iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 3 **Issue:** XII **Month of publication:** December 2015

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

SQL Injection Attack and Method for Detection and Prevention in ASP.NET Web Applications

Rahul[1], Sharad Chauhan[2], Kamal Sharma[3]
[1]*M.Tech Student, Department of CSE, E-Max Group of Institutions, Ambala, Kurukshetra University, India*
[2]*Assistant Professor, Department of CSE, E-Max Group of Institutions, Ambala, Kurukshetra University, India*
[3]*Professor, Department of CSE, E-Max Group of Institutions, Ambala, Kurukshetra University, India*

Abstract-- In This paper, we propose a technique, which uses runtime validation to detect the occurrence of such attacks, which evaluation methodology is general and adaptable to any existing system. There is a need to protect databases. In this a IDPS system is developed to analyze the values submitted by users through HTML forms and look for possible attack patterns. Once the system finds such a pattern, it blocks the attack and makes a record of the activity. If an attacker continues to pass such attack patterns, the system blocks access by this user altogether. This provides a combination of pattern-based detection and anomaly-based detection to create a reasonably robust intrusion detection system(IDS), with respect to SQL-I attacks.

Keywords-- HTML, SQL injection, Attacks, IDPS.

I. INTRODUCTION

Security in web application is always a big headache for the programmers but providing secure environments is one of the key principles in the process of gaining customer confidence . In this era of web applications, almost all websites are dynamic, i.e., all the websites are database driven and large data will be accepted from user.

SQL injection is a technique that exploits a security vulnerability occurring in the database layer of an application. The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. It is in fact an instance of a more general class of vulnerabilities that can occur whenever one programming or scripting language is embedded inside another.

SQL Injection Attacks (SQLIA's) are one of the most severe threats to web security. They are frequently employed by malicious users for a variety of reasons like theft of confidential data, website defacement, sabotage etc. The number of SQLIA's reported in past years has been showing a steadily increasing trend and so is the scale of the attacks. It is, therefore, importance to prevent such types of attacks, and SQLIA prevention has become one of the most active topics of research in the industry and academia.

SQL injection refers to a class of code-injection attacks in which data provided by the user is included in the SQL query in such a way that part of the user's input is treated as SQL code. It is a trick to inject SQL query or command as an input possibly via the web pages. They occur when data provided by user is not properly validated and is included directly in a SQL query. By leveraging these vulnerabilities, an attacker can submit SQL commands directly access to the database.

