PEER- REVIEWED INTERNATIONAL JOURNAL

Aarhat Multidisciplinary International Education Research Journal (AMIERJ)

ISSN 2278-5655

Bi-Monthly

VOL - II | ISSUES - V | [2013]

Chief-Editor:

Ubale Amol Baban

[**Editorial/Head Office**: 108, Gokuldham Society, Dr.Ambedkar chowk, Near TV Towar,Badlapur, MS

PLUGIN TOOL: OPEN SOURCE APPLICATION FOR IDENTIFICATIONS AND PREVENTION OF ATTACKS ON WEB APPLICATION

Girish J. Navale
RMD Sinhgad Institute of Technology,
Pune, India

Vanita Babanne
RMD Sinhgad Institute of Technology,
Pune, India

Abstract

Threre are various attacks in internet. The attacker gain s the unauthorized acess to the web based applications and performs data theft and modification to the database. SQL injection attacks are mostly found on web based applications. The unauthorized user gain acess to the system by creating sql queries that are totally different what the application needs. The social web sites like facebook, orkut, twitter involves sql injection attacks where hacker gets the access of user accounts and involves changing the profile information passing vulgar messages to the friends. The online banking transaction involves sql injection attacks where transaction by where unauthorized user is done. The The username password and confidential information is leaked. The plug-in tool is open source tool for detecting the sql injection and cross side scripting. It supports all kind of database and operating system.

Introduction

The fast growth of internet involves exchange of information and transaction .The more use and complexity of internet increases the more security related issues issue arrises. The web security consist of hiding confidential details and restrict the unauthorized acess of the system[1]. The web attacks are due to poor design of architecture of the system. Designers are not designing proper tier architecture and validation of the data passed between the applications. The Security issues of each web based application should be involved after the web based application is created and deployed on internet[2][3].The various attacks on web based application are as follows.

1) **Cross side scripting** :- The web based applications redirect the user data to a web browser without encrypting and validating. This drawback allows hacker to send user data which is executed in web browser. This attack involves phishing of web site attack and hijack of the data[11][12]
2) **Injection flaws** :- Interpreter software takes the user data as input. The attacker can take the data.
3) **Failure to restrict URL access**:- For security purpose the web application restrict user acess to certain URL.The attacker monitors the restricted URL and gets the acess of web pages and [4]
4) **Insecure communication**:- The network data is in unencrypted format which leads to hijacking of the data.
5) **Insecure cryptography**:- The insufficient encryption of sensitive
6) **information**

Background:-

SQL Injection is kind of attack used to hijack any types of SQL database. The malicious user inject SQL Commands into SQL statements through the input of web page [3][4][5]. The injected SQL Commands can compromise the security Of web application

SQL Injection based on 1=1

Select * from college where id=007 or 1=1

The above statement is correct and will return all the record sets since 1=1 is always true.

The SQL Injection based on = “ “ =””

SQL = “SELECT * from users where Name = “ “ or “”=” and pass “” or “”=”

Where “”=” always true.

Cross side scripting:-

This attack occurs where attacker uses a web application to send malicious code in terms of browser side script to different end users. The hacker transfers the malicious script to the user who is not suspecting. The target browser is unable to detect that the script is not authorized and execute it. The malicious script take the access of the session, cookies, token or any other sensitive information of the browser. The screen modify contents of HTML page. It occurs when entry of data is done from unauthorized source.

Psersistent XSS attacks:- Permanent storage on target server of injected script.

Reflected XSS attacks:- it includes error messages , search result or any other response which include some of the input sent to server such as part of requests. When a user click on the malicious link the injected code travels to the vulnerable site which reflects attack back to the users browser. The browser then executes the code because it comes from trusted source[9].

Related Work:-

Waves[3] performs the machine learning approaches for GUI testing to identify the points which causes web attacks

Rusell A.McClure[6] implements object oriented approach . It consist of SQLdomgen . After exection of this exe the output is dynamic link library SQL DOM which contains class files to implement dynamic SQL queries without further changes in the string . The drawback it detects the attack at compile time and unable to detect the attack at run time. It is not that much effective against store procedure

William G.J Halfond[7] implements a model based approach AMNESIA which is effective for static and dynamic analysis . It uses hotspot concept which monitors the query when it is passed as input. It detects 1470 attacks. It is not suitable for detecting both code injection and cross side scripting attacks.Engin Kirda, Christopher Kruegel, Giovanni Vigana implements a web based firewalls .It is present at client side . It is benefitial since less traffic is there since all validation are done at client side. It reduces budon on server to check the validations. It lacks SSL support and XSS attacks.

David Scott and Richard[10] sharp implements application level firewall which follows all the standards of security levels. It contains all the features which are required for security measure. The limitation of this method is unable to detect the XSS attacks.

