Scientific Journal of Impact Factor(SJIF): 3.134
e-ISSN(O): 2348-4470
p-ISSN(P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 2, Issue 3, March -2015

WEB APPLICATION PROTECTION AGAINST SQL INJECTION ATTACK

Pranita Talekar[1], Rajshri Misal[2], Tanuja Nevase[3], Prof.Sanchica Bajpai[4]

[1,2,3,4]*Department of Computer Science. University of Pune MH (India), JSPM's BSIOTR. College of Engineering, Pune, Maharashtra, India*

Abstract— *SQL injection is one of the top threats to any web application which interacts with a database system. It is also one of the highly dangerous threats because it is easy to generate, difficult to design a defense mechanism and the data vulnerable to this type of attack is highly sensitive such as passwords, credit card details, etc. Injection attack is a method that can inject any kind of malicious string or anomaly string on the original string. The proposed algorithm shows that everything is well against the SQL Injection Attack. The Proposed a detection and prevention technique for data using Aho–Corasick pattern matching algorithm. This algorithm is classic algorithm. The results show that model protects against 100% of tested attacks before reaching the database layer.*

Keywords—*SQL Injection Attack; Pattern matching; Static Pattern; Dynamic Pattern Crafting, SQLIA, Vulnerabilities, Web Application Security, Cybercrime.*

1. INTRODUCTION

Web applications [1] are running over a network such as the internet or an intranet. The web application processes commands and verifies security access to the database through middleware like JDBC, SQLJ, or JDO API, ODBC. Web application enable website to become dynamic by making connections within the database. The high level system components of web applications are shown in figure 1.

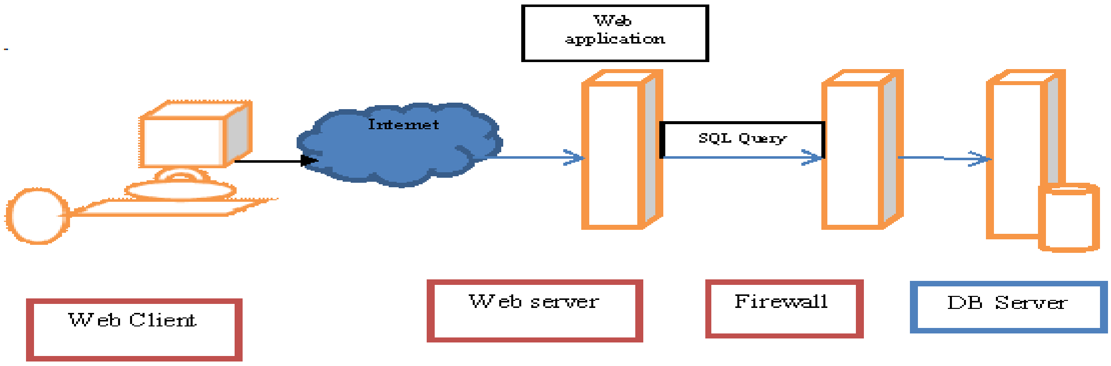


Fig 1: **Web Application Architecture**

In the web application architecture there are different types; browsers, networks, web servers, web applications and databases. Firstly, the client requests a page either a it can be static page or dynamic page after this, the web browser passes this request through the firewall to the web server[6]. Next, the web server handles this request based on an initial configuration like HTTP, HTTPS, etc.

This can also handles these requests by "decoding" the webpage. After, the web server passes this request to the web application server. Finally, the web application passes these requests to database using firewall to protect. Web applications uses queries statements to generate set of strings to interact with the database. Web applications are well known for bad quality of security vulnerabilities that can be victimized by writers of malware and hackers [8]. One of the main causes of SQL injection vulnerability is bad programming [9]. This vulnerability is exploited by attackers using different types of SQL injection attacks. The attackers usually inject malicious queries into HTTP requests [6]. Although there are infinite possible ways of framing an attack, they can be classified under seven types.