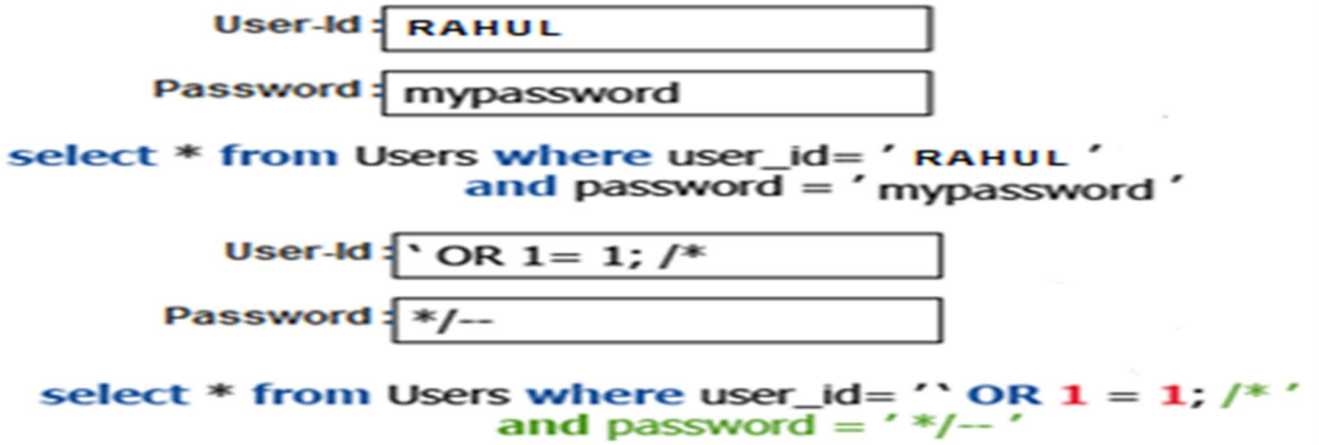


Figure 1: SQL Injection

II. RELATED WORK (IDPS) DESIGN

The system discussed is called the SQL Intrusion Detection and Prevention System (IDPS). The particular system discussed here is an extension of a particular system that protects a web application system from CGI attacks. However, the original system did not guard against SQL Injection attacks directed at databases connected to the system.

A. SQL-I Attack Detection

The IDPS system uses both signature-based and anomaly-based detection models to identify threats and attacks on the system. Signatures are carefully selected to implement the signature-based model. Anomaly-based detection is based on the number of times a user attempted to access the system, regardless of whether any SQL-I patterns have been detected.

IDPS is case-insensitive while trying to use the signature-based detection method to detect SQL-I attacks. The IDPS deals with White Space Manipulation attack by removing any white space before comparing text with known SQL-I attack patterns. The IDPS deals with Comments attack by looking for comment characters in the submitted text. The String Concatenation attack is dealt with by looking for the concatenation operation characters or CONCAT function. The keyword "UNION" is searched for by the IDPS in case an attack tries to perform the UNION Injection attack. The system will also look for binary, hexadecimal, and decimal characters in the submitted text to catch instances of this SQL-I attack variation. Sample patterns may be found in the SQLI_PATTERNS table described in the IDPS database schema. Even when no SQL-I attack pattern is detected in the submitted form text, the IDPS monitors the frequency of the login attempts to implement the anomaly-based detection method. When the number of visits has exceeded a predetermined threshold, the system automatically blocks the visitor for a time. It is significant to note that the screen the user sees when he or she has entered an incorrect password or when an SQL-I pattern is detected in the text matches. No feedback is communicated to the user as to whether or not the system has detected an attack to ensure the system limits unnecessary information broadcast**.**

B. Storing and Blocking Attacks

Even when a text is deemed to not contain an SQL-I attack signature, the access attempt along with user IP address and username and password issued is logged into the VISITORS table. This table may be reviewed by the system administrator or an analyst at a later time for possible attacks or new attack patterns. Normal users usually have a more predictable number of login attempts, and this number may be determined by observing everyday activity of these users. In their attempt to determine the weaknesses of the system, attackers will attempt to access the system significantly more often. If a user attempts to access to the system more than an arbitrary number over a predetermined amount of time, which is stored in a value in the CONFIG table, this behavior is suspicious and hence the user is a potential threat. Thus, this user should be automatically blocked for a certain amount of time. The user may attempt to access the system after this period of time has elapsed. This block is recorded in the HACKERS table with the reason "LIMIT_EXCEED" to serve as a permanent record. If an SQL-I attack is detected, the user's IP address, the text that matched the SQL-I pattern, the browser the user was using, and the time of the login attempt are all recorded in the HACKERS table as a permanent record. The attempt may be further reported by the IDPS by the use of an e-mail notification.

C. E-mail Notifications

The system administrator has the option to set up an e-mail address to received e-mail alerts. The system administrator may adjust the exact e-mail address, the frequency of the e-mails, and the type of activity he wishes to be notified of in the administration panel.

III. IDPS COMPONENTS AND INTERFACE

The system designed and implemented in this project is a complete IDPS which uses a combination of the signature-based detection model and the anomaly-based detection model. To detect attacks, the IDPS contains a list of signatures. The system contains the following crucial components to provide the complete web server security against the SQL-I attacks.

A. SQL-I Attacks Console Admin

The interface named SQL-I attacks lists all past attacks detected by the IDPS with the most recent ones listed first. This interface contains a table of attacks which contains the following attribute

1) Attack ID (iterative)
2) Attacker's IP address linked to 'Attacker History' interface
3) The login name or password issued to perform the attack
4) Browser that was used in the attack
5) Timestamp of when the attack was detected

Clicking on an IP address entries takes you to the 'Attacker History' interface page displaying the attack history of the clicked IP address. At the top of the page, there are links to the 'Blocked Attackers' page and MyAdmin interface. Figure 2 contain the screenshot of Attacks interface.