Shaukat Ali , Azhar Rauf , Huma Javed[**8**] uses store procedure and hash values for identification . The user hash values for user name and password is generated automatically . This method detected only tautology type SQLIA and not other types of SQL injection attack.Rattipong putthacheron , Pratheep Bunyatnoparat

Proposed Methodology:

The plug-in tool is open source application

The invalid query input and malicious script from the user. It checks the input data from the user. Check all the validations of query and script. If it passes all the Validation then only it is redirected towards the target server. When user login to the web based application the session tracker provides session id which is unique for every user[12]. The session number is valid until the user log out from the system. The Query identifier checks the the query syntax as per the business validations and script identifier checks the suspected scripts in the attached with the application. It monitors the HTML Tag which are given as input. The filtering process removes all the unwanted and suspicious tages from the input and process with the remaining tag which are validated as safety tag . The block diagram is as shown in figure 1.

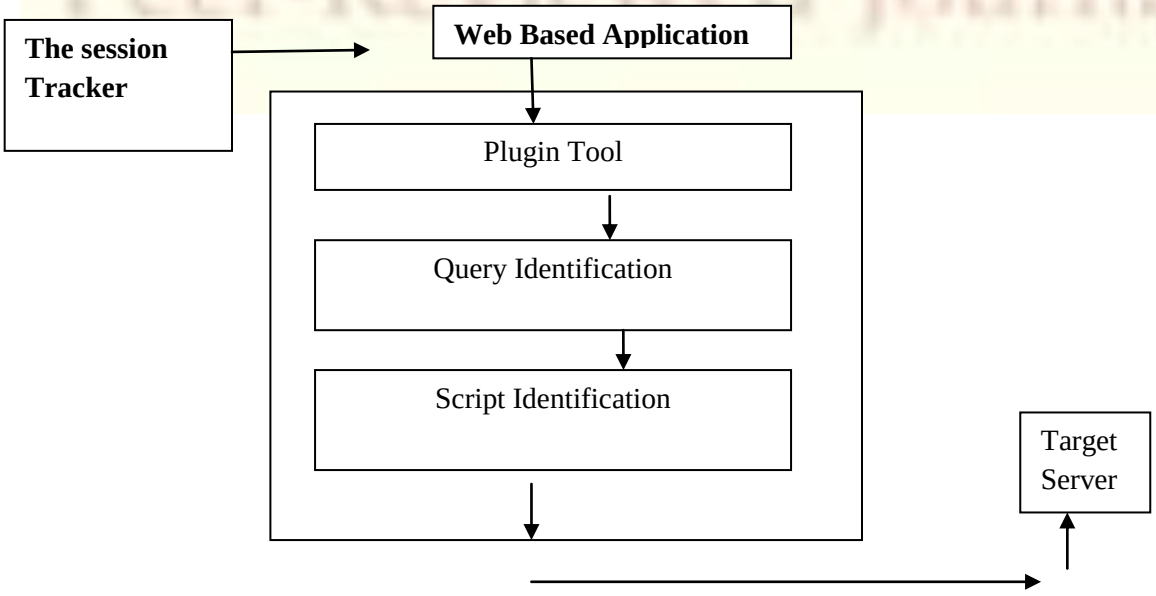
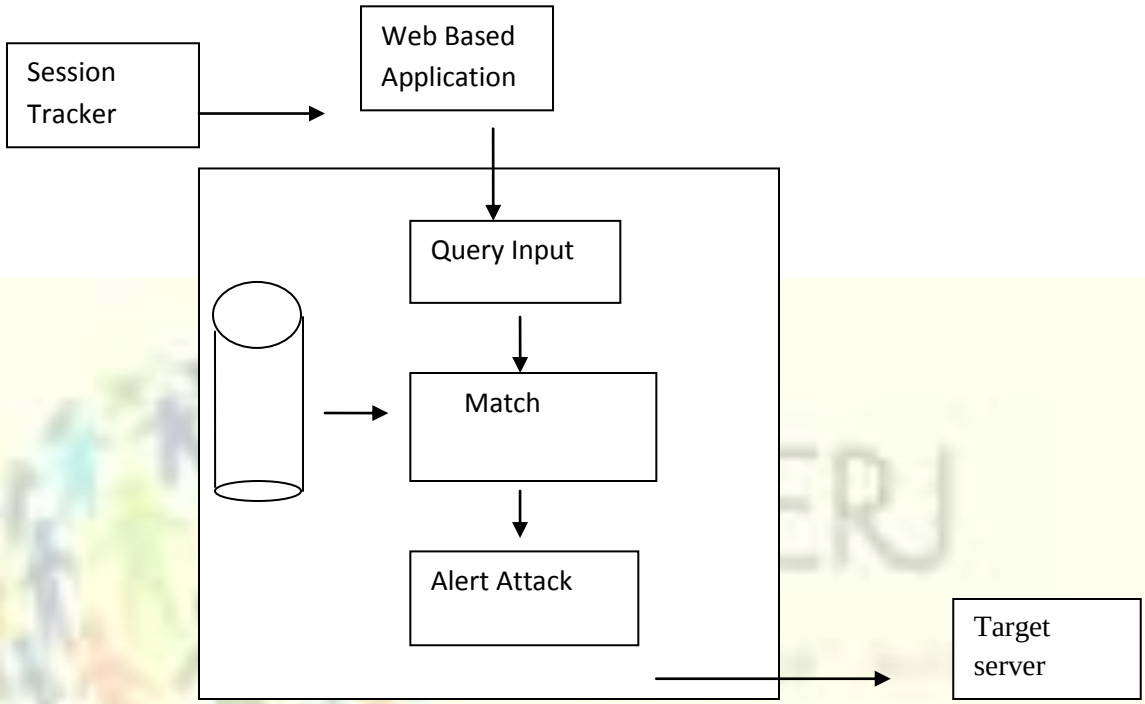