- Tautology Attack
- Piggy-Backed Queries
- Union Query
- Illegal/Logically Incorrect Queries
- Stored Procedures

- Alternate Encodings
- Inference

II.LITERATURE REVIEW

SQL injection attack has different types of properties, such as a useful under threat, vulnerabilities being exploited and attack techniques utilized by threat agents. The efficient method in preventing an SQL injection attack is Model-based guard constructor prevention. SQL Injection Attack (SQLIA) is a type of attack on web application which occurs when an attacker inputs malicious strings as parameters in legitimate SQL statement. Web application security generally focuses on identifying vulnerabilities and malicious strings within web applications layer. In 2006, Scientist Ke Wei et al. [2] suggest that by using SQL injection attacks, an attacker could thus obtain information. They also suggest that an attacker could even use a SQL injection as IP/Port scanner of the internal corporate network.

This technique combines static application code analysis with runtime validation. It use to eliminate the occurrence of such attacks. The deployment of this technique can be automated and used on a need-only basis.

In 2008, Authors Mehdi Kiani et al.[3] describe an anomaly based approach which utilizes the character distribution of certain sections of HTTP requests to detect previously unseen SQL injection attacks. Their approach requires no user interaction, and no modification, either the backend database or the source code of the web application itself. They also evaluate the effectiveness of their model.at different types of SQL injection attacks.

III.PROPOSED SCHEME

The proposed scheme has the following two phases,
1) Static Phase,
2) Dynamic Phase

In the Static Pattern List, maintain a list of known Anomaly Pattern (Query) In Static Phase, the user generated SQL Queries are checked by applying Static Pattern Matching Algorithm.

In Dynamic Phase, if any new anomaly is occur then Alarm will indicate and new Anomaly Pattern will be generated. The new anomaly pattern will be updated to the Static Pattern List. The following steps are performed during Static and Dynamic Phase,

1 Static Phase

Following are the steps in Static phase
Step 1: In this User generated SQL Query is send to the proposed Static Pattern Matching Algorithm
Step 2: This Algorithm is given in Pseudo Code is below
Step 3: When the user gives any Query the pattern matching algorithm compare that query with stored static pattern list.
Step 4: If the query pattern is match with one of the stored pattern in the Pattern List then the Query is affected with SQL Injection Attack.

2 Dynamic Phase

Following are the steps in Dynamic phase
Step 1: In Dynamic phase, Anomaly Score value is calculated for the user generated SQL Query.
Step 2: if the Anomaly Score value is more,
Then the Threshold value, then a Alarm is given and Query will be pass to the Administrator.
Step 3: When Administrator receives any type of Alarm then the Query will be analyze by manually. If the query is affected by any type of attack then a pattern will be generated and the pattern will be added to the Static Pattern list.

In this We consider Thresholds value 30%.beacuase depend on threshold value attacker can match user data(Username Or password).So the depend on given threshold value attacker can match only less character. Maximum 2 or 3 character.

IV. ALGORITHMS

1. Pattern Matching Algorithm

In this system architecture, Static Pattern Matching Algorithm is the main part and the pseudo code for the Static Pattern Matching Algorithm is given below,

A. Static Pattern Matching Algorithm

```
1) Input        query      <- user query
     Pattern list []       <- patterns from database

2) Loop: for (String pattern from pattern list)

2.1) if pattern == query
                       Reject query and display login page with error.
                       Continue loop;
                     End if

2.2) anomaly score = (match (pattern, query) * 100)/length (pattern).

2.3) if anomaly score >= THRESHOLD
                      Return alarm <- Administrator

2.4) else
                      Return query <- accepted
                   End if

        3) End for.
```

B. Keyword Matching Algorithm (match (pattern, query))

```
1) Input        query      <- user query
                Pattern    <- pattern from database


2) char [] query array = query.toCharArray ();
             char [] pattern array = pattern.toCharArray ();
             Int match=0;
            int total = query_array.length


3) for(Character ch_query from query array)
               3.1) if ch_query == ch_pattern
                         Match++;
                     End if

4) Match = match/total;


 5) Output match.
```

