Hybrid approach for Detection and Analysis of SQL and XSS vulnerabilities

Monali Shetty[1], Chirantar Nalawade[2]

[#1] *Assistant Professor , Computer Department, Fr. CRCE ,Mumbai University,Maharashtra , India*
[#2]*Student, Computer Engineering, Fr. CRCE ,Mumbai University, Maharashtra ,India*

Abstract - *Web applications have become one of the most popular targets of cyber-attacks during the last few years. According to Open Web Application Security Project report, SQL injection and XSS are top two vulnerabilities found to be present in majority of web application. As a result, identification and analysis of vulnerabilities present in the web applications are important to prevent potential attacks. Current industrial approaches involve white-box testing which examines source code of applications; whereas black-box testing makes use of external attacks on the application. However, white-box testing produces large number of false positives which decreases overall efficiency, whereas detection rate of vulnerability in black box testing is low. In this research paper, we present a new technique to find vulnerabilities which are able to enhance detection rate of vulnerabilities and increases efficiency by decreasing number of false positives as well as false negatives. We focus on an innovative tool that implements hybrid approach which combines white-box and black-box testing techniques. At the end we have given an evaluation table, which compares our scanner with other two web scanners.*

Keywords — *Web application security, SQL injection, XSS injection, vulnerability detection, hybrid approach, white -box testing, black-box testing.*

I. INTRODUCTION

Dependence on web applications is increasing very rapidly in recent time for social communication, health problems and financial transactions. For example, sale of on-line retail music accounts for fifty percent of total market place. One out of every four internet users make use of internet banking .As the range of web enabled smart-phones continues to grow; web applications have become enormous part of world economy. However, the presence of security loopholes or vulnerabilities allows malicious users to exploit these vulnerabilities, which shows that security of this segment is not in sync with its growth and reliability. However web developers are barely aware of the extent of threats to their sites, fragility of codes they develop, or the length at which attackers can leap to gain access to their systems.

According *Open Web Application Security Project* report 2016, SQL injection and XSS are the top two vulnerabilities found to be present in majority of web applications [1]. SQL injection vulnerability is database driven loophole in which malicious user can execute SQL queries directly without any validation from system. This attack is performed by changing the semantics of original SQL query by inserting special keyword, operators or statements with the help of non-validated input. At the successful performance of attack, malicious code gets executed in the database.

Cross Site Scripting Vulnerability is commonly known as XSS vulnerability, which aims to send malicious scripts to web browser generally making use of JavaScript .It occurs when a web application takes user inputs via HTTP request, or files without proper validation. The attacker can steal various session-id and cookies which allows attacker to impersonate the victim by performing social engineering attack, tricking the victim to divulge its private information e.g passwords .Malicious codes can be inserted in target field, addresses, comments etc.

White-box and black-box testing techniques are used for detection and minimization of vulnerabilities present in web application. Efficiency of testing techniques is measured by two factors which are false positives and false negatives as stated in [2], [3] and [4]. False positives are total number of vulnerabilities detected by testing techniques which are not actually present or exploitable. False negatives are total number of vulnerabilities present in the system which are not detected by a testing technique.

Black-box testing refers to testing a system without having specific knowledge of the internal workings in the system, no access to the source code, and no knowledge of the architecture. Black-box is also known as a dynamic approach for detection of vulnerability, as it requires execution of deployed application for detection of vulnerabilities.

