SQLI-XSS Attacks and Countermeasures

Monica M. Koli
Department of Computer Engineering
RamraoAdik Institute of Technology
Nerul, Navi Mumbai, India
Email: monicakoli26@gmail.com

Pooja S. Kale
Department of Computer Engineering
RamraoAdik Institute of Technology
Nerul, NaviMumbai , India
Email: kpooja550@gmail.com

Pranali H.Karbhari
Department of Computer Engineering
RamraoAdik Institute of Technology
Nerul,Navi Mumbai
Email: karbharipranali777@gmail.com

Prathmesh N. Gunjgur
Department of Computer Engineering
RamraoAdik Institute of Technology
Nerul,Navi Mumbai
Email: gunjgur.prathmesh@gmail.com

***Abstract*—**SQL Injection (SQLI) and Cross-site scripting (XSS) attacks pose serious security threats that are used to sneak into web applications today. These attacks can lead to total breach of security to the privacy of the clients of a particular web site. In this study, we propose a vulnerability scanner that would be able to detect absolute SQLI and XSS attacks. This scanner can be used as a tool to discover the vulnerabilities present in a web application as well as to test the efficiency of websites before hosting websites. In the hindmost part of our work we came up with a security mechanism to counter such aforementioned attacks. We study various proposals in this regard and the state-of-the–art performance analysis is presented followed by our notations.

***Keywords* —** *SQL Injection(SQLI)Attacks, Cross-site scripting attack, Vulnerability scanner.*

*****

I. INTRODUCTION

In past few decades use of web application for information exchange has augmented a lot as these applications provide simple interface through which client can communicate. Thus service providers use databases for storing user sensitive data. These databases can be exploited by attackers for gaining users personnel details as web applications are easy way for attackers to access database. Moreover ever-increasing numbers of high profile data violation have plagued organizations over the past decade which is called as 'injection attacks'. The Survey of Open Source Web Application Security Project (OWASP) states that SQLI and XSS attack are the ones that are most widely performed.

SQL injection (SQLI) attack occurs when malicious SQL code is inserted into the input submitted by user. Thus SQLI executes if the input is not properly validated. Through SQLI queries or script attacker can gain unauthorized access of database, execute queries and then perform data manipulation operation on databases. For injecting attack queries can be dynamically constructed as the part of input.

Apart from SQL injection attack the other common attack found is XSS attack. This type of exploits target the HTML output function that sends data to the browser. The basic idea behind XSS injection is to use special characters which would cause Web browser interpreters to switch from data context to code context. These special characters includes the <script> tag using which the attacker can invoke java-script interpreter. Thus, attacker can perform exploits on webpage's like web content manipulation, hacking user cookies etc. The input source that is manipulated by attacker mainly includes HTML forms, cookies and outgoing webpage's. These attacks may hamper the security aspect that is CIAA.

II. LITERATURE SURVEY

Sangita Roy *et al*. [1] proposed that attack signatures can be found by adding filter to HTTP request.If the web developer wants to check a web application, they need the home pageas input of the web application. The proposed tool crawlseach and every page of the web applicationand searches for vulnerable spots. After that it inserts SQL query to these spots and reports result of the injection.

S.Chandrasekar*et al*. [2] proposed that initially we will encode all JavaScript calls to syntactically invalid identifiers. After that we watchscripts trying to execute them by loading requested page in web browser. Finally,we decrypt already generated script IDs to restore the original source code.Then we verify that there is no malicious script in the page, and give the page to the user.

V. B. Gaikwad*etal.* [3] proposed the approach of detecting the web request parameters to detect XSS in it and it used HTML Parser to parse and modify the valid HTML with java script

 ISSN: 2321-8169
692 - 695

tags, expressions, method and method calls by injecting comments along with random tokens. To extract authentic response produced by server Script Engine is used.

