Literature Review on Studying the Effectiveness of Various Tools in Detecting the Protecting Mechanisms Implemented in Web-Applications

Ms. Shweta Thakre
M.E. Student, ARMIET, Mumbai.
Shweta.thakre1@gmail.com

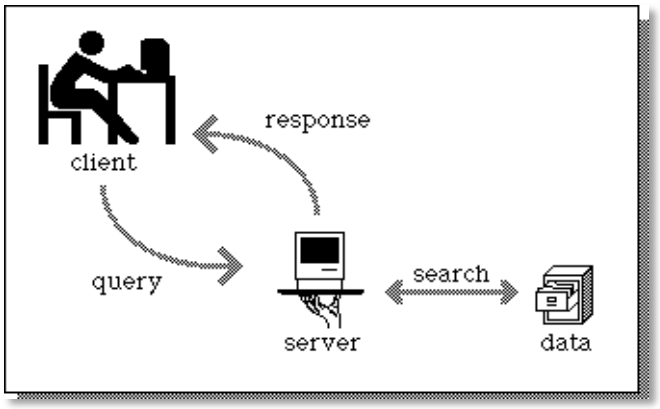
Prof. Sachin Bojewar
Associate Professor,
VIT, Mumbai.

Abstract* — *Web application security has become a big issue because of common vulnerabilities found in web applications. This paper illustrates a study on conducting security testing on an example application. The example application will be tested using a number of tools such as SQLmap, Acunetix, VEGA, IronWasp, WebCruiser, etc. Manual testing was also conducted. The testing results of different tools and manual testing will be compared and discussed. Our study shows manual testing is very important since some vulnerability types can only be found through manual testing and tester's observations, and it is important to utilize a variety of tools as well as conduct careful manual testing in order to find the most number of vulnerabilities in a web application. Based on this study, hands-on labs can be developed for teaching web security, software security testing, tools and other topics.

Keywords*—*Web application security, Security tools, SQL, Software security testing.

I. INTRODUCTION

Web applications today are highly functional, and rely upon a two-way flow of information between the server and browser. Security becomes a big issue because no one wants to use a web application if they believe their information will be disclosed to unauthorized parties [5]. Data in online world according to its level of confidentiality and value is in vulnerable condition when proper security measures are not taken to secure it. Flexibility is another most important point when we are providing banking, financial transactions and management things. When we have these requirements for flexible system with high security or error free environment then there is a tough task and that involve much better algorithms and security measures. However technology and vulnerabilities grew together and will. Attackers find loopholes, vulnerable part of applications and then get benefits of this all. Simultaneous growth in technology and new tactics can only survive better and can achieve the security level they are designed for. One another most important point is use of right platform and technology for desired security is use of right practices and technology, it's a fact that most of web applications have some user driven loopholes those which are responsible for applications vulnerable to attackers. The reason behind this is as simple as it is, for example a developer develops an application for servicing or aim is to provide functionality demanded and they are also yet unknown, not all of them but some from the vulnerabilities. Attacks increased as the increase in query languages because these query languages have the queries and operations for data alteration, download and upload, register etc. Mainly when these queries applied a single loophole can cause major data loses. Query when executed gives access to next level of services but if some attacker succeeds in performing a query by inserting some malicious input then he can modify or alter the database which is crucial to be secure.



Fig(1) Basic client server model diagram

Vulnerabilities commonly found in web applications include injection, cross-site scripting, cross-site request forgery, security misconfiguration, broken authentication and session management, and many more.

Penetration testing and static code analysis may be used to assess the vulnerabilities of web applications. Penetration testing is a method of security testing through the simulation of an attack. Static code analysis, also known as source code analysis is a code review process that examines the software's

source code for common coding errors and defects without execution.

Our key area in this project will be to demonstrate how any of the vulnerability occurs. Now a day's mostly applications make use of databases to store data. Since these applications are having back end thus a large number of queries are executed for performing insertion, deletion, retrieve or update of data from the database. Major part of these queries is formed using user input as these inputs decide the functionality or the result of the execution of the query. The purpose of this query can be modified maliciously by including some syntactic content to the user input. It can be harmful to the data stored in the database if the application responds to this malicious input provided by the attacker. Due to such attacks invalid data can be added to database or existing data can be modified or deleted. Thus it is very essential to securing web applications from such type of attacks. Another vulnerability I have focused upon is XSS [13][12][9]. In Cross-site scripting attacks, attackers inject malicious code into web applications i.e. scripts from outside sources. Most of the applications are vulnerable to this attack method due to the number of possible injection points and techniques available. The major difference between scripting attacks and other web application vulnerabilities is the victim of this attack i.e. they attack the users of that application not the Infrastructure of the application, still they can result in damage to a great extent. Modern day's pen-testers as well as attackers are mostly dependent on automated tools for detecting these vulnerabilities.