White-box testing which is also known as clear box testing refers to testing a system with full knowledge and access to all source code and architecture documents. Having full access to this information can reveal bugs and vulnerabilities more quickly than the "trial and error" method of black-box testing. Additionally, you can be sure to get more complete testing coverage by knowing exactly what you have to test. However, it has been found out that white-box and black-testing both have major drawbacks. While black-box has advantages of less

manual intervention, more automation, usability and non-dependence of web application technologies, it has major drawbacks like less detection rate and higher false negatives. Moreover, black- box testing coverage domain is highly restricted and is dependent on performing tests which decreases its performance [3].Although white-box testing provides major advantages like complete code coverage, exact detection of cause of vulnerability in the source code, it has certain drawbacks like large number of false positives [4]. Primary purpose of our research is to design a tool that will implement hybrid approach to capitalize on strengths of both white-box and black-box testing whereas it will lead to decrease in both false positives and false negatives thereby increasing the overall efficiency of vulnerability detection. In the remaining part of the paper, Section 2 will discuss literature review of current research. Section 3 covers the challenges faced by existing systems and their drawbacks. The methodology and implementation details of our tool approach using hybrid analysis will be covered in Section 4. Section 5 will cover result analysis obtained on website. The final conclusion will be covered in Section 6.

II. RELATED WORK AND CURRENT TECHNOLOGIES

A. Related work

A novel methodology developed to identify potential attack scenarios targeting web application based on the dynamic analysis of the application following a black-box approach. This methodology is able to automatically exhibit causal dependencies between vulnerabilities and identify attack scenarios. This methodology has been successfully tested on a basic and simple web application. However, it is yet to be tested with a production-ready large scale web application to check whether this approach is scalable.[5]

A network based vulnerability scanner was able to detect majority of the pages in a web application which are vulnerable to SQL injection attacks. The tool also generated a report which helps the developers to fix the vulnerabilities. The false positives associated with the tool were also found to be minimal and the efficiency of the tool is dependent on the number of systems connected within the network. However, the tool had quite high requirements in terms of network bandwidth and number of systems. Also, in a single-user environment, the tool's performance was average and unimpressive. Thus, the tool is suitable for a multi-client networked environment rather than a single-user environment.[6]

[7] This paper proposed the mechanism for scanning web application based on injection point, which obtains the information of injection point and using black-box testing to analyze potential vulnerability tackled through vulnerable injection point. They stated that when using this approach, the bugs can be identified easily, thus reducing the debug time and increasing the efficiency. They also claimed that their approach could increase the vulnerability detection accuracy. They focused on SQL injection and XSS attacks.

This method analyzed the accuracy and time costs of various web application security scanners. [8] They found out that even when web application scanners are directed to vulnerable pages of a website, there is a significant discrepancy in the number of findings. Also, there is great difficulty of achieving accurate results over an infinite target space of custom web applications. Given the large number of vulnerabilities missed by tools even when fully trained it is clear that accuracy should still be the primary focus of security teams looking to acquire a tool.

B .Current Technologies

W3af: Its main goal is to create a framework to help secure your web applications by finding and exploiting all web application vulnerabilities. It is written in python which makes use of black box testing method.

Figure 1: Current major technologies used for detection of vulnerabilities in web pages.

Nikto is an Open Source (GPL) web server scanner which performs comprehensive tests against web servers for multiple items, including over 6700 potentially dangerous files/CGIs, checks for outdated versions of over 1250 servers, and version specific problems on over 270 servers. It also checks for server configuration items such as the presence of multiple index files, HTTP server options, and will attempt to identify installed web servers and software. Scan items and plugins are frequently updated and can be automatically updated.

Wapiti performs "black-box" scans, i.e. it does not study the source code of the application but will scans the webpages of the deployed web application, looking for scripts and forms where it can inject data.

Vega is a free, open source scanner and testing platform to test the security of web applications. Vega can help you find and validate SQL Injection, XSS, inadvertently disclosed sensitive information, and other vulnerabilities. It is written in Java; GUI based, and runs on Linux, OS X, and Windows. Vega includes an automated scanner for quick tests and an intercepting proxy for tactical inspection. The Vega scanner finds XSS, SQL injection, and other vulnerabilities. Vega can be extended using a powerful API in the language of the web: Java-script.

III. CHALLENGES FACED BY EXISTING SYSTEMS

Static approach or white-box testing can be applied in earlier stages of application; however it fails to detect the vulnerabilities in arguable amount of time if source code is huge. Also static approach creates a large number of false positives and false negatives. False positive is alarming developer about vulnerabilities that is not actually present in reality, whereas false negatives are neglecting the vulnerability even though it might exist in the system.