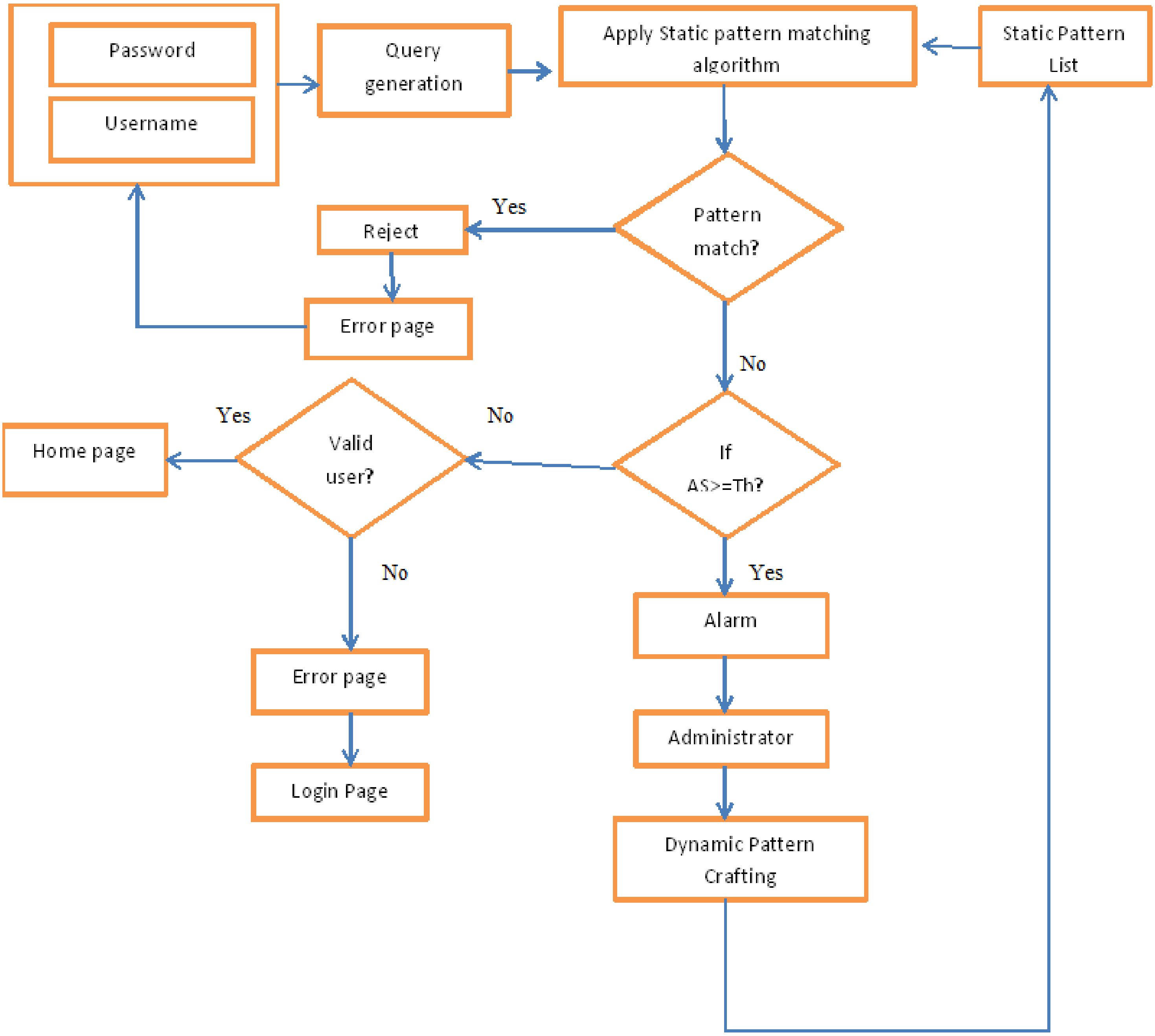
