International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 5, Issue 11, November 2015)

Detection and Automatic Prevention against SQL Injection Attacks and XSS Attacks perform on Web Applications

Chaitali Khairnar[1]

[1]M.tech Scholar, Department of Computer Science & Engineering, Maharashtra Institute of Technology (MIT), Aurangabad, Maharashtra, India

***Abstract*— SQL Injection Attacks (SQLIA) and Cross-site scripting (XSS) Attacks are one of the most harmful threats to the security of database driven applications. Most of the Web Application have critical insect, affecting their security, which makes them vulnerable to attacks by hackers and malicious users. This paper proposes a method name as Reverse Proxy Server Approach, in which all of the input requests are redirected to the Proxy Server. Query detector and sanitizing application are placed in Proxy Server which helps in preventing SQL Injection (SQLI) and Cross Site Scripting Attacks (XSS). In This paper our focus is on detection and prevention of all types of SQL Injection and cross-site scripting attacks by providing strong security majors. This paper also shows comparison of Reverse Proxy Server with all other methods describe earlier.**

***Keywords*— Attack, *Detection*, Prevention, Reverse Proxy Server, Tools.**

I. INTRODUCTION

In modern day to day life the Internet and a web application are becoming very important for daily activities. Domain like private sectors, Government Sectors and an every individual entity require the Internet and a Web Application for accomplishing task or to perform their work. Use of Net banking, Online Transaction, Paying electricity bills, Online shopping all those things are becoming much simpler or done in few clicks with the use of a web application. The Internet and Web Applications has lots of benefits and features but on the other hand side they have a dangerous disadvantages too as we never think of. Now a day's a Web Applications are surviving from various attacks, most harmful attack is Code Injection Attack. Code Injection Attack is broadly classified SQL Injection Attack (SQLIA) and Cross Site Scripting Attack (XSS).There are 4 Broad Categories which makes a Web Application Vulnerable.1)Poorly written code in it 2) Configuration mistakes 3) Poor design of system 4) Lack of input validation. These vulnerable applications are targeted by attacks or malicious users to perform their illegal activities.

A. Definition of SQLIA

SQLIA use structure query language as an input to Web Application. Attacker manipulate valid SQL query by changing syntax, semantics or logic by adding certain SQL keywords or operator. Attacker can shape their illegitimate input as a part of final query string which flows from front end of system to backend databases, And an attacker gain illegal/ authorized access of a database. After gaining unlimited access over the database an attacker perform activities like shipping of goods without charge, retrieving password of users present in database, credentials and confidential data has been leaked, deleting tables in database, deleting entire database, changing someone's Valuable information. SQL Injection Attacks are performed sequentially or together.

B. Types of SQLIA

Tautologies

The main purpose behind this attack is by pass authentication by providing malicious input to web application. Attacker inject malicious input in input field, if that particular Web application is vulnerable to Tautological attack ,an Attacker gain access of any individuals account. Fig 1 shows Password field contains malicious input ' ' OR 1=1 which authenticate the user without checking password, because whole generated query returns true.

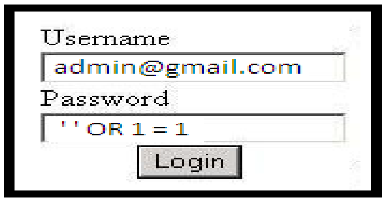


FIGURE I: Tautology Attack

Illegal/Logically Incorrect Queries

The main purpose behind this attack is to know the details of database or to gain some knowledge about database.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 5, Issue 11, November 2015)

For an every wrong input, an error message is generated from Backend server including certain debugging information also it sometimes provides information about column names fields exist in database. This error messages plays important role in attack success. With the help of those messages attacker perform attack.

Union Query

The main purpose behind this attack is not to bypass authentication but also retrieving data from database using keyword UNION. An Attacker use UNION keyword between two or more SQL queries to carry out an attack. A valid SQL query and a malicious query join using UNION keyword.

Piggy-backed Queries

The main purpose behind this attack is to make a use of a query delimiter such as ";" .With the help of query delimiter, attacker execute multiple queries consequently. Normally first query and is a valid SQL query and second query is a malicious query input.

Stored Procedure

The main Purpose behind this attack, make a use of stored procedures provided by developer at a time of development. Attacker use stored procedure for an attack because; stored procedures are as vulnerable as Web application code.

Blind Injection

In this type of attack, attacker faces generic pages instead of error messages. In this scenario SQLIA would be more difficult but not an impossible.

Timing Attack

The main purpose of this attack is to observe the time delays. WAITFOR keyword is use is a response with time delays.