Because of the sheer complexity of architectures and volume of source code, white-box testing introduces challenges regarding how to best focus the testing and analysis efforts. Also, specialized knowledge and tools are typically required to assist with white box testing, such as debuggers and source code analyzers.

Also as number of lines of code increases there is tremendous increase in number of false positives. As a result overall efficiency is compromised.

Dynamic approach or black- box testing creates less number of false positives and false negatives than white-box testing, usually makes use of complex algorithms to perform an attack. However, since it requires several execution paths to be used, it requires a large number of test cases to build so that enough confidence levels can be built. Also black-box testing can notify the presence of vulnerability only in deployment stage of software.

In addition, if white box testing is performed using only static analysis techniques using the application source code and without access to a running system, it can be impossible for security analysts to identify flaws in applications.

Also, it has been found by Jason Basu, Elic Bursztein [9] that popular open source black-box testers like drupal, wordpress, 1.5strayhorn, phpBB12 are unable to detect even 50 % of total vulnerabilities as shown in Table 1.

Category	Drupal		phpBB12		Wordpress	
XSS	6	2	5	2	13	7
SQL	2	1	2	1	8	4

Table 1. Efficiency of well-known black-box testers.

Due to the lack of internal application knowledge, the uncovering of bugs and/or vulnerabilities can take significantly longer. Black box tests must be attempted against running instances of applications, so black box testing is typically limited to dynamic analysis such as running automated scanning tools and manual penetration testing. Also on basis of experiment conducted by Adam Doup´e [10], it has been found that commercial web application black box testers such as Acunetix, Appscan, Burp, w3af perform have been reported 60% of false negatives rate i.e. rate of detection is less than 40%.

This analyzer will overcome the existing system drawbacks since hybrid analysis will ensure that advantages of both black-box and white-box testing are used.

White Box testing: It will involve source code analysis using the method of taint analysis, since taint analysis provides large coverage area. Sources in taint analysis are identified automatically unlike the previous manual identification. After identification of user variables all other variables are examined whether they are tainted or not. Proper analysis of tainted variables and vulnerable statements from the database results in identification of vulnerability.

Black Box testing will involve building sufficient test cases to perform various attacks on the system. Both SQL and XSS test cases are independently run on the web application. Black box test cases which determined presence of vulnerability are examined to find out payload variables and exploited parameter in the web application.

The combined result of both white-box and black-box testing are analyzed to form the list of actual exploits in the source code. A report will be generated mentioning major vulnerabilities that were identified using black-box testing and their source code identified with help of white-box testing which makes overall process hybrid as explained in figure 2.

IV. IMPLEMENTATION OF HYBRID APPROACH

The proposed system will make use of hybrid approach to detect the SQL injection and cross site scripting vulnerabilities by making use of hybrid approach. Hybrid approach is current research topic in which both white-box and black-box testing techniques are combined to create model which will lead to reduction in false alarms of white-box testing by using black-box testing. In this project we shall perform both black-box and white-box testing .

This tool is divided in two main parts namely white-box tester and black-box tester.

Vulnerability Database: Vulnerability database contains data about the code which can bring vulnerability in the application. Vulnerability code database depend on the language of web application e.g. it is different for web application build in java and web application build in PHP.