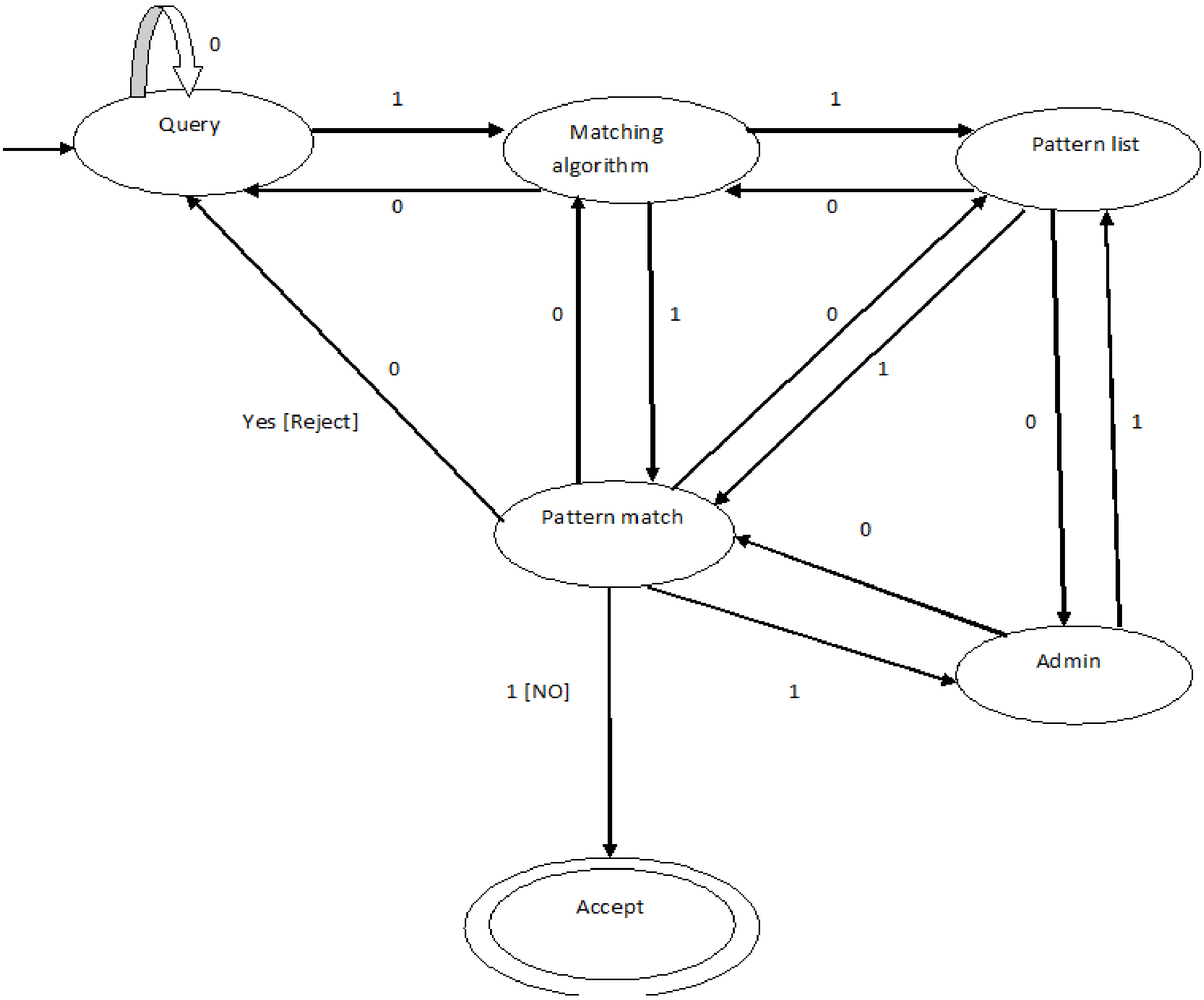