Fig – 1) Plugin Tool for Query identification and script valdations:-



Tags which are given as input. The filtering process removes all the unwanted and suspicious tages from the input and process with the remaining tag which are validated as safety tag . The block diagram is as shown in figure.

HTML Filter:-

The HTML filters suspicious and unwanted tags, attributes from the given tag. The standard tag list is made and every tag is validated against the standard list. The list is made by the Open Web Application Security projects. There are standards define by the security projects. There are some business validations and logics which filters the malicious tags in HTML input. It also filters the attributes which are not safe. The hyper link attributes which are suspicious are filtered.

The filtering process de vides the HTML strings into tokenizer.

Tokenizer:-

The tokenizer devides the HTML text as group of words defined as tokens. Token is collection of strings It can be tag start (, comment /*, closing) . Thus a list of tokens generated and compare

With the standard list of security projects .

HTML Encoder:-

The tokens which are matched with the standard list are encoded by HTML encoder.

Script Pattern :-

The pattern contains list of all start tags , end tags, comment part, URL pattern

Pattern Matcher:-

It accepts the input given by the users in the form of tokens and compare it with the standard Script patterns Those who are not matching are in rejected list and matching tags are processed For encoding.

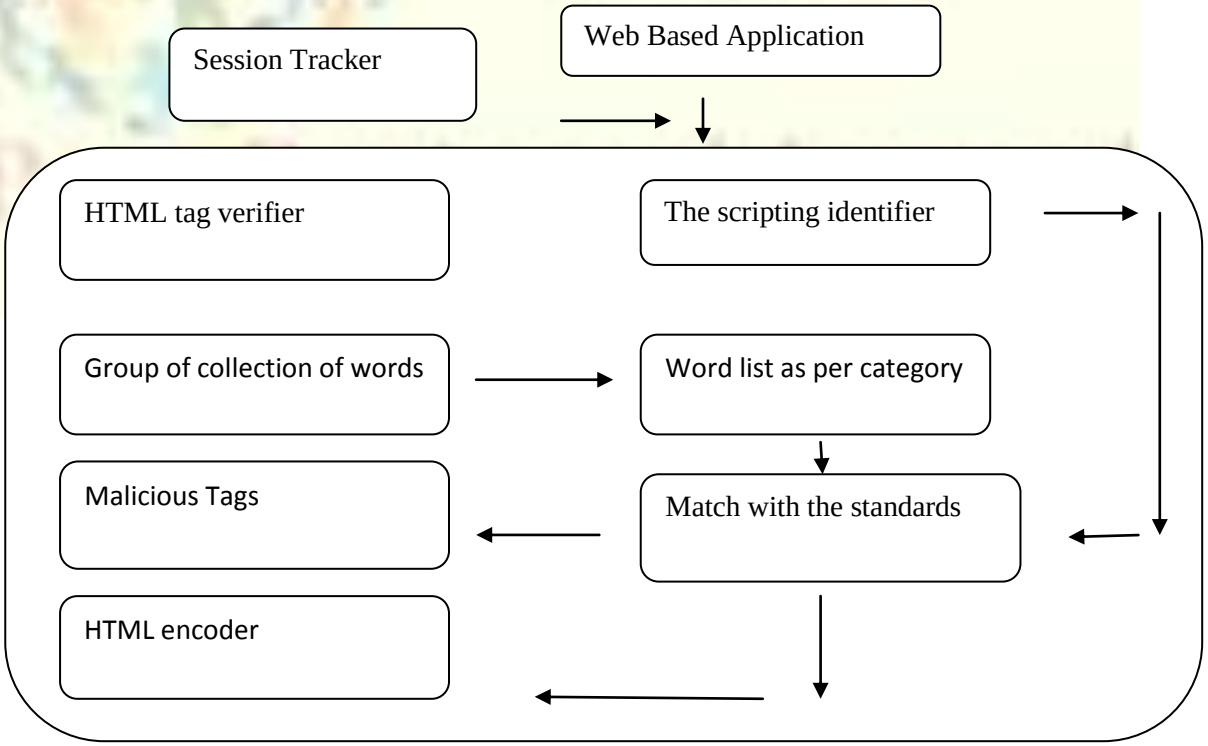


Fig-3) Block diagram of script identifications.

System Implementation:-

Algorithm:-

Step 1: Take the user name and passwords from the login of web application

Step 2: The session is active for the user name of web based application

Step 3: Store the username in the file

Step 4: Initialize the attak with false . Initially there will be no attack

Step 5:- Repeat for each line of query input

Check the query input matches with standard query validations which are stored in a file Until the end of query input

Step 6:- Set the Query input as string

Step 7:- If the query input is not matched then

Set webattack -> True

Else

Set webattack -> False

End if

Step 8:- if webattack equals to true

Restrict the user entry from entering into the main database.

Else

The user is valid and permission to acess the data from database

End

The user request is given to the query identifier which matches the query input with standard input file . If it matches then entry to the database is permitted otherwise it filters the user request from accessing the database.

Script Identifier:-

Step 1: Accept the user input in terms of scripts, tags, links or Urls

Step 2: Combine the input in collection of words .

Step 3:- The collection of words are stored in a list

Step 4:- With the list of collection of words check if the word is allowed to enter

Repeat { for every word }

Step 5:- if a collection of word is comment then discard it.

Step 6:- if { a collection of word is start tag }