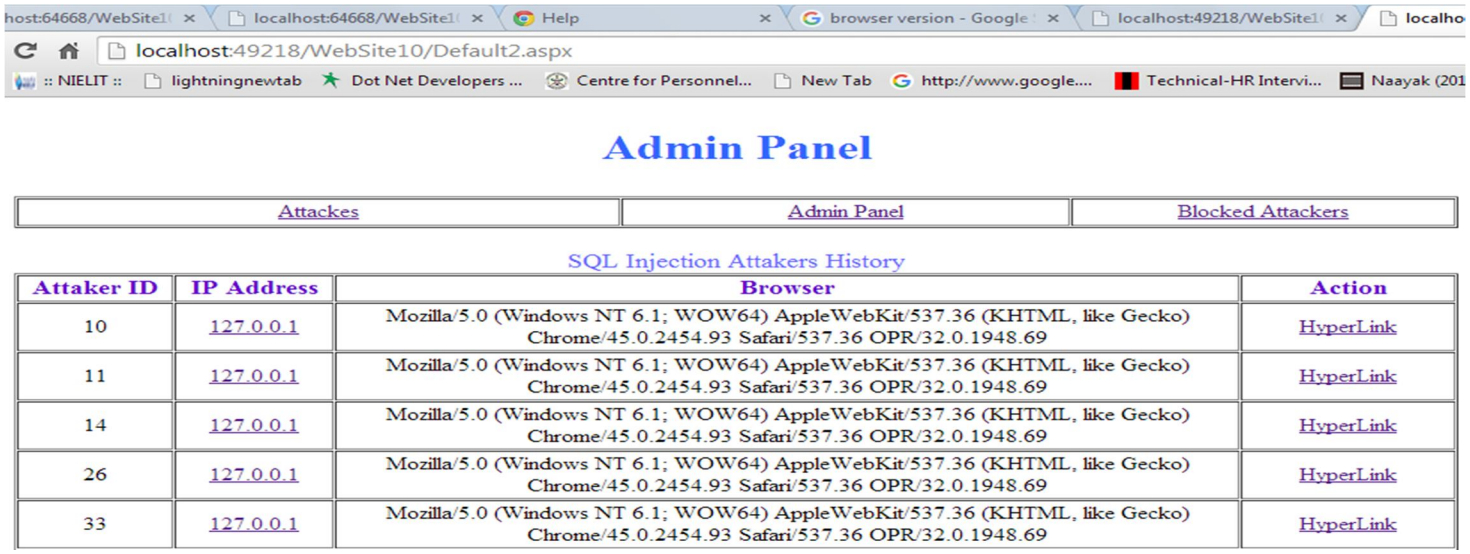


Figure 2 IDPS SQL-I Attacks (Admin) Console Interface

B. Attacker History Interface

Clicking the IP address of the attack from the Attackers page brings up the 'Attacker History' Interface page. The results on this page contain the attack history of the attacker. Information contained on this page include:

1) Attacker's IP address
2) Total number of attacks that were detected
3) 'Block IP' button that blocks the IP address from accessing the server Figure 4 illustrates this interface.