Priti Singh et al.[4] schemed that their SQLI Detection method looks for attack signature by using filter for HTTP request send by the client. There system is comprehensive in the sense that it can be used with any web application. After that they have checked their proposed scanner by comparing it with other scanners. In cross site scripting detection method we provide server side protection by filtering JavaScript present in web application.

Digambarpatil et al. [5] recommended approach will try to classify the request as a query based request and then , will detect the various type of attacks if present. SQLIattack and XSS attack can both be detected by using this methodology. By the technique of SQL Injection attackers gain control over back-end databases by feeding the malicious code through front-end.

Thus some of the challenges faced are:

- The basic cause of SQLI attack is insufficient validation of user input.
- The XSS mainly occurs when attackers try to access stored cookies.
- Moreover every web application developers is not expert in the area of web security.
- As the databases underlying web applications often contain confidential information, thus there are chances of data leaks from attacks.

III. PROPOSED SYSTEM

This system is developed to replace the existing manual system for the scanning process which is a time consuming, costly and less interactive. The main features of this system will be to find different types of vulnerabilities by scanning data and after that it generates a report of whole scanned websites.

In our proposed system Fig.1, we recommend a SQL Injection Detection method that looks for attack signatures by using filter for the HTTP request which is send by user. We also propose a detection method for XSS attack. Thus this tool can be used by many web applications. This solution helps to protect against information leakage that happens at the user's environment. Cross-Site scripting attacks are difficult to detect and prevent but relatively easy to execute. Finally we test the proposed security method by carrying out the comparison of the proposed vulnerability scanner along with some other well-known scanners.

This system tends to replace the existing manual system which is costly, time consuming and less interactive. The main features of this system will be creating report by finding various types of vulnerabilities. This is done by storing Scanned data, initiating process, and after that by generating a report of whole scanned websites.

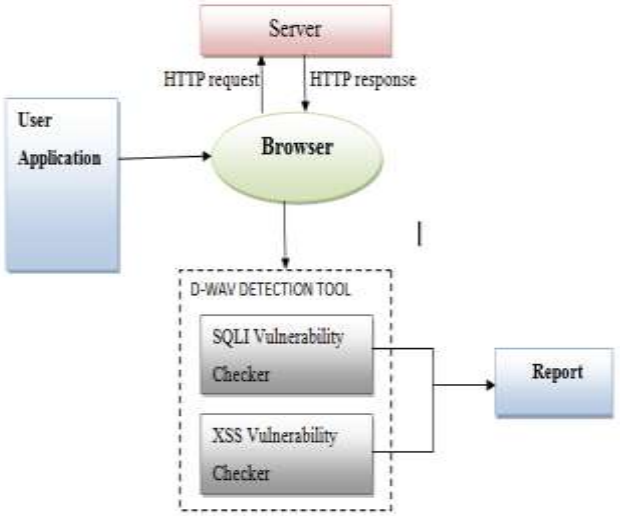

Fig.1. Proposed System Architecture

A. SQLI Detection

The proposed tool is used by a web application developer for SQLI detection. Here as input we need index or homepage of the web application. The tool scans each page of the application. Hence it finds vulnerable spots and injects SQL code to these vulnerable points. At last itreports result of the injection.

B. SQLI Detection Algorithm

In this section we present the SQL Injection scanner [9]. This tool has been specially designed for web developer. We need index or home page of the web application as input so as to find vulnerable spots. The tool crawls through the each page of web application for finding vulnerable spots. It then insertsSQL queries to these susceptible spot and later reports the exact result of the SQLI. The Scanner checks for SQLI vulnerabilities in three steps.

i. Crawling through the whole web application

We first go through the entire web application to search the input spots. Over here web application is constructed in the format of tree like structure.Then depth first search method is used for visiting pages of application.To check whether a page has been visited or not STATUSCHK variable is used.

ii. Scanning for vulnerable spots

The tool analyses the URL of each visited page and tries to identify the input points where SQL injection can be done. If the page accepts malicious user inputs then it is marked as susceptible to attack. For example, if we get an
URL like http://rait.ac.in/department/cs/cscourse.html then we conclude that, in this page we do not have any vulnerable spots. But if the URL is like http://pqr.site.com/cost.php?cost=60
Here if we write following SQL query:
SELECT * FROM cost WHERE cost=60.