Alternate Encoding

The main purpose behind this attack is to use encoding technique to perform attacks. Different encoding scheme such as Hexadecimal, ASCII and Unicode are to be used.

C. Definition of XSS

Cross-site scripting is also called as XSS Attack, is very harmful type of a Code injection attack. Scripting languages are use to perform this Attack.

D. Types of XSS

Reflected

Sometimes user clicking on a malicious link and submit a crafted form, then user redirected to the un-trusted server. After that attack is reflected back to the browser from that Un–trusted server and if browser executes that code attack is perform. This type of attack is not permanently stored on a system.

Stored Attack

In this type of an attack a malicious code is stored on a targeted server. If a particular web page visited by user, code gets executed and attack is performed. And it sends user confidential data to the attacker's site.

II. RELATED WORK

Stephen W. Boyd and Angelos D. Keromytis [] in 2004, proposed a method named as Instruction Set Randomization. In this approach a random integer is appended with the valid sql keyword before sending to database. Recognition of a random integer is difficult for both attacker and database too. So author proposed independent module which decodes SQL keywords with their original name before sending to database. This method has negligible performance overhead.

William G.J. Halfond and Alessandro Orso [3] in 2005, propose an approach AMNESIA (Analysis and Monitoring for NEutralizing SQL Injection Attacks).This approach is based on static and dynamic analysis of queries. In Static phase, query model is generated at each point of access to the database. In dynamic phase, queries are intercepted before sending to the database and are checked against statically to build model. Performance of this approach is totally based on the static analysis for building query models.

Russell A. McClure and Ingolf H. Krüger [4, 13] in year 2005, this approach is based on the object oriented programming. Their solution consists of an executable Sqldomgen which is executed against a database [b].This is referred as a sql domain object model (SQL).These classes are useful to construct a dynamic SQL statement with manipulating any string. In this approach every valid SQL statement is constructing using an object data model. Next, they obtain the schema of the database, and then iterate through the tables and columns contained in the schema and output number of files containing a strongly typed instances of the abstract object model [13].

G.T. Buehrer and B.W.Weide et al. [9] in 2005, propose an approach SQL Guard. In this approach queries are checked at runtime. Here, the runtime evaluation of a query based on a model which is expressed as a grammar that only accepts legal queries. SQL Guard approach use secret key to delimit user input during parsing by runtime checker. SQL Guard approach is stopping all type of SQLIA except stored procedures.

Su and Wassermann [5] in 2006, proposed an approach named as SQL Checker here they implement their algorithm on a real time environment and are checked at a runtime. A secret key is used to delimit the user input. Overhead of this method is low.

Prithvi Bisht, P. Madhusudan [7, 8] in 2007, proposed an approach CANDID. This method is based on a dynamic candidate evaluations method, which automatically prevent SQLIA. This framework dynamically extracts the query structures from an every SQL query location which are projected by the developer (programmer).This approach solves the issue of a manually modifying the application to create the prepared statements.

Rattipong Putthacharoen and Pratheep Bunyatnoparat [13] in 2011, proposed a method based on a concept name as a dynamic cookies rewriting. This approach is work with cookies. A proxy agent is placed between client and server. Cookies rewriting method can changes the value of name attribute in the cookies field. Those cookies are stored with their original names at server. As browser's database do not store original information of cookies, so even if attackers steal cookies from the database, they cannot be used later to impersonate the users [13]. The tool detected both categories of XSS attack without having any changes made at the client and server site [13]. But the proxy failed to intercept https requests coming from the client [13].

E. Galan, A. Alcaide, A. Orfila, J. Blasco [13] in 2010, proposed a method name as a multi-agent scanner. This method is work well with both type of XSS Attack namely stored and reflected attack. Proposed method is tested in different scenarios; secured and unsecured, but only basic attack vectors were tested, more vectors can be added to test the accuracy of their approach [13].

Atul S. Choudhary, M.L.Dhore [13] in 2012, proposed an approach CIDT (code injection detection tool).Here query detector is used to detect SQLIA and script detector is used to detect XSS. This approach is work with both type of code injection attack. But this method is fail to detect stored procedure attack.

Pankaj Sharma, Rahul Johari and S.S.Sarma[11] in 2012, proposed an extension to the MHAPSIA model. This model is works in a two phase Safe or in a production mode.

In a safe mode legitimate query model is created for SQL and XSS statements. and in a production mode all queries are complies with query model generated safe mode.

III. Proposed Framework

System architecture of Reverse Proxy Server approach is shown in Fig 2. Reverse Proxy Server plays an important role in avoiding SQL injection and Cross-site Scripting Attack. Reverse Proxy is technique which is used to sanitize the user's input that may transforms from client system to backend database system. Working of this approach is described below.