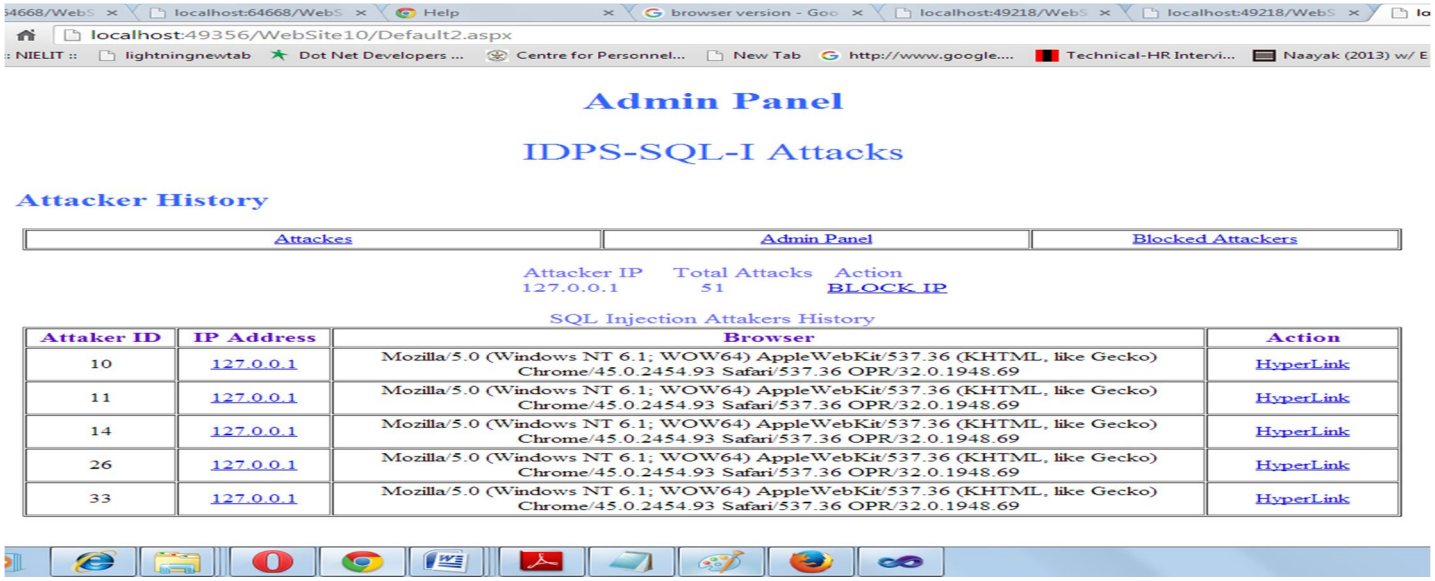


Figure 3 IDPS Attacker History Interface

Clicking on the Block IP button on the 'Attacker History' interface page blocks the IP address from accessing the system and then immediately displays the 'Blocked Attackers' interface page to the administrator.

C Blocked Attackers Interface

If the system administrator blocked an IP from the Attacker History page, or if he clicked the 'Blocked Attackers' link at the top of the page of most interfaces, then he will see the 'Blocked Attackers' interface page.

The following are the attributes that are displayed on this page:

1) The IP address blocked

2) The option to unblock this IP address Figure 4 illustrates this interface.

Figure 4 IDPS Blocked Attackers Interface

D. IP Address White List Management Interface

It is possible the system administrator wishes to designate IP addresses that he never wishes to be blocked by the IDPS system. These may be IP addresses from trusted locations, organizations, or individuals. Click on the White List Management link at the top. In the subsequent page, type in the IP address in the text box, then click on the 'Add to White List' button to add the address to the white list. A message then appears to confirm that the IP address was added to the white list.

E. E-mail Generation

Every time an SQL-I attack incident is detected, an e-mail is generated and sent to the administrator to ensure that the administrator is always kept informed about the attacks. The administrator can then choose to log into the system to perform further investigation into the suspicious activity and block the user. The e-mail contains the following information in the message

1) The IP address of the attacking machine.