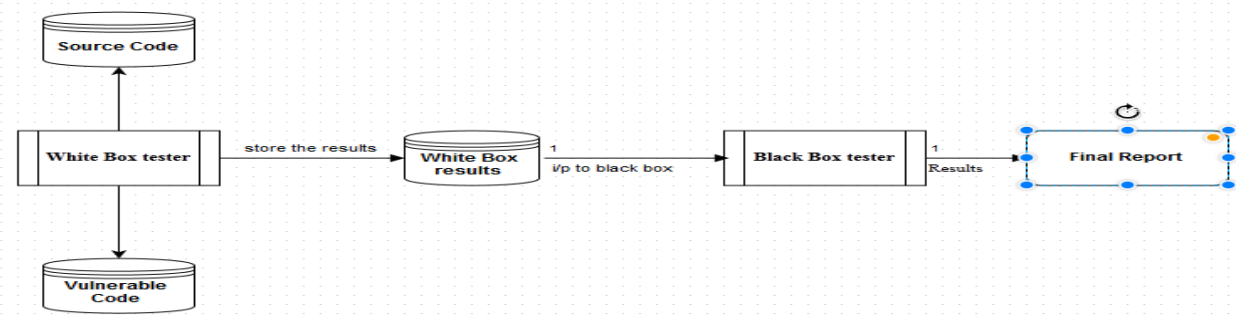

Figure 2. Block diagram

White Box tester: White box tester receives two inputs from each database namely vulnerability database and source code database. The white-box tester performs static analysis by comparing the code present in web application with all the code present in vulnerability database. It generates a list of vulnerabilities which contains many false positives and false negatives.

Black Box tester: Black-box tester receives input from result database to perform the various vulnerability tests to correctly identify all possible vulnerabilities. Each test consists of specific attack pattern.

V. RESULT ANALYSIS

Static analysis detects the root cause of vulnerabilities, however many vulnerabilities get either reported false or undetected. In black-box testing all the vulnerability detection is very accurate but needs a large amount of time. However this tool took advantages of both methods and has also curbed the huge amount of time required by black-box testing to give most accurate results.

Dummy website was created in order to perform various test cases and prove the working of the code. All the various test cases have been performed over this dummy website to generate result which can be compared with the realistic websites, and also with those generated by the open source tools. Figure 3 and 4 shows outputs of our toolkit with one of test case of dummy website.

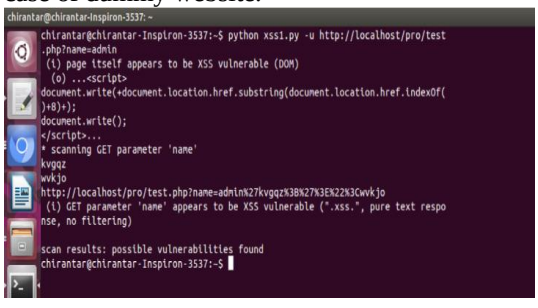

Figure.3 The output of XSS module of black-box tester of our toolkit which performed external attacks on website

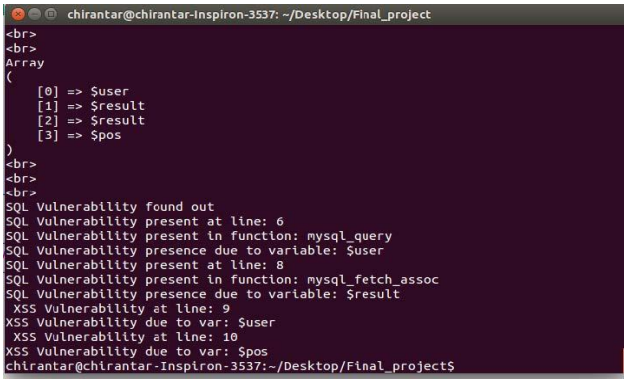

Figure 4. Detection of SQL injection through white- box tester

The hybrid approach implemented in the tool integrates both the results of white box and black box, forming a final integrated report of all vulnerabilities as shown in figure 5.