Extract the tags and all its attributes

If (Malicious Tag}

Discard the tag.

Step 7: if { accepted tag} then perform Separate every attribute of the tag

Check the hyper link and different sources of the tags which are in embedded format.

Check the style attribute.

Encoding of tag is done

Else

The tag is suspicious and removed. This algorithm takes the user input and makes the list of tokens. After that matching with standard list of script identifier and regular expressions The malicious tags are removed

Before it is passed to the target server.

Discussion:-

SQL injection attacks done on web application fall apart security of the systems and alteration in the database of the web application.SQL injection and cross side scripting is the frequent types of attacks which are carried out for the fall apart security .The plugin tool helps to detect and remove all kinds of SQL injection and cross side scripting attack

Conclusion:-

The plug-in system is useful for identifying the different SQL injection attacks and diverted the attacks from altering the web application altered. This plugin system accepts all supports all kind of databases Without modification in code. The identification and preventation of attack is done automatically.

There is no limitation of no of requests which can be tackled by plugin tools and without much delay.

The proposed work is to enhance the existing systems and withdraw the undependable security Points in plugin tools.

8. REFERENCES

Bernard Menezes, Indian Institute of Tech, Mumbai, "Network Security and Cryptography", Cenage Learning Publiactions.

Research Report by Ponemon Institute "Second Annual Cost of Cyber Crime Study Benchmark Study of U.S. Company" Sponsored by ArcSight, an HP Company Independently conducted by Ponemon Institute LLC, Publication Date: August 2011.

Inyong Lee, Soonki Jeong, Sangsoo Yeoc, Jongsub Moon, "A novel method for SQL injection attack detection based on removing SQL query attribute values", Volume 55, Issues 1–2, January 2012, pp 58–68.

Diallo Abdoulaye Kindy and Al-Sakib Khan Pathan "A Survey on SQL Injection: Vulnerabilities, attacks, and Prevention Techniques" 2011 IEEE 15th International Symposium on Consumer Electronics, pp 468-471.

K. Amirtahmasebi, S. R. Jalalinia, S. Khadem, "A survey of SQL injection defense mechanisms," Proc. Of ICITST 2009, pp.1-8, 9-12 Nov. 2009.

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.

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.

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.

Mr. Dan Kuykendall, "Detecting Persistent Cross-Site Scripting", White paper, Volume 11211, eEye Digital Security, 2010.

David Scott, Richard Sharp, "Abstracting Application Level Web Security", WWW '02 11th International

Conference on World Wide Web, ACM, Network 2002, pp 396-407..

Chris Palmer "Secure Session Management with Cookies for Web Applications", iSEC Partners, Inc San Francisco, Version 1.1, Sept 10 2008.

"Ethical Hacking Tutorials", http://www.breakthesecurity.com/2012/01/how-to-do-cookie-stealing-with-cross.html.