..(4)

Again a query at runtime could be as

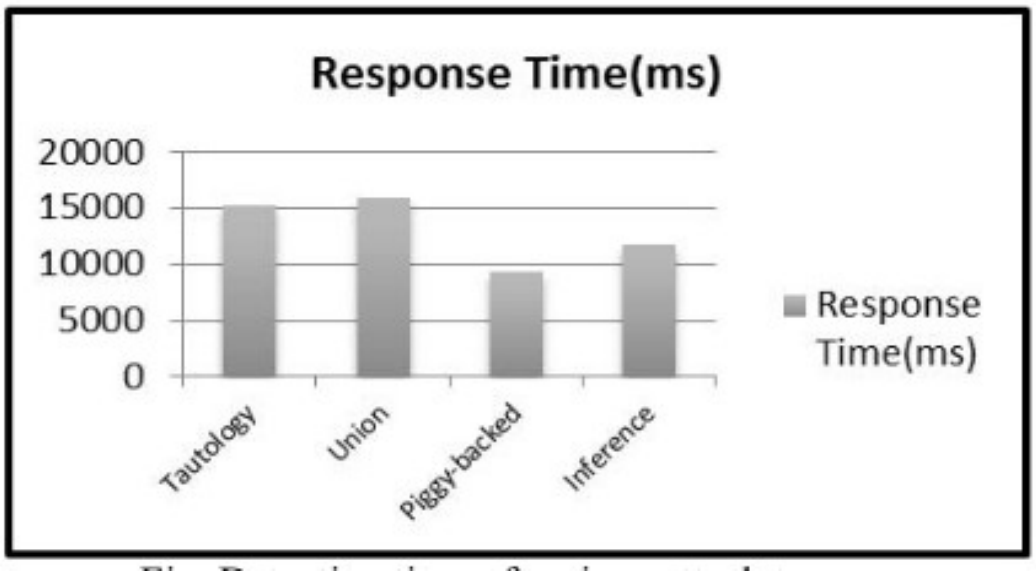
RTQ2= SELECT * FROM STUDENT WHERE RNO='1' or'1=1'—'AND NAME='AJAY'; …..(5)

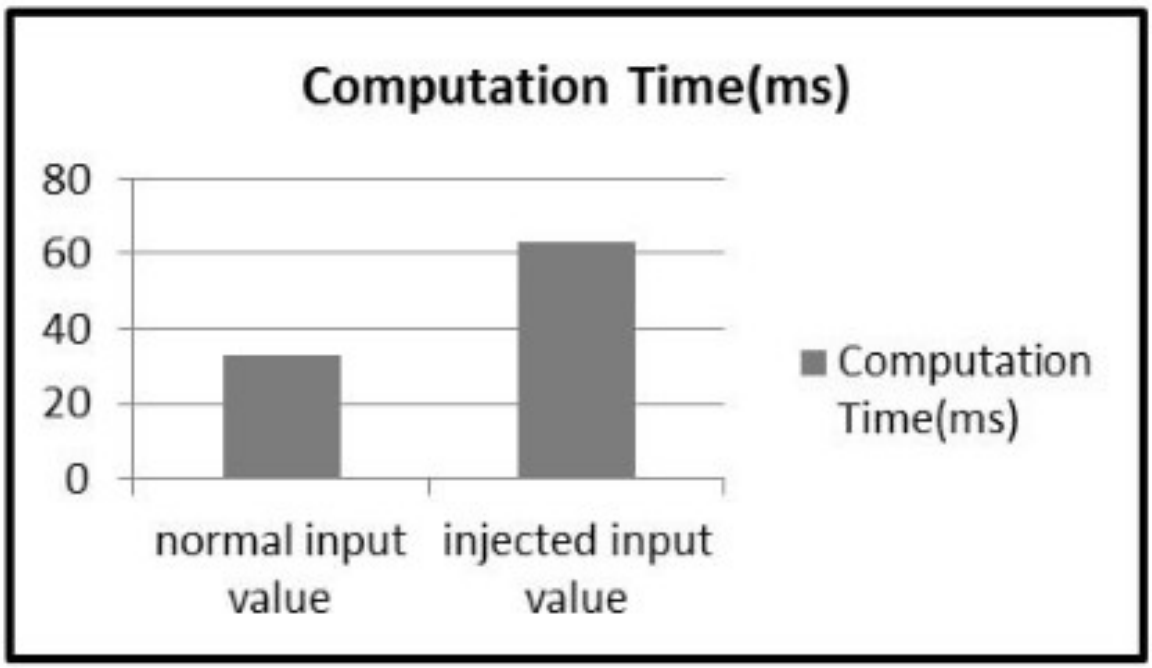
By applying the function delete () to the query RTQ2. The result is:

DFXQ2= delete (RTQ2) = SELECT * FROM STUDENT WHERE RNO=' ' or ' '—' 'AJAY'; ….(6)

When the static and dynamic query after removing the parameter values are compared, if both are same then the query is normal. If there is some difference then the query is not normal, i.e. code injection is present in the query.

Framework Analysis





Comparison with Previously Proposed Models

Detection/ Prevention Method	Source code adjustment	Static analysis	Applicable type of web application	Detected type of attacks
AMNESIA	Not needed	String analysis	Runtime monitoring	All but except stored procedures
Framework and database firewall method	Not needed	W3af	Database firewall	All
Proposed method	Not needed	String analysis	Query string with dynamic parameter.	All

Detection Analysis:

Web page	Attacks detected/ Malicious input	Attacks detected/ Normal input	Detection rate (%)
SQLIA page	40/45	75/75	96
Detect Page	45/45	75/75	100
Prevent page	45/45	75/75	100