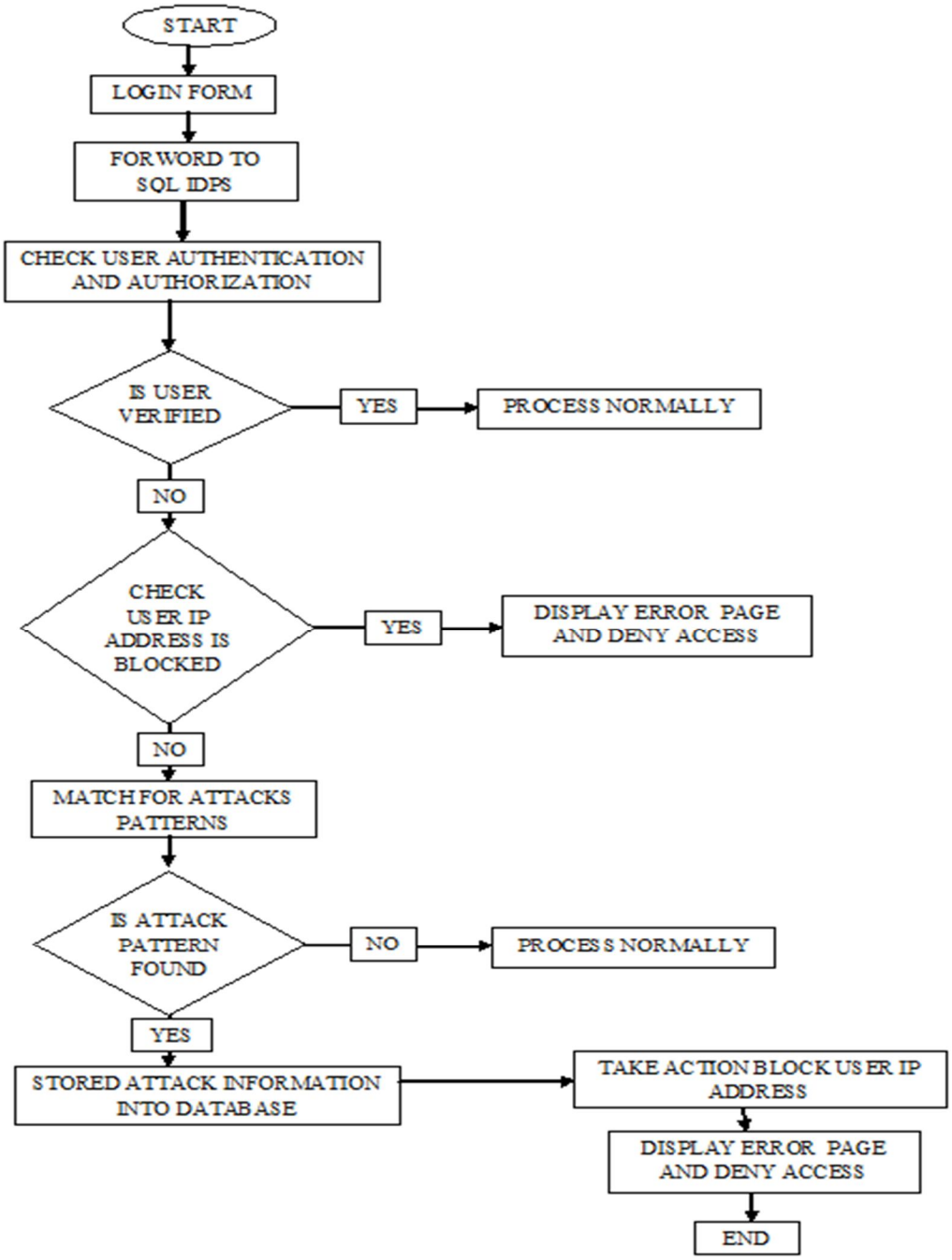
2) The text used in the SQL-I attack.

3) The web browser used to perform the attack.

IV. IDPS REQUEST HANDLING FLOW CHART

When the form data is submitted to the web server, the data is forwarded to the SQL-I Intrusion Detection and Prevention System. The login attempt information —such as visitor IP address, time of login, and login username and password is stored in the IDPS database for future reference. The IDPS checks to see if the visitor is on the Trusted. If so, then the username and password is passed through the sql_string function to clean up the escape characters, and then processed normally. Otherwise, the IDPS checks to see if the visitor is marked Blocked If the visitor is, an error page is returned and the visitor is denied access to the system.

IDPS Request Handling Flow Chart



The IDPS will continue to check if the client attempting to log in has exceeded the threshold allowed per the time period on the

system. If yes, then an error page is returned and the visitor is denied access temporarily.