E. User Interface

Banking Application is develop to show, how web application weaknesses affect the level of security. Client system is dump system normally used by the attacker to perform illegal activities. Normally user puts a valid username and password to access their account. Most of the web application front page is a login page and that page is targeted by highly malicious users. Attacker makes a use of username and password to perform an attack so; work on these fields is an important task. Here, the password is encrypted using encryption algorithm MD5. And this information is send to the filtering module which performs its task.

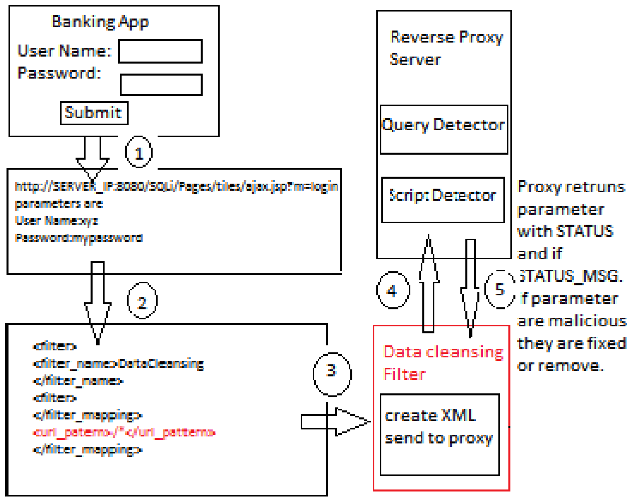


Figure II: System Architecture

F. Filtering Module

Filtering module acquire the details coming from user end (Client) System.

These request need to filter be filter with high level of security policies. For a security purpose or to sanitize user requests, data is forwarded to the Data Cleansing Filter.

G. *Data Cleansing Filter*

Data is always forward in a specific structure from one end to another. So, Here XML file is created which send from client to Reverse Proxy Server. XML file having 2 attribute request and header. Request can take information about user name and password, Where as header can take Browser name, IP address, Date, Time information.

H. *Reverse Proxy Server*

Reverse proxy Server which takes XML file as an input and work on it. Here the query detector module and Script detector is placed which sanitize user request and provide response to user.

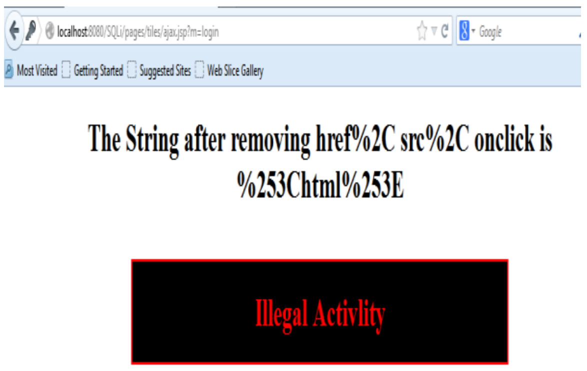
I. *Query Detector*

Query detector is used to sanitize the SQL request coming from front end side. SQL pattern matching function which perform matching with predefined SQL keyword and operator like =. When illegal symbol found or match occurs, malicious request is detected and a response is send to user's browser.

J. *Script Detector*

Script detector used to sanitize the scripting request. Attacker use scripting language is a tag based language to perform attacker.

If input is a scripted form it will send to the script detector. Script detector checks input request token by token with predefined script pattern. Fig 3 shows the output on attackers screen when input is in a scripted from using tags like <html>.



The String after removing href%2C src%2C onclick is %253Chtml%253E

Illegal Activity

FIGURE III: Output of a Detected Attack

IV. Result And Analysis

Result of Reverse Proxy Server approach is shown in table 1.This Reverse Proxy Server approach is tested with different attack types.

TABLE I
Comparison of Reverse Proxy Server With Alternative Approaches

Approach	Type of Attacks								
	SQLIA							XSS Attack	
	Tautology	Illegal	Union	Piggy-back	Stored	Inference	Alter Encoding	Stored	Reflected
SQLrand	*	X	*	*	X	*	X	X	X
AMNESIA	*	*	*	*	X	*	*	X	X
CANDID	P	P	P	P	P	P	P	X	X
SQL check	*	*	*	*	X	*	*	X	X
SQL Guard	*	*	*	*	X	*	*	X	X
SQL Dom	*	*	*	*	X	*	*	X	X
Dynamic cookies rewriting	X	X	X	X	X	X	X	*	*
Multi-agent Scanner	X	X	X	X	X	X	X	*	*
CIDT	*	*	*	*	X	*	*	*	*
Reverse proxy server	*	*	*	*	*	*	*	*	*