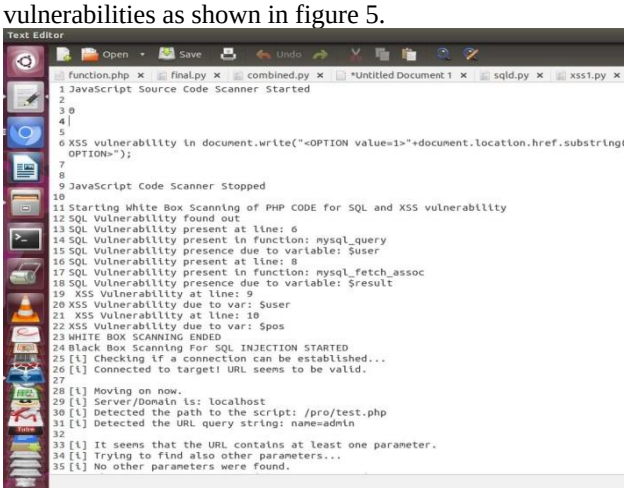

Fig. 5 Final generated report of our toolkit helps to detect vulnerabilities by static approach and then verified with help of black-box, giving more efficiency and accuracy

Parameter	Our tool	Pixy	Wapiti
Vulnerabilities detected	18	8	11
False Positives	3	12	3
False Negatives	1	4	8

Table 2. Experiment Results

It is thus found that overall efficiency of detection i.e false negatives is clearly far better than the open source tools used for testing like Wapiti and Pixy respectively as shown in table 2. Also, overall false positives developed due to use of white box testing are significantly reduced with help of black box testing. As a result, our tool has significantly low false positives compared to standard testing tools like pixy.

VI.CONCLUSION

Since aim of our project was to do research on combination of black-box and white-box testing technique to create a hybrid approach which will scan the vulnerabilities of websites, it was successfully completed with a hybrid tool that is able to detect major vulnerabilities in websites .Our tool detects top two OWASP's vulnerabilities which are SQL and XSS and testing of project and its modules is done by use of demo vulnerable websites and realistic vulnerable websites .It has been concluded that hybrid tool not only identifies the source code line-no at which vulnerability is present but it also has black box testing module which performs sophisticated attacks on hosted web applications. It has been observed that false positives i.e. non-detection of vulnerability and false negative i.e. detection of vulnerability which is not present has been drastically reduced by use of hybrid when compared with open source tools like Wapiti and Pixy. Thus this tool is one way solution to integration of advantages and removal of disadvantages of black-box and white-box testing and paved a new research in field of security testing.

References

[1] OWASP, Top. "Top 10–2013." The Ten Most Critical Web Application Security Risks (2013).

[2] Sekar, R. "An Efficient Black-box Technique for Defeating Web Application Attacks." NDSS. 2009.

[3] Z. Djuric, "A black-box testing tool for detecting sql injection vulnerabilities," in *978-1-4673-5256-7/13 IEEE*, 2013.

[4] G. S. Mukesh Kumar Gupta, M.C. Govil, "Static analysis approaches to detect sql injection and cross site scripting vulnerabilities in web applications: A survey," in IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014), May 2014.

[5] Eric Alata, Mohamed Kaaniche, Vincent Nicomette and Rim Akrout,"A Clustering Approach for Web Vulnerabilities Detection" in 17th Pacific Rim International Symposium, Dependable Computing (PRDC), IEEE ,2011.

[6] Avinash Kumar Singh and Sangita Roy,"A network based vulnerability scanner for detecting SQLI attacks in web applications" in 1st International Conference of Recent Advances in Information Technology (RAIT), IEEE, 2012.

[7] Jan-Min Chen and Chia-Lun Wu, "An automated vulnerability scanner for injection attack based on injection point" International Computer Symposium, IEEE, 2010.

[8] Larry Suto, "Analyzing the Accuracy and Time Costs of Web Application Security Scanners, San Francisco", 2012.

[9] Jason Bau, Elie Bursztein, Divij Gupta, John Mitchell," State of the Art: Automated Black-BoxWeb Application Vulnerability Testing" in IEEE Symposium on Security and Privacy, 2010.

[10] Adam Doup´e, Marco Cova, and Giovanni Vigna,"Why Johnny Can't Pentest: An Analysis of Black-box Web Vulnerability Scanners" in Detection of Intrusions and Malware, and Vulnerability Assessment, pp 111-131, Springer, 2010.