Next, a scan of the submitted form text for SQL-I attack patterns is performed. If no attack pattern is found, then it is filtered using sql_string and processed normally. If an attack pattern was found, then the attacker's information is recorded into the database. If the occurrences of an attack are less than a threshold, then filter and process the data normally. However, if it is greater than the threshold, then the attacker information is stored in the database, the attacker is blocked indefinitely, an e-mail is sent to the system administrator, and an error page is displayed. In flow chart that show how requests are handled by the IDPS, and how it intercepts and handles an SQL-I attack event.

V. SAMPLE TEST PLAN AND RESULTS

The test plan consists of test cases, each of which contains a different attacks input for the IDPS system, and represents a particular type of SQL-I attack pattern that may be used to attack the system. Each of these will be issued against the system. Then, the output and results will be recorded in the system to be analyzed and determine how effective the IDPS is. A brief of the test cases and results are outlined in table1 below.

Table 1 Sample Test Plan and Results

ATTACK TYPE	USERNAME INPUT	PASSWORD INPUT	RESULT
AND/OR	Admin	' OR '1'='1	Blocked
AND/OR	Admin	' OR '0'<>'1	Blocked
White Space Manipulation Variation of AND/OR	Admin	'OR'1'='1	Blocked
White Space Manipulation Variation of AND/OR	Admin	'OR'0'<>'1	Blocked
Comments	admin'--	Xyz	Blocked
Comments	admin'#	Xyz	Blocked
Comments	' or 1=1--	Xyz	Blocked
Comments	' or 1=1#	Xyz	Blocked
Comments	admin'/*	Xyz	Blocked
Comments	' or 1=1/*	Xyz	Blocked
Comments	') or '1'='1--	Xyz	Blocked
Comments	') or ('1'='1--	Xyz	Blocked
Comments	admin'--	' OR '1'='1	Blocked
String Concatenation	' UNI' + 'ON' + ' select "test" from dummy'	Xyz	Blocked
UNION	' UNIVI. ON select all from dummy --'	Xyz	Blocked