In this query we can identify that costis the parameter and its value is 60. The parameter element is every time stable but an attacker can easily modify the value element. Thus this URL has a vulnerable spot.

iii. Generate Attack and produce Report

The last step is to insert the attack code at the susceptible spots recognized in the final step and report the result of the attack. Following things are required for attack generation:

- **Setup Payload**: Attack payload is created in this phase based on the existing universal SQLI attacks. Here payload is generated by using list of existing SQLIwhich are used by maximumnumber of attackers. Concealed database used is responsible for the result that we get from the attack.

- **Attack Generation:** Here attack is carried out by combining the attack payload to the original URL of the web application.

By sending various specially crafted attacks we request thescanner to check if SQL injection vulnerabilities liein a web application or not. We have reservedall the database error related to different SQLI attacks in tables which is used for testing susceptibility present. In this tool we use the content analysis to find database error. After sending the attack request the tool undoubtedly detects the result to check if there is any database error present. These errors are previously marked in our system. There is vulnerability present in the input of the page if error is detected in the response page.

C. XSS detection:

The fundamental components of our proposed architecture are:

1. A JavaScript detection component is used for determining if script tag is present or not. It is also used to know server's requestand response.

2. A XSS filter component, which is used to clean destructive script from the HTTP request and HTTP response.

3. A HTML Input filter component is determined in front of servlet component,which is used to investigate escape comments,Remove blanks space, balance HTML tags, protocol attributes from the incoming HTTP request and encrypt this parameters.

D. Cross Site Script Guard Algorithm:

This algorithm works by evaluating the probability of number of scripts that match with white and black listed sites.

Algorithm: E-Guard Algorithm

This algorithm uses following terms.

W: Whitelist;

B: Blacklist;

G: Greylist;

1. intXSSGuard(G} {
2. Read every line from console(G)
3. **If**(character==’<’){
   Increment the array of scripts
   Until character==’>’
   }
4. Print number of scripts in the input file.
5. Read the input file from console

//white listed scripts are checked

6. **If** (scripts in G matches with scripts in W){
   Increment the array of scripts
   Print the matched scripts
   Print number of matched scripts

}

//black listed scripts are checked

7. Read the input file from console
8. **If** (scripts in G matches with scripts in B){

Increment the array of scripts
Print the matched scripts
Print number of matched scripts
}

9. **If**(number of matched scripts with W>B){

Print the site as W and redirect it to
the respective website
}

10. **Else if**(number of matched scripts with B>W){
    Print the site as B

Print warning message to the user
}

11. **Else if**(no. of matched scripts equals W && B){

Print that check status of this site later
}

IV. PERFORMANCE ANALYSIS

A*: *Accuracy

Accuracy refers to how close a sample statistics to a population parameter. Thus, if you know that a sample mean is 99 and the true population mean is 100, you can make a statement about the sample accuracy.

$$A(M) = \frac{TNC + TPC}{(TNC + FPC + FNC + TPC)} \quad (1)$$

Where

TNC= number of true negative cases
FPC= number of false positive cases
FNC= number of false negative cases
TPC= number of true positive cases

B: False Positive Rate:

When healthy individuals incorrectly receivepositive test result this percentage is known as false positive rate.

 ISSN: 2321-8169
692 - 695

The false positive rate is

$$FPR = FP/(FP + TN) \quad (2)$$

Where,FP is number of false positives.
TN is number of true negatives.