To demonstrate this we create a dummy website (college institution website/complaint logging website) which has following vulnerabilities:

- Injection attack → SQL Injection
- Cross Site Scripting XSS
- Session Management

Also this website will demonstrate most of the OWASP Top 10 web vulnerabilities. The latest OWASP Top 10 – 2013 vulnerabilities are as follows:

- A1 Injection
- A2 Broken Authentication and Session Management
- A3 Cross-Site Scripting (XSS)
- A4 Insecure Direct Object References
- A5 Security Misconfiguration
- A6 Sensitive Data Exposure
- A7 Missing Function Level Access Control
- A8 Cross-Site Request Forgery (CSRF)
- A9 Using Components with Known Vulnerabilities
- A10 Unvalidated Redirects and Forwards

This demo website will be helpful for both attackers & pen-testers as it will help security developers in making a decision that which security approach is best to be followed in order to protect their application from SQL-i & XSS attacks and for attackers there are exploit codes so that they can bypass these security filters.

II. Literature Survey

A. *About Security Testing Methodology:*

One of the basic tenets of software development is that you can't control the thing which you can't measure. There is nothing different in Security testing. Unfortunately, measuring security is a quiet difficult process. First of all what do we mean by testing? During the development phase of a web application there are many things that are need to be tested.

The Merriam-Webster Dictionary describes testing as:

- to put to test or proof.
- to be assigned a standing or evaluation based on tests.

In simple words testing is a process of comparing a system's state against a set of internationally accepted standards or an ideal system. In the security industry such tests are done against a set of criteria that are neither well defined nor complete. For this reason security testing is sometimes known as black art.

B. *Automatic Testing:*

When security testing is done with the help of click and scan type tools then such kind of Scanning is called Automatic Testing. There are many tools available in market for this purpose; some are open source while others are commercial tools. The role of automated tools. As we know Automated security analysis and testing tools are sold by a number of companies in day to day market. Limitations of these tools should be kept in mind so that you can use them for what they're good at.

C. *Manual Testing:*

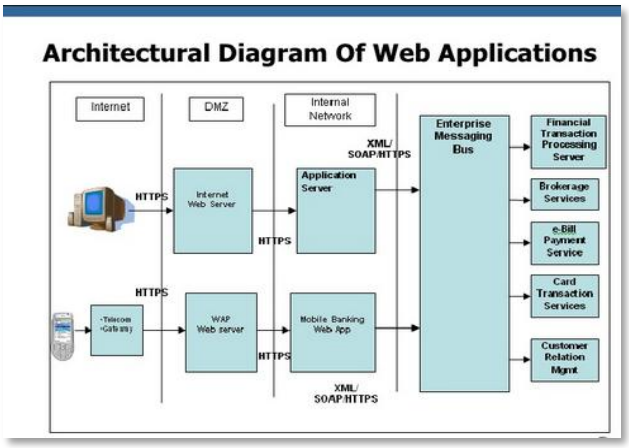
Manual testing is a set of human-driven inspections that are typically used for testing the Security issues about the people, policies, and processes. This set can also include inspection of decisions related to technology for example architectural designs, etc. Activities such as reviewing the documentation, secure coding policy standards, security requirements of company, and architectural designs all should be accomplished using manual inspection.

III. Problem Statement

To do Assessment of vulnerability scanners in detecting effectiveness of protection mechanisms implemented in the Web-applications. The importance of doing this assessment is that most of the developers use these tools in performing the penetration testing of their application and using the result of this assessment they can choose a right vulnerability scanner to perform penetration testing and detecting any flaws in their security approaches before attacker do so. After detecting these vulnerabilities security developers can harden the security measures in the application. Another issue is that some kind of manual exploits against the security approaches have to be developed so that the developers of the vulnerability scanners can use these exploits in upgrading their products and helping out the penetration tester in keeping their application secure.

IV. Problem Defination

This web application is a dummy application (college institution website/ complaint logging website) for getting a basic understanding of how to do vulnerability assessment and penetration testing in a web application by using tools & without tools i.e. manually.



Fig(2) Basic Architecture Diagram of Web Application.

V. Methodology/Architecture

A. *Front End:*

The programming will be done using the language Java i.e. Java Servlet pages (JSP). It is Sun Microsystems strategic language for platform independent programming. It is easy to use, efficient and flexible. This language is preferred because one can build a program using this object oriented and platform independent programming with less effort than with any other language.

B. *Back End:*

MySQL is one of the open source database management systems available on the market today. It is easy to use and administer, and comes with tools and wizards that make it easy to develop applications. The database itself has been redesigned to automatically perform many tuning functions, leaving you free to focus on most important tasks.

C. *Brief Overview of Tools & Technology to Be Used:*

For successfully performing security auditing of the dummy application, a number of tools are used; some are open source while some are commercial tools; some are automatic & some are semi-automatic or manual tools. Here only name of all those tools are given and other details about these tools like features, working and usage i.e. how to use them along with their different options is available on internet [21][22][23].

List of the Tools Used is as follows:

Sr No	Tool Name	License Type
1	SQLmap	Open Source
2	Acunetix	Commercial
3	VEGA	Commercial
4	IronWasp	Commercial
5	WebCruiser WVS	Open Source
6	Wikto	Open Source
7	W3af	Open Source
8	Xenotix XSS exploit Framework	Open Source
9	Zenmap	Open Source
10	Netsparker	Commercial
11	Nikto	Open Source

D. *Vulnerability Customization:*

Whenever some protection mechanism like client side or server side input or output validation is used in an application then the probability of attack on this application is reduced to some extent but there are some chances of getting the application compromised by attacker. It means that the security approaches will reduce the attackers success rate but then also after some tries attacker will succeed in compromising the application i.e. these security approaches will provide safeguard to the application from basic attacks & malicious inputs but attacker with advanced skill-set can breach this security level hence even after taking security measures there is a vulnerability which could be exploited by hackers, These vulnerabilities are called customized vulnerability. In other words vulnerability customization can be defined as a technique to hide specific vulnerabilities inside the project such that it is difficult for someone to detect the loophole either manually or using some tools.

VI. Conclusion

As we have seen earlier, lots of tools are available for detecting the vulnerabilities. For our testing or demonstration four to five of them are used. Comparative study of all the tools used, and seeing if any of them is better. Also, we can see the importance of manual approach in detecting xss and sql attacks. Most of the commercial frameworks/tools are costly, one can use this thesis to make an open source framework/tool to detect the vulnerabilities. We will be making a tool with the help of shell scripting language python to detect one of the vulnerability.

The scripts for making tool to detect below vulnerability will be made using python language:

- Click jacking
- Sql Injection in GET parameter

References

[1] OWASP testing guide Volume 3, https://www.owasp.org/index.php/Testing-Guide-Foreword

[2] The OWASP Testing Framework, https://www.owasp.org/index.php/The-OWASPTesting-Framework

[3] Puspendra Kumar, R.K. Pateriya ”A Survey on SQL Injection Attacks, Detection and Prevention Techniques”, IEEE-20180

[4] An Example to show the harm of SQL injection attack, http://www.securityfocus.com/news/6194

[5] Erik Couture (2013), ”Web Application Injection Vulnerabilities: A Web Applications Security Nemesis?”, SANS reading room

[6] RSnake, ”SQL injection cheat sheet filter evasion”, OWASP.org

[7] SQL Injection Cookbook - Oracle, https://www.owasp.org/index.php/SQL-Injection- Cookbook-Oracle

[8] Florian Kerschbaum, ”Simple Cross-Site Attack Prevention”, IEEE

[9] RSnake, ”XSS attacks cheat sheet filter evasion”, OWASP.org

[10] What is Reflected cross site scripting Vulnerability?,

[11] https://nilminus.wordpress.com/web-application-penetration-testing/data-inputvalidation/

[12] cross-site-scripting/reflected-cross-site-scripting/

[13] What is stored cross site scripting Vulnerability?,

[14] https://nilminus.wordpress.com/web-application-penetration-testing/data-inputvalidation/

[15] Steven Cook (2003), ”A Web Developers Guide to Cross-Site Scripting”, SANS reading room

[16] DOMbased XSS vulnerability, http://www/webappsec.org/projects/articles/071105.html

[17] All about Security risks in robots.txt File, http://www.robotstxt.org/

[18] Issac Museong Kim (2012), ”Penetration testing of a web application using dangerous http methods”, SANS reading room

[19] What is Authentication: Wikipedia?, http://en.wikipedia.org/wiki/Authentication

[20] CAPTCHA: As a Solution to DOS, http://en.wikipedia.org/wiki/CAPTCHA

[21] ACUNETIX WEB VULNERABILITY SCANNER, A REAL WORLD REVIEW, MSI::Labs, May, 2006

[22] Cross Site Request Forgery Vulnerability, https://www.owasp.org/index.php/CrossSite-Request-Forgery-%28CSRF%29

[23] OWASP Zed Attack Proxy Project documentation, https://www.owasp.org/index.php/OWASP-Zed-Attack-Proxy-Project

Combating click-jacking with x-frame options,

[24] http://blogs.msdn.com/b/ieinternals/archive/2010/03/30/combating-clickjackingwith-x-frame-options.aspx?Redirected=true

[25] Netsparker Web Vulnerability Scanner Product Brochure, https://www.netsparker.com/s/NetsparkerProductBrochure.pdf

[26] ESAPI: Enterprise security application programme interface,

[27] https://www.owasp.org/index.php/Category:OWASP-Enterprise-Security-API

[28] Working with Secure Flags in Cookies, https://www.owasp.org/index.php/SecureFlag

[29] How to perform testing for XML injection, https://www.owasp.org/index.php/Testing-for-XML-Injection(OTG-INPVAL-008)

[30] Martin Johns, Bjorn Engelmann, and Joachim Posegga (2008), ”XSSDS: Server-side Detection of Cross-site Scripting Attacks”, Annual Computer Security Application Conference

[31] WAN Min, LIU Kun (2012), ”An Improved Eliminating SQL Injection Attacks Based Regular Expressions Matching”, International Conference on Control Engineering and Communication Technology.

[32] Mansour A. Alharbi (20010), ”Writing a Penetration Testing Report”, SANS reading room

[33] Ivano Alessandro Elia, Jos Fonseca, Marco Vieira (2010), ”Comparing SQL Injection Detection Tools Using Attack Injection: An Experimental Study”., 2010 21st International Symposium on Software Reliability Engineering.