Fig 2: **.System Architecture**

2. MATHETICAL MODEL



V. PROJECT IMPLEMENTATION

In web based security problems, SQLIA has the top most priority. SQL Injection is a method of exploiting the database of web application. It is done by injecting the SQL statements as an input string to gain an unauthorized Access to a database. In this project we implement code in java to how protect personal information from hackers. For that use net beans 7.2 versions as front end. And Heidi SQL as back end. And GUI for representation. In this project if user gives legal query then all information will display. Otherwise it shows invalid username and password.in this Static phase and Dynamic phase are generated as shown below. And in Dynamic phase Alarm will generated if user give unknown query.

1. NETBEANS 7.2

Net Beans use mostly use in this world, for project work. The Net Beans IDE is an integrated development environment [12]. It is available for Windows, Mac, and Linux. The Net Beans project consists of an open-source IDE and also application platform that enable developers to create web, enterprise, desktop, and mobile applications using the Java platform, as well as PHP, JavaScript, Ajax, and C/C++. The Net Beans project is supported by a vibrant developer community and offers extensive documentation and training resources as well as a diverse selection of third-party plugins. Net Beans IDE 7.2 provides a significantly improved performance and coding experience, with new static code analysis. Net Beans IDE 7.2 is available in English, Brazilian Portuguese, Japanese, Russian, and Simplified Chinese. It also Support for PHP 5.4[12].

2 Heidi SQL

Heidi SQL, previously known as MySQL-Front[11]. It is a free and open source client, or frontend for MySQL. its forks like Maria DB and Percona Server, Microsoft SQL Server and Posture SQL, developed by German programmer Ansgar Becker and a few other contributors in Delphi. To manage databases with Heidi SQL, users must login to a local or remote MySQL server with acceptable credentials, creating a session. Within this session users may manage MySQL Databases within the connected MySQL server, disconnecting from the server when done. Its feature is sufficient for most common and advanced database, table and data record operations but remains in active development to move towards the full functionality expected in a MySQL Frontend[11]. In this there is different types of operations like insert, delete, update and more..

3 GUI

A **GRAPHICAL USER INTERFACE** (**GUI**, sometimes pronounced "gooey" or "gee-you-eye") is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation. GUIs were introduced in reaction to the perceived steep learning curve of command-line interfaces (CLIs), which require commands to be typed on the keyboard [10]. The actions in a GUI are usually performed through direct manipulation of the graphical elements. In addition to computers, GUIs can be found in hand-held devices such as MP3 players, portable media players, gaming devices and smaller household, office and industry equipment. The term "GUI" tends not to be applied to other low-resolution types of interfaces with display resolutions, such as video games (where HUD is preferred), or not restricted to flat screens, like volumetric displays because the term is restricted to the scope of two-dimensional display screens able to describe generic information, in the tradition of the computer science research at the PARC (Palo Alto Research Center).[10]

VI. RESULT AND DISCUSSION

- **Test case with empty username and password:** This is most base positive test case, if hacker or user do not enter any password or username then it show that please fill the empty block.
- **Test case of username and password with Sql Query:** If hacker gives any SQL query that will be related to static pattern list Query then it will show SQL INJECTION ATTACK DETECTED USING STATIC PHASE.
-
- **Test case with valid username and password:** If user gives valid username and password then he/she can access information.
- **Test case with different SQL Query as Username and password:** If hacker gives any SQL query that will be related to static pattern list Query then it will show SQL INJECTION ATTACK DETECTED USING DYNAMIC PHASE.
- **Verify the correct error messages like** incorrect combination of user name and password. If you are getting anything like Incorrect username or Incorrect password then be conscious because you application is giving half the information to hacker and your application is in great danger.