TABLE I: COMPARATIVE STUDY OF SQLI AND XSS SCANNERS

Parameter	Netsparker	Acunetix	WebCruiser	Proposed Scanner
Vulnerability Detected	16/21	18/21	15/21	21/21
Accuracy	55	60	40	80
False Rate	3	12	2	1

From TABLE I we can consummate that existing scanners try to evaluate SQL queries that are given as response to submitted http request. Moreover they haphazardly apply all possible malicious code to all vulnerable points hence these tools require very long time to scan. These scanners also conclude an attack as successful if they receive a response different from previous one, this affects accuracy of result.

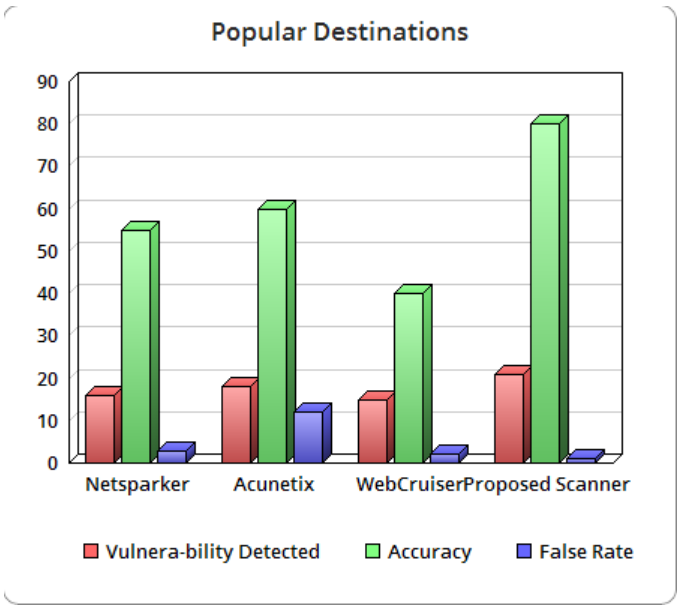

Fig.2. Vulnerability Detection Graph

The Fig.2 shows the graphical representation of TABLE I in formal bar graph.

V. CONCLUSION AND FUTURE ENHANCEMENT

Different taxonomy of methodsis available to prevent and detect SQL injection attacks as well as XSS attack. Thus web application developers must check their web applications for information leaks before making them public to ensure security of the application. The proposed scanner serves the same goal, it scans a web page in a controlled environment and discovers the SQLI and XSS vulnerabilities present and generates a report. Thereby it enhances security of application. This scanner discovers the vulnerabilities by crawling the web pages.

As detection and prevention techniques cannot provide complete protection against SQLI and XSS attacks thus thre is need for future enhancement. The proposed mechanisms encapsulate a wide range of SQLI and XSS attacks which will apex into a more secure and reliable database that can be protected against SQLI and XSS Attacks. But we can improve it further by combining some more attack detection techniques which will thereby increase security.

REFERENCES

[1] William G.J. Halfond, Jeremy Viegas, Alessandro Orso., "A Classification of SQL-Injection Attacks and Countermeasures," International Symposium on Secure Software Engineering, March 2006.

[2] S.Chandrasekar, "Detection and Prevention Techniques," 2nd International Conference on Computational Intelligence, United Kingdom pp. 216-221.

[3] L.K. Shar, H.B.K. Tan, "Mining Input Sanitization Patterns for Predicting SQL Injection and Cross Site Scripting Vulnerabilities," ICSE 12, IEEE, 2012, pp. 1293-1296.

[4] Sangita Roy, Avinash Kumar Singh and Ashok Singh Sairam, "Detecting and Defeating SQL Injection Attacks,"

[5] Acunetix of Acunetix Ltd [online]. Available: http://www.acunetix.com.

[6] OWASP, "Testing_for_Cross_site_scripting" [online]. Available: https://www.owasp.org/

[7] Janus Securities Web Cruiser [online]. Available: http://sec4app.com

[8] "Waterfall Model Diagram" [online]. Available: https://www.waterfall-model.com