TABLE II
PERCENTAGE OF TOOLS DETECT OR PREVENT ATTACK

Approach	% Tool stop all Attack type (*)	% Tool can address Attack Partially (p)	%Tool cannot stop Attack of that type(X)
SQLrand	44	0	56
AMNESIA	67	0	33
CANDID	0	78	22
SQL check	67	0	33
SQL Guard	67	0	33
SQL Dom	67	0	33
Dynamic	22	0	78
Multi-agent	22	0	78
CIDT	89	0	11
Reverse proxy	100	0	0

Reverse Proxy Server successful to detect and prevent in all types of SQL Injection Attack and XSS Attack. * represent total prevention from Attack, p represent partial prevention of attack and tools cannot stop particular type of attack represented by X. Table 2 illustrates the percentage of tools that stop all types of Attack and that is calculated by following formula:

$$\frac{\text{No. of successfully detected attacks}}{\text{Total No. of attack}} * 100$$

According to above calculations which are shown in table 2, Reverse Proxy Server Approach is superior among all other tools or approaches describe earlier. Because it not only Detect the attacks but also automatically prevent attacks.

V. CONCLUSION

In this paper all SQL injection and Cross-site Scripting attacks are listed. Also a brief study about all approaches which contributing in Detection and Prevention of Attacks.This approach provides automatic prevents against SQLIA and XSS Vulnerabilities.

The Reverse Proxy Server approach is successful in detecting both types of SQL Injection Attack and Cross-site Scripting Attack.

This approach is works only with existing attack. Future scope of this approach is, this approach is needed to be check with more complex attack inputs.

REFERENCES

[1] S. W. Boyd and A. D. Keromytis. SQLrand: Preventing SQL Injection Attacks. In Proceedings of the 2nd Applied Cryptography and Network Security Conference, pp 292–302, June 2004.

[2] S.Fouzul Hidhaya, Angelina Geetha ‘Intrusion protection against SQL Injection Attacks Using a Reverse proxy’ Natarajan Meghanathan, et al. (Eds): SIPM, FCST, ITCA, WSE, ACSIT, CS & IT 06, pp. 129–144, 2012.

[3] William G.J. Halfond, Alessandro Orso, “AMNESIA: Analysis and Monitoring for NEutralizing SQL-Injection Attacks”, ACM-05 USA, November 7-11, 2005, pp 174-183 Long Beach, California.

[4] R.A. McClure, and I.H. Kruger, "SQL DOM: compile time checking of dynamic SQL statements," Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on, pp. 88- 96, 15-21 May 2005.

[5] Z. Su and G. Wassermann “The essence of command injection attacks in web applications”. In ACM Symposium on Principles of Programming Languages (POPL’2006), January 2006.

[6] Sruthy Manmadan and Mahesh T.2012”A Method Of Detecting SQL Injection Attack To Secure Web Applications”IJDPS.2012.3601.

[7] P. Bisht, P. Madhusudan, and V. N. Venkatakrishnan. CANDID: Dynamic Candidate Evaluations for Automatic Prevention of SQL Injection Attacks. ACM Trans. Inf. Secur., 13(2):1–39, 2010.

[8] Prithvi B, Madhusudan P, Venkatakrishnan VN (2010) CANDID: dynamic candidate evaluations for automatic prevention of SQL injection attacks. ACM Trans Inf Syst Secur 13(2):1–39.

[9] G.T.Buerer,B.W.Weide,and P.A.G Siviliotti. Using Parse Tree validation to Prevent SQL Injection Attacks.In International Workshop on Software Enggineering and Middeware(SEM)2005.

[10] S. Ali, SK. Shahzad and H. Javed, “SQLIPA: An Authentication Mechanism against SQL Injection,” European Journal of Scientific Research ISSN 1450-216X Vol.38 No.4 (2009), pp 604-611.

[11] Pankaj Sharma , Rahul Johari , S. S. Sarma’ Integrated approach to prevent SQL injection attack and reflected cross site scripting attack’ Int J Syst Assur Eng Manag (Oct-Dec 2012) 3(4):343–351.

[12] Atefeh Tajpour, Maslin Masrom and Suhaimi Ibrahmi “SQL Injection Detection and Prevention Tools Assessment”, IEEE, 2010, 978-4244-5539-3/10/.

[13] Atul S. Choudhary and M.L.Dhore,”CIDT: Detection of Mali-cous code Injection Attacks on Web Application” International Journal of Computer Application, Volume 52-No.2, august 2012.