USER LOGIN	RESULT
Username: Empty Password: Empty	Please fill out this field
Username: admin@gmail.com Password:adm	Data will accept. Personal Information will be display.
Username: 1' or '1' = '1 Password: 1' or '1' = '1	Sql injection attack detected using static phase**.**
Username: 0'OR'1=2 Password: 0'OR'1=2	Sql injection attack detected using dynamic phase**.** And Alarm will generate.
Username: pranjit@gmail.com Password: pranitan	Invalid username or password...!!

Table 2: Result after the Execution

VII. CONCLUSION

This work for detection and prevention of SQL Injection Attack using Aho–Corasick pattern matching algorithm. The work is evaluated by using well known attack patterns. Most of the web applications uses intermediate layer to accept a request from the user and retrieve information from the database. Most of the time they use scripting language to build intermediate layer. Future evaluation work direction on evaluating the technique precision, stability, flexibility and effectiveness in practice to show strength and weakness of the techniques. The SQL injection attack also stopped using approach such as Blacklist malicious hosts, Minimize admin-level access to a database, Normalize inputs, Model Based Hybrid Approach, SVM.

VII. REFERENCES

[1]. D. Parsons, Dynamic Web Application Development Using XML and Java: Cengage Learning EMEA, 2008.

[2]. Ke Wei; Muthuprasanna, M.; Kothari, S., "Preventing SQL injection attacks in stored procedures," Software Engineering Conference, 2006. Australian, vol., no., pp.8 pp.,, 18-21 April 2006.

[3]. Kiani, M.; Clark, A.; Mohay, G., "Evaluation of Anomaly Based Character Distribution Models in the Detection of SQL Injection Attacks," Availability, Reliability and Security, 2008. ARES 08. Third International Conference on, vol., no., pp.47,55, 4-7 March 2008.

[4]. Pinzón, C.; De Paz, J.F.; Bajo, J.; Herrero, A.; Corchado, E., "AIIDA-SQL: An Adaptive Intelligent Intrusion Detector Agent for detecting SQL Injection attacks," Hybrid Intelligent Systems (HIS), 2010 10th International Conference on , vol., no., pp.73,78, 23-25 Aug. 2010.

[5]. Elia, I.A.; Fonseca, J.; Vieira, M., "Comparing SQL Injection Detection Tools Using Attack Injection: An Experimental Study," Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on , vol., no., pp.289,298, 1-4 Nov. 2010.

[6]. Kai-Xiang Zhang; Chia-Jun Lin; Shih-Jen Chen; Yanling Hwang; Hao-Lun Huang; Fu-Hau Hsu, "Trans SQL: A Translation and Validation- Based Solution for SQL-injection Attacks," Robot, Vision and Signal Processing (RVSP), 2011 First International Conference on , vol., no., pp.248,251, 21-23 Nov. 2011.

[7] Hal fond, W. G. and Orso, A , “AMNESIA: Analysis and Monitoring for Neutralizing SQL-Injection Attacks”, in Proceedings of the 20th IEEE/ACM international Conference on Automated Software Engineering, 2005.

[8] C.J. Ezeife, J. Dong, A.K. Aggarwal, “Sensor WebIDS: A Web Mining Intrusion Detection System”, International Journal of Web Information Systems, volume 4, pp. 97-120, 2007.

[9] Ramya Dharam, Sajjan. G. Shiva; "Runtim Monitors to Detect and Prevent Union Query Based SQL Injection Attacks" in *proceedings of the 10th International Conference on Information Technology: New Generations (ITNG)*, Las Vegas, NV: IEEE (2013). pp: 357-362. DOI: 10.1109/ITNG.2013.57.

[10] Definition of GUI at Dictionary.com". Retrieved January 2010.

[11]"ZeosLib - Delphi database components for MySQL, PostgreSQL, Interbase, Firebird, MS SQL, Sybase, Oracle and SQLite". Sourceforge. 2014-10-21. Retrieved 2014-12-01.

[12].”Net Beans Release Information From Net beans 7.2.

.