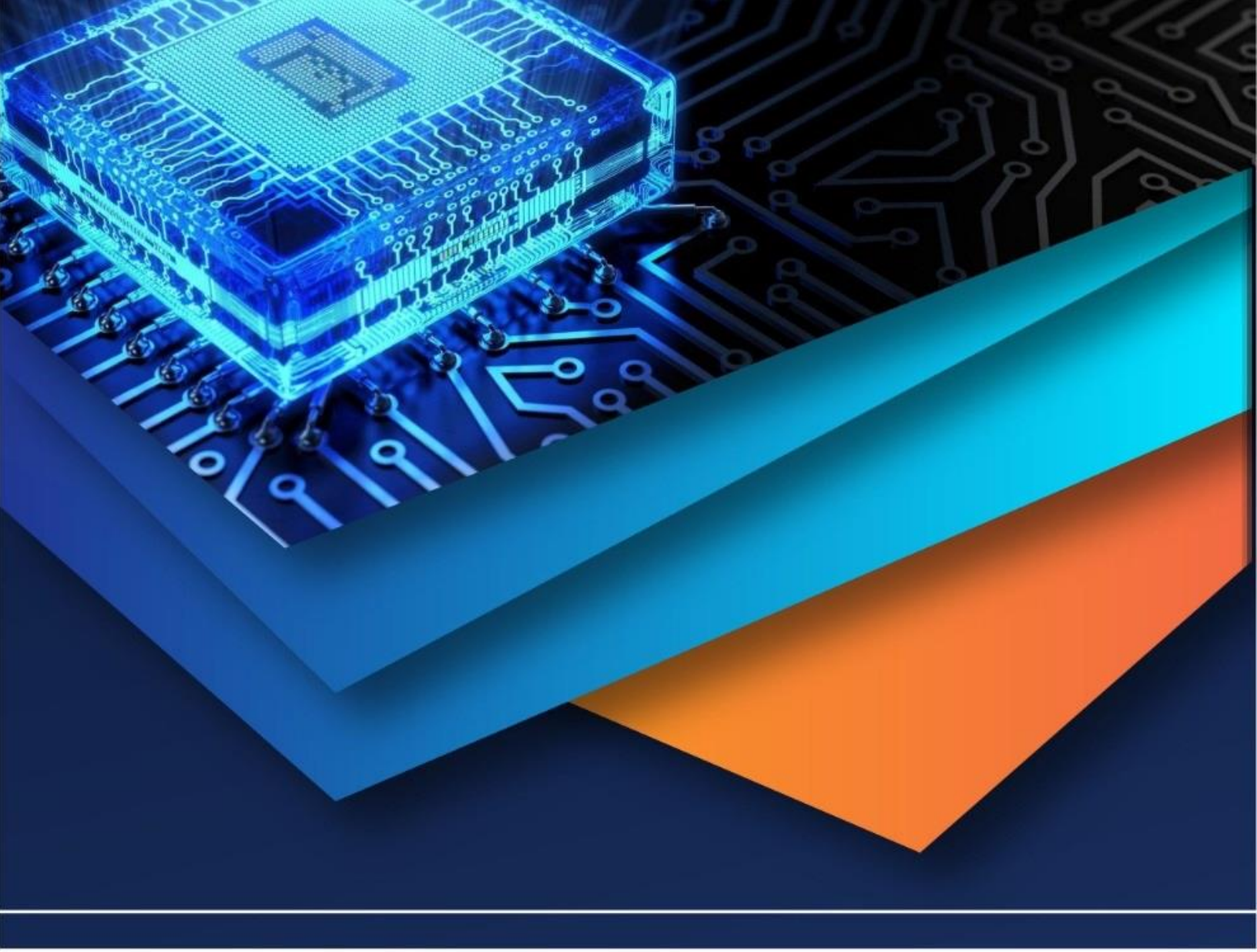
VI. CONCLUSION FROM TEST PLAN RESULTS

The range of attacks patterns were tested against the system according to the test plan above result test table is made. When attacker sending an attack pattern to the server, the web server is able to successfully prevent the attacks, log the attacks were entry in the

database, and report the administrator of the attack through e-mail and further attacks are not possible because the attacker's IP address is blocked.

REFERENCES

[1] W. Halfond, J. Viegas, and A. Orso. A Classification of SQL-Injection Attacks and Countermeasures. Proceedings of the IEEE.
[2] http://www.w3resource.com/sql/sql-injection/sql-injection.php
[3] Sincy George, Member, IEEE, and Vivek Agarwal, Senior Member, IEEE"Optimum Control of Selective and Total Harmonic Distortion in Current and Voltage Under Nonsinusoidal Conditions" IEEE TRANSACTIONS ON POWER DELIVERY, VOL. 23, NO. 2, APRIL 2008.
[4]http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5615711&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D5615711.
[5] P. Finnigan. SQL Injection and Oracle - Parts 1 & 2. Technical Report, Security Focus, November 2002. http://securityfocus.com/infocus/1644.
[6]T.O.Foundation. Top Ten Most Critical Web Application Vulnerabilities, 2005. http://www.owasp.org/documentation/topten.html.
[7]C. Gould, Z. Su, and P. Devanbu. JDBC Checker: A Static Analysis Tool for SQL/JDBC Applications. In Proceedings of the 26th International Conference on Software Engineering (ICSE 04) –Formal Demos, pages 697–698, 2004.
[8] E. M. Fayo. Advanced SQL Injection in Oracle Databases. Technical report, Argeniss Information Security, Black Hat Briefings, Black Hat USA, 2005.
[9] N. W. Group. RFC 2616 – Hypertext Transfer Protocol – HTTP/1.1. Request for comments, The Internet Society 1999.
[10] W. G. Halfond and A. Orso. AMNESIA: Analysis and Monitoring for NEutralizing SQL-Injection Attacks. In Proceedings of the IEEE and ACM International Conference on Automated Software Engineering (ASE 2005), Long Beach, CA, USA, Nov 2005. To appear.
[11] W. Halfond and A. Orso. AMNESIA: Analysis and Monitoring for NEutralizing SQL-Injection Attacks. In Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering (ASE), pages 174–183, 2005.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY