AN OVERVIEW OF PENETRATION TESTING

[1]Aileen G. Bacudio, [1]Xiaohong Yuan, [2]Bei-Tseng Bill Chu, [1]Monique Jones

[1]Dept. of Computer Science, North Carolina A&T State University, Greensboro, North Carolina, USA
agbacudi@ncat.edu; xhyuan@ncat.edu; mfjones@ncat.edu
[2]Dept. of Software and Information Systems, University of North Carolina at Charlotte, Charlotte, North Carolina, USA
billchu@uncc.edu

ABSTRACT

Penetration testing is a series of activities undertaken to identify and exploit security vulnerabilities. It helps confirm the effectiveness or ineffectiveness of the security measures that have been implemented. This paper provides an overview of penetration testing. It discusses the benefits, the strategies and the methodology of conducting penetration testing. The methodology of penetration testing includes three phases: test preparation, test and test analysis. The test phase involves the following steps: information gathering, vulnerability analysis, and vulnerability exploit. This paper further illustrates how to apply this methodology to conduct penetration testing on two example web applications.

KEYWORDS

Security Testing, Vulnerability Assessment, Penetration Testing, Web Application Penetration Testing

1. INTRODUCTION

Security is one of the major issues of information systems. The growing connectivity of computers through the internet, the increasing extensibility of systems, and the unbridled growth of the size and complexity of systems have made software security a bigger problem now than in the past [1]. Furthermore, it is a business imperative to adequately protect an organization's information assets by following a comprehensive, and structured approach to provide protection from the risks an organization might face [2]. In an attempt to solve the security problem and comply with the mandated security regulations, security experts have developed various security assurance methods including proof of correctness, layered design, software engineering environments and penetration testing.

Penetration testing is a comprehensive method to test the complete, integrated, operational, and trusted computing base that consists of hardware, software and people [1]. The process involves an active analysis of the system for any potential vulnerabilities, including poor or improper system configuration, hardware and software flaws, and operational weaknesses in the process or technical countermeasures [3].

Penetration testing is different from security functional testing. The latter demonstrates the correct behavior of the system's security controls while penetration testing determines the difficulty for someone to penetrate an organization's security controls against unauthorized access to its information and information systems. It is done by simulating an unauthorized user attacking the system using either automated tools or manual method or a combination of both.

This paper provides an overview of penetration testing. It discusses the benefits of penetration testing, penetration testing strategies and types, as well as the methodology for penetration testing. It further illustrates the process of conducting web application penetration testing using two example web applications: TuneStore and BOG. These web applications were developed in

the Department of Software and Information Systems, University of North Carolina at Charlotte to demonstrate vulnerabilities in web applications.

This paper is organized as follows: Section 2 describes the benefits of penetration testing, Section 3 discusses penetration testing strategies and types, Section 4 describes the methodology for penetration testing. Section 5 illustrates the process of web application penetration testing using two example applications. Section 6 provides further discussion on penetration testing and Section 7 concludes the paper.

2. WHY PENETRATION TESTING

The main goal of vulnerability assessment is to identify security vulnerabilities under controlled circumstances so they can be eliminated before unauthorized users exploit them. Computing system professionals use penetration testing to address problems inherent in vulnerability assessment, focusing on high-severity vulnerabilities. Penetration testing is a valued assurance assessment tool that benefits both business and its operations.

2.1 Benefits of Penetration Testing from Business Perspective

From a business perspective, penetration testing helps safeguard the organization against failure through preventing financial loss; proving due diligence and compliance to industry regulators, customers and shareholders; preserving corporate image; and rationalize information security investment [4].

Organizations spend millions of dollars to recover from a security breach due to notification costs, remediation efforts, decreased productivity and lost revenue. The CSI study estimates recovery efforts alone to be $167,713.00 per incident [5]. Penetration testing can identify and address risks before security breaches occur, thus preventing financial loss caused by security breaches.

Industry has mandated regulatory requirements for computing systems. Non-compliance can result in the organization's receiving heavy fines, imprisonment, or ultimate failure [4]. Penetration testing, as a proactive service, provides unassailable information that helps the organization to meet the auditing or compliance aspects of regulations.

A single incident of compromised client data can be devastating. Loss of consumer confidence and business reputation can put the entire organization at risk. Penetration testing creates heightened awareness of security's importance at all levels of the organization. This helps the organization avoid security incidents that threaten its corporate image, put its reputation at risk and impact customer loyalty.

Penetration testing evaluates the effectiveness of existing security products and provides the supporting arguments for future investment or upgrade of security technologies. It provides a "proof of issue" and a solid case for proposal of investment to senior management [5].

2.2 Benefits of Penetration Testing from Operational Perspective

From operational perspective, penetration testing helps shape information security strategy through quick and accurate identification of vulnerabilities; proactive elimination of identified risks; implementation of corrective measures; and enhancement of IT knowledge [4].

Penetration testing provides detailed information on actual, exploitable security threats if it is encompassed into an organization's security doctrine and processes [5]. This will help the organization to identify quickly and accurately real and potential vulnerabilities.

By providing the information required to effectively and efficiently isolate and prioritize vulnerabilities, penetration testing can assist the organization fine-tune and test configuration changes or patches to proactively eliminate identified risks.

Penetration testing can also help an organization quantify the impacts and likelihood of the vulnerabilities. This will allow the organization to prioritize and implement corrective measures for reported known vulnerabilities.

The process of carrying out a penetration test entails a lot of time, effort and knowledge to deal with the complexities of the test space. Penetration testing will therefore enhance the knowledge and skill level of anyone involved in the process.

3. WHAT IS INVOLVED IN PENETRATION TESTING

There are two areas that should be considered when determining the scope and objectives of penetration testing: testing strategies and testing types used [2].

3.1 Penetration Testing Strategies

Based on the amount of information available to the tester, there are three penetration-testing strategies: black box, white box and gray box.

In black box penetration testing, the testers have no knowledge about the test target. They have to figure out the loopholes of the system on their own from scratch. This is similar to the blind test strategy in [2], which simulates the actions and procedures of a real attacker who has no information concerning the test target.

On the contrary, in white box penetration testing, the testers are provided with all the necessary information about the test target. This strategy is referred to in [2] as targeted testing where the testing team and the organization work together to do the test, with all the information provided to the tester prior to test.

Partial disclosure of information about the test target leads to gray box penetration testing. Testers need to gather further information before conducting the test.

Based on the specific objectives to be achieved, there are two penetration testing strategies which include external and internal testing.

External testing refers to any attacks on the test target using procedures performed from outside the organization that owns the test target [2]. The objective of external testing is to find out if an outside attacker can get in and how far he can get in once he has gained access.

Internal testing is performed from within the organization that owns the test target. The strategy is useful for estimating how much damage a disgruntled employee could cause. Internal testing is centred on understanding what could happen if the test target was successfully penetrated by an authorized user with standard access privileges.

3.2 Penetration Testing Types

There are three areas to test in penetration testing: the physical structure of the system, the logical structure of the system, and the response or workflow of the system [6]. These three areas define the scope and the types of penetration testing which are network, application, and social engineering.

Network penetration testing is an ethical and safe way to identify security gaps or flaws in the design, implementation or operation of the organization's network. The testers perform analysis and exploits to assess whether modems, remote access devices and maintenance connections can be used to penetrate the test target.

Application penetration testing is an attack simulation intended to expose the effectiveness of an application's security controls by highlighting risks posed by actual exploitable vulnerabilities [7]. Although organizations use firewall and monitoring systems to protect information, security can still be compromised since traffic can be allowed to pass through the firewall.

Social engineering preys on human interaction to obtain or compromise information about an organization and its computer systems [8]. It is used to determine the level of security awareness among the employees in the organization that owns the target system. This is useful to test the ability of the organization to prevent unauthorized access to its information and information systems [2]. Thus, this is a test focused on the workflow of the organization.

4. HOW TO CONDUCT PENETRATION TESTING

Penetration testing is not merely the serial execution of automated tools and generation of technical reports as it is frequently viewed. It should provide a clear and concise direction on how to secure an organization's information and information systems from real world attacks. One critical factor in the success of penetration testing is its underlying methodology. A systematic and scientific approach should be used to successfully document a test and create reports that are aimed at different levels of management within an organization. It should not be restrictive to enable the tester to fully explore his intuitions.

Generally, penetration testing has three phases: test preparation, test, and test analysis as shown in Figure 1.

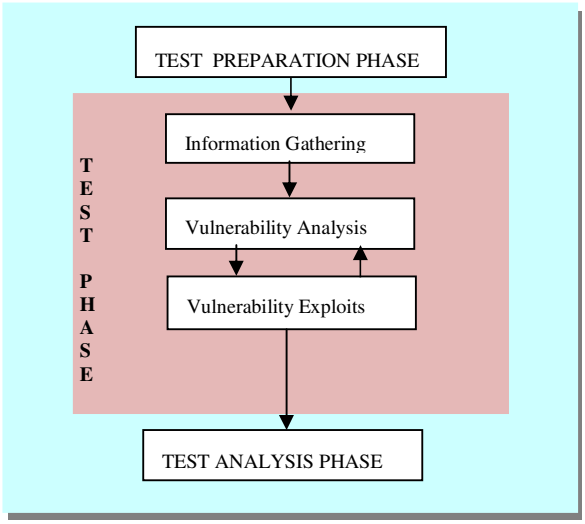

Figure 1. Penetration Testing Methodology

All the necessary documents for the test are organized and finalized during the test preparation phase. The testers and the organization meet to decide the scope, objectives, timing, and duration of the test. Issues such as information leakages and downtime are resolved and put into legal agreement document. Other legal agreements that are deemed necessary are concluded and signed during this phase.

The bulk of the penetration testing process is done during the test phase. A variety of automated tools can be used in this phase. Table 1 lists some of these tools. This phase involves the following steps: information gathering, vulnerability analysis, and vulnerability exploits.

Table 1. Penetration Testing Tools.

Name of Tool	Specific Purpose	Cost	Portability
Nmap [10]	• network scanning • port scanning • OS detection	free	Linux, Windows, FreeBSD, OpenBSD, Solaris, IRIX, Mac OS X, HP-UX, NetBSD, Sun OS, Amiga
Hping [11]	• port scanning • remote OS fingerprinting	free	Linux, FreeBSD, NetBSD, OpenBSD, Solaris, Mac OS X, Windows
SuperScan [12]	• detect open TCP/UDP ports determine which services are running on those ports • run queries like whois, ping, and hostname lookups	free	Windows 2000/XP/Vista/7
Xprobe2 [13]	• remote active OS fingerprinting • TCP fingerprinting • port scanning	free	Linux
p0f [14]	• OS fingerprinting • firewall detection	free	Linux, FreeBSD, NetBSD, OpenBSD, Mac OS X, Solaris, AIX, Windows
Httprint [15]	• web server fingerprinting • detect web enabled devices (e.g., wireless access points, routers, switches, modems) which do not have a server banner string • SSL detection	free	Linux, Mac OS X, FreeBSD, Win32 (command line and GUI)
Nessus [16]	• detect vulnerabilities that allow remote cracker to control or access sensitive data • detect misconfiguration, default password, and denial of service	free for personal edition, non-enterprise edition	Mac OS X, Linux, FreeBSD, Oracle Solaris, Windows, Apple
Shadow Security Scanner [17]	• detect network vulnerabilities, audit proxy and LDAP servers	free trial version	Windows but scan servers built on any platform
Iss Scanner [18]	• detect network vulnerabilities	free trial version	Windows 2000 Professional with SP4, Windows Server 2003 Standard with SO1, Windows XP Professional with SP1a
GFI LANguard [19]	• detect network vulnerabilities	free trial version	Windows Server 2003/2008, Windows 2000 Professional, Windows 7 Ultimate/ Vista Business/XP Professional/Small Business Sever 2000/2003/2008
Brutus [20]	• Telnet, ftp, and http password cracker	free	Windows 9x/NT/2000
Metasploit Framework[21, 22]	• develop and execute exploit code against a remote target • test vulnerability of computer systems	free	All versions of Unix and Windows

The information gathering step requires that the tester scan the physical and logical areas of the test target and identify all pertinent information needed in the vulnerability analysis phase. Depending on the information gathered or provided by the organization, the tester then analyzes the vulnerabilities that exist within the target's network, host and application. The tester may opt to use the manual method to do this step but automated tools also exist to help the tester. The last step allows the tester to find exploits for the vulnerabilities found in the previous steps. When exploits do not lead to what is intended, for example, root access, then further analysis should be done. This is represented by the loop between vulnerability analysis and vulnerability exploit phases.

The results of the test are thoroughly investigated during the test analysis phase. These results are provided to the organization so it must be comprehensive and systematic. Preparation of a mitigation plan is important in penetration testing [9]. It is therefore mandatory to include a mitigation plan section in the analysis report.

5. WEB APPLICATION PENETRATION TESTING

Penetration testing is the systematic probing of a system that could be any combination of applications, hosts, or networks [6]. The general methodology of penetration testing described in Section 4 is a useful process to uncover and resolve security weaknesses of applications, especially web applications.

This section illustrates the process of penetration testing using two example web applications, TuneStore and BOG. It should be noted that the areas tested here consist of only a small subset of the areas that should be tested. Therefore, the penetration testing process may be more comprehensive and may take more time and effort when performed on more complicated web applications.

It is assumed that the test preparation phase has been completed prior to the test phase. There are three steps involved in the test phase: information gathering, vulnerability analysis, and vulnerability exploit. These phases are called discovery phase, vulnerability phase and attack simulation phase in [15]. The test phase is followed by test analysis phase.

5.1 Information-Gathering Step

In this step, the testers collect as much information about the web application as possible and gain understanding of its logic. The deeper the testers understand the test target, the more successful the penetration testing will be [23]. The information gathered will be used to create a knowledge base to act upon in later steps. The testers should gather all information even if it seems useless and unrelated since no one knows at the outset what bits of information are needed.

This step can be carried out in many different ways: by using public tools such as search engines; using scanners; sending simple HTTP requests or specially crafted requests [24]; or walking through the application. The testers can identify the purposes of the application by browsing them. They can fingerprint the web server, spot the applications on both the client and the server side, and determine the content and functionality manually or using an automated tool. Table 2 lists some common tools that can be used in web application penetration testing. In what follows, we describe how to conduct information gathering on two example applications, BOG and TuneStore.

Table 2. Web Application Penetration Testing Tools

Name of Tool	**Specific Purpose**	**Cost**	**Portability**
Nmap [10]	Find web server	Free	Linux, Microsoft Windows, FreeBSD, OpenBSD, Solaris, IRIX, Mac OS X, HP-UX, NetBSD, Sun OS, Amiga
Fiddler [25]	A web debugging proxy	Free	Windows XP / 2K3 / Vista / 2K8 / Win7 / Win8 Microsoft .NET Framework **v2.0** or later
Nikto [26]	Identifies web server type, version, add-on and other interesting files, performs quick analysis of web server and applications and reports back common server and software misconfigurations, default files and programs, insecure files and programs and outdated servers and programs	Open source	Windows, Mac OSX, Linux and Unix (including RedHat, Solaris, Debian, Knoppix, etc)
WebScarab [27]	Interceptor, identifies new URLs on the test target, session ID analyzer, parameter fuzzer	Free	All platforms that support Java in any version not older than 1.4
w3af [28]	Vulnerability tester, interceptor, fuzzer	Open source	Linux, Windows XP, Windows Vista, Open BSD and any platforms that support Python
Firefox Extension – Firebug [29]	Inline editing, for breaking forms, messing with JavaScript, making rougue sites and man-in-the-middle components	Free	Any platforms that supports Firefox
Firefox Extension – TestGen4Web [30]	Record and playback clicks during surfing	Free	Any platforms that support Firefox 1.5 beta
Cenzic Hailstorm [31]	Web vulnerability scanner	1-week trial free	Windows 7 Pro or Windows XP Professional with Service Pack 3
Core Impact [32]	Identify, validate and exploit vulnerabilities, test web application for XSS, Reflective XSS, SQL Injection, Blind SQL Injection, Remote File Inclusion for PHP applications	Commercial	Windows 7, Windows Vista, Windows Server 2008 SP2, Windows Server 2003 SP2
Nessus 4 [6]	Detect vulnerabilities that allow remote cracker to control or access sensitive data, misconfiguration, default password, and denial of service	Free for personal edition non-enterprise edition	Mac OS X, Linux, FreeBSD, Oracle Solaris, Windows, Apple
Metasploit Framework[21 ,22]	Develop and execute exploit code against a remote target machine, test vulnerability of computer systems	Free	All versions of Unix and Windows

5.1.1 Brief Description of the BOG and TuneStore Applications

BOG and TuneStore are two web applications developed by Dr. Bei-Tseng Bill Chu's project team to demonstrate web application vulnerabilities. The main functionality of the BOG and TuneStore applications can be captured by thoroughly browsing the web applications.

BOG is an online blogging site that allows user to post or edit textual blogs. The users can gain access to the site as a blogger or as a visitor. Both blogger and visitor can post and edit a blog. The main page of the application includes a list of posted blogs which can be clicked to reveal its contents.

TuneStore is an online store of songs. A user needs to register and log in to gain access to the store. The application provides the following options for registered users: "add balance", "friends", "profile", "CDs" and "log out". The "add balance" option allows the user to add money to his account by entering his credit card number and the amount of money to be added. The "friends" function allows the user to add a friend to his list of friends. The "CDs" function allows the user to view the CDs he purchased. The "profile" function allows the user to change his password. The main page of the application includes images representing the tunes that can be purchased in the store. The "comment" function allows users to view and submit remarks regarding the tune.

5.1.2 Web Server Fingerprinting

Web server identification is critical when analyzing vulnerabilities [23]. Knowing the version and type of the running web server helps the testers to determine what vulnerabilities they need to uncover and what exploits they are going to perform.

Figure 2 shows the HTTP headers of TuneStore using the automated tool, Fiddler. The miscellaneous branch under the response headers indicates that the server is Apache-Coyote 1.1. Using Fiddler we found that BOG also uses Apache-Coyote 1.1 web server.

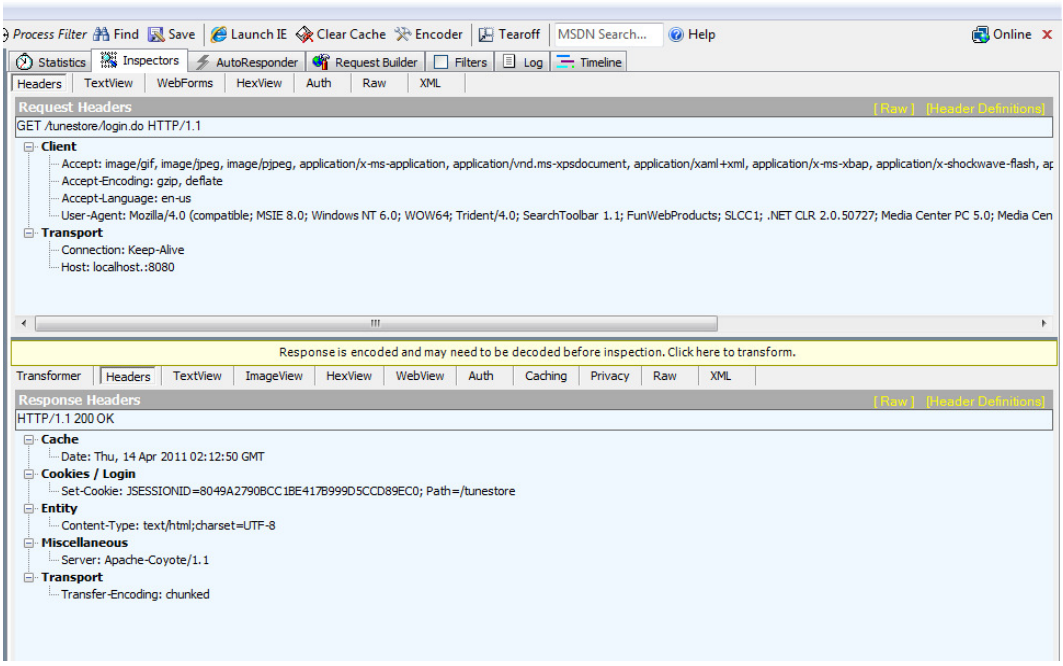

Figure 2. HTTP Headers of TuneStore Using Fiddler

5.1.3 Application Fingerprinting

The goals of application fingerprinting are to identify the file types that the target system handles and the resources that exist on the target. The file types often disclose the programming language and the platform used to implement the functions, which are useful in the discovery of vulnerabilities.

These information may be found in the HTTP headers and in error messages. Figure 3 illustrates another HTTP header of TuneStore using the Fiddler tool. The JSESSIONID field from the cookies branch under the request header indicates that the application uses the Java platform. The content-type under the entity branch points out that Cascading Style Sheets are used.

Figure 4 shows an error message generated by the BOG application. This message shows that Microsoft Active Server Pages was used to implement the system. Yet another error message of the BOG application (Figure 5) indicates the use of My Standard Query Language (MySQL) for the backend database of the application.

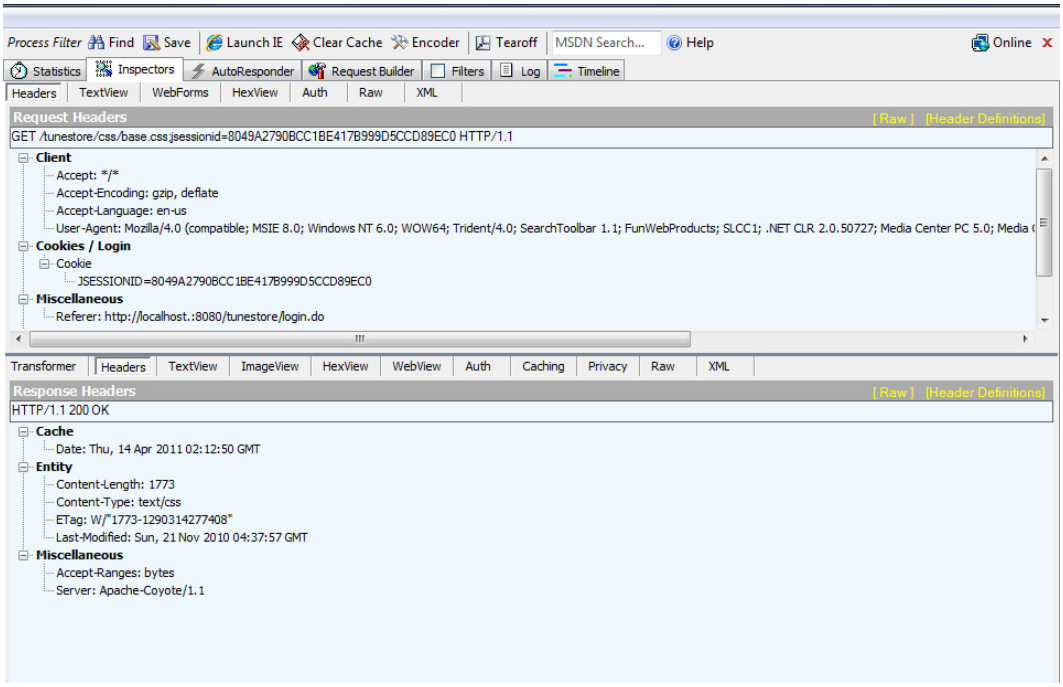

Figure 3. HTTP Headers of TuneStore using Fiddler

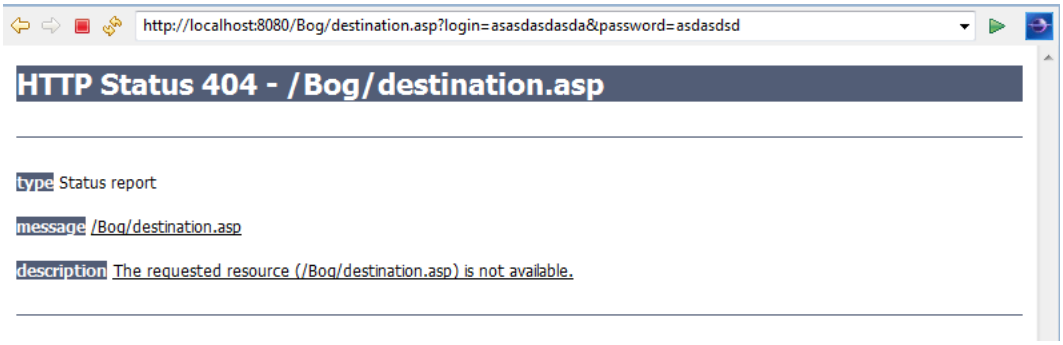

Figure 4. Error Message Generated by BOG

Figure 5. Error Message Generated by Bog with Invalid Password

5.1.4 Enumerating Content and Functionality

The functionality of a web application can be identified by walking through it starting from the main initial page, following every link and navigating through all multistage functions. Figure 6 illustrates the site map of TuneStore using the manual browsing method. Using WebScarab's Spider feature (Figure 7), further content can be found such as images, css, and js directories.

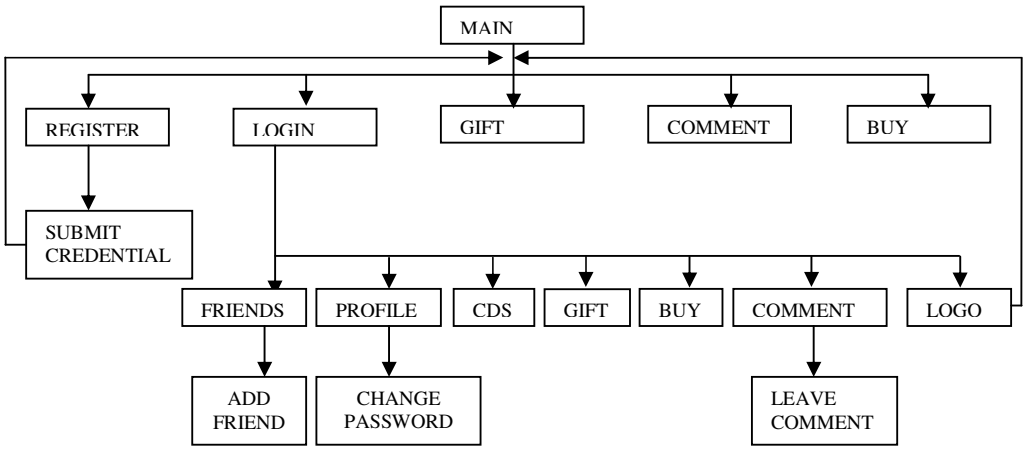

Figure 6. Site Map of TuneStore Using Manual Browsing

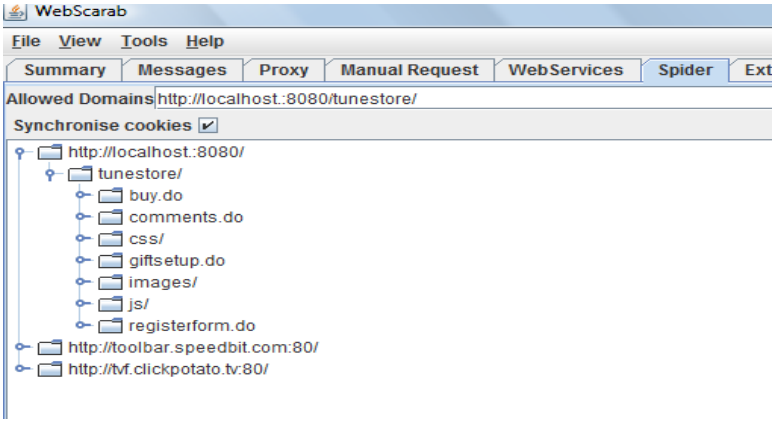

Figure 7. Screenshots of Using WebScarab Spider Feature

5.2 Vulnerability Analysis Step

Using the knowledge collected from the information gathering step, the testers then scan the vulnerabilities that exist in the web application. The testers can conduct testing on configuration management, business logic, authentication, session management, authorization, data validation, denial of service, and web services [24]. In this step, web server vulnerabilities, authentication mechanism vulnerabilities, input-based vulnerabilities and function-specific vulnerabilities are examined.

5.2.1Web Server Vulnerabilities

When examining web server vulnerabilities, the testers need to test for default credentials, default content, dangerous HTTP methods, proxy functionality, and virtual hosting misconfiguration [26]. Both TuneStore and BOG applications used the Apache web server. The scanning for vulnerabilities within the Apache web server is not discussed here.

5.2.2 Authentication Mechanism Vulnerabilities

It is necessary to test for any design and implementation flaws in a web application's authentication mechanisms. Password quality, username enumeration, password guessing, "remember me", account recovery, username uniqueness, credential predictability, fail-open logic, and multistage processes should be tested [26].

BOG and TuneStore were tested for password quality by entering very short, very long, blank and common dictionary words or names on the username and password fields. BOG application allowed any user to log in. For example, it allowed an unauthorized user to login using blank user name and password. However, the TuneStore application issued an error when an unregistered user tries to log in. Neither application issued any verbose error message during the testing for authentication mechanism vulnerabilities. Neither the Uniform Resource Locator (URL) bar during transmission of credentials gave out information about the presence of vulnerabilities. There are no "remember me" and account recovery functions that could possibly give more clues about the vulnerabilities of the applications.

5.2.3 Input-Based Vulnerabilities

Data-driven web applications are prone to input-based vulnerabilities, which can be uncovered by testing for Standard Query Language (SQL injection) and Cross Site Scripting (XSS).

Testing for SQL Injection Vulnerabilities

SQL injection allows attackers to inject SQL statements to read and modify data stored within the database. BOG was tested for SQL injection vulnerabilities using the manual method. Different SQL injection attack strings were tried on the username and password fields but the application responded in the same way as when a blank username and password are used. One string though, "***' OR "="***", gave anomalous messages when tried on the "Blogger login" and "Visitor registration" functions (Figure 8 and Figure 9). One can infer from the messages that "username" and "password" may have been used as fieldnames in the database table. This information is vital for the vulnerability exploit step.

When using manual method to test for SQL injection vulnerabilities on the TuneStore applicaiton, it was found that when the string, "***anyuser' OR 'x'='x***", was used as the username and password, the system allowed the user the same level of access as a registered user "aileen" (Figure 10).

The Fuzzer feature of WebScarab was used to automate the discovery of SQL injection vulnerabilities on the TuneStore application. Figure 11 shows the SQL strings that were used for the test and Figure 12 shows the results. Each entry in the output corresponds to an attack string. The attack string that corresponds to HTTP response with status code 200 indicates the attack string successfully exploited the system.

Figure 8. Error Message of BOG Blogger Login Function (Username and Password is ***'OR "='***)

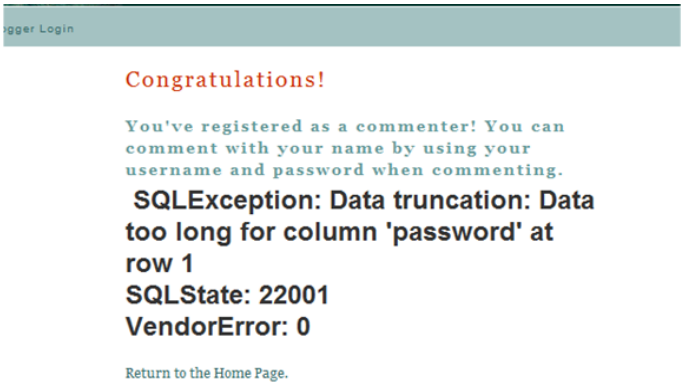

Figure 9. Error Message of BOG Visitor Login Function (Username and Password is ***'OR "='***)

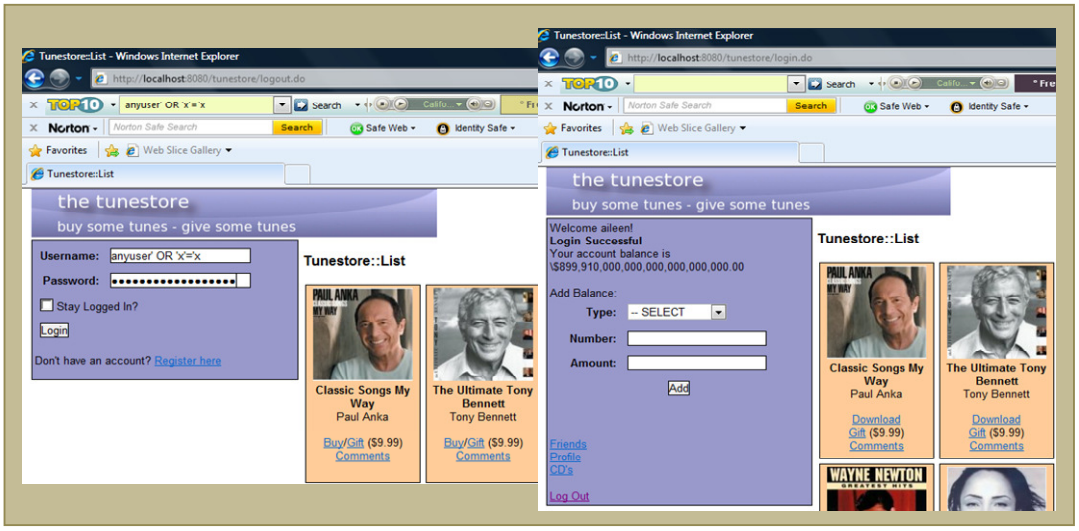

Figure 10. TuneStore after an Unregistered User Logged in (Username and Password is ***anyuser' OR 'x'='x***)

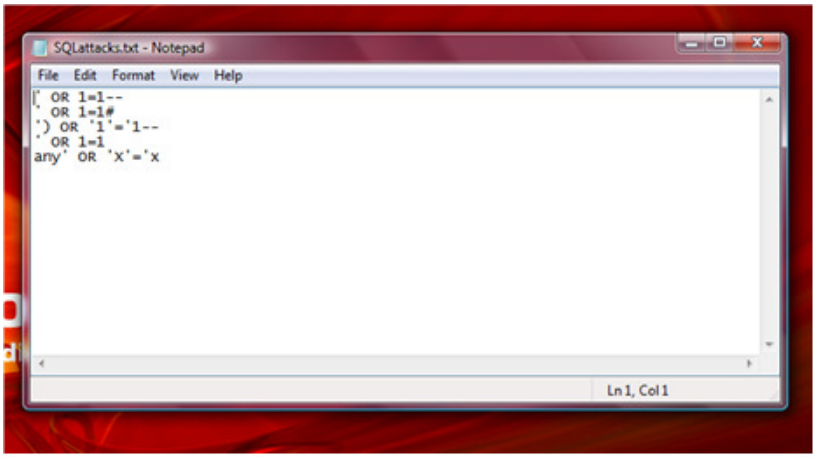

Figure 11. Attack Strings Used to Fuzz TuneStore for SQL Injection

Parameters

Location	Name	Type	Value	Priority	
Cookie	JSESSIONID	STRING	9F0BDB9C18916A940F1CBC...	0	
Body	username	STRING	aileen	0	SQL
Body	password	STRING	bacudio	0	SQL

Total Requests : 5
Current Request : 5
Sources

ID	Date	Method	Host	Path	Status	Paramete...	
50	2010/12/03...	POST	http://localhost.:8080	/tunestore/login.do	500 Internal Server Error		Fuzzer
51	2010/12/03...	POST	http://localhost.:8080	/tunestore/login.do	200 OK		Fuzzer
53	2010/12/03...	POST	http://localhost.:8080	/tunestore/login.do	500 Internal Server Error		Fuzzer
52	2010/12/03...	POST	http://localhost.:8080	/tunestore/login.do	500 Internal Server Error		Fuzzer
54	2010/12/03...	POST	http://localhost.:8080	/tunestore/login.do	200 OK		Fuzzer

Figure 12. Results of Fuzzing TuneStore for SQL Injection

Testing for XSS Vulnerabilities

XSS is a common web application vulnerability that enables attackers to inject client-side script into web pages viewed by other users. XSS often occurs in Web applications that allow users to submit comments. For the BOG application, the functions "new post" and "read comment" were used to test for XSS vulnerabilities. The string ***"<script> alert ("This application is vulnerable to XSS")</script>***" was entered in the Text Box of creating a post (Figure 13)**.** The application accepted the post. When the blog was revisited, an alert window as shown in Figure 14 appeared which indicates the presence of XSS vulnerabilities

The Fuzzer feature of WebScarab was used to test whether TuneStore is vulnerable to XSS. The user name and password fields were tested. It was discovered that the attack string "***<IMG SRC=javascript.alert("Attacked4")>***" got HTTP response with status code 200. This shows that the application is vulnerable to XSS.

Figure 13. XSS Vulnerability Testing on BOG

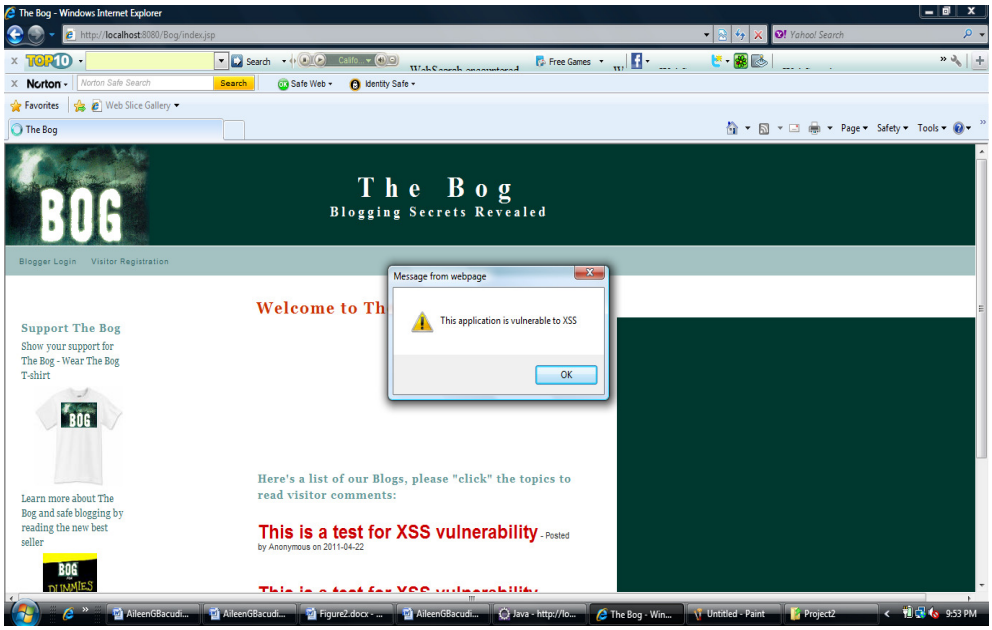

Figure 14. The result of the XSS exploit on BOG

5.2.4 Function-Specific Vulnerabilities

In addition to input-based vulnerabilities, there is a range of vulnerabilities that normally manifest themselves in particular kinds of functionality such as integer vulnerabilities [26]. Since the TuneStore application has functions that involve monetary computations, it is necessary to test the application's responses to very long input on fields that require integer input, such as the "add balance" function. TuneStore did not issue any error even if the money added is more than a billion dollars. Figure 10 shows a user with a huge amount of balance. This behaviour is surprising and anomalous that it is further investigated in the step that follows.

5.3 Vulnerability Exploit Step

After the vulnerability analysis step, the testers should have a good idea of the areas that will be targeted for exploits. With the list of vulnerabilities on hand, the two applications were then exploited.

Exploiting Vulnerabilities in BOG Application. It was discovered that the application has vulnerabilities in its authentication mechanisms, and is vulnerable to SQL injection and XSS. The absence of rigorous validation mechanisms allowed any user to gain entry into the system. An exploit to this vulnerability is to flood the site with content that compromise the security of the system such as adding blogs that damage other users' reputation, link to scammer's site, or any potentially dangerous site.

Knowing that the application is vulnerable to SQL injection backed up with the information gathered from the error messages, the different functions of the application were explored further with the objective of getting more ideas to exploit the system. This scenario illustrates the loop between the vulnerability analysis and vulnerability exploit steps.

Figure 15 shows the content of the site when the URL is http://localhost:8080/Bog/individual.jsp?id=54. By persistently manipulating the URL using UNION operation, a couple of pages that contain vital information were encountered (Figure 16 and Figure 17). This information seems to be the usernames and passwords of legitimate users. These usernames and passwords can be used to gain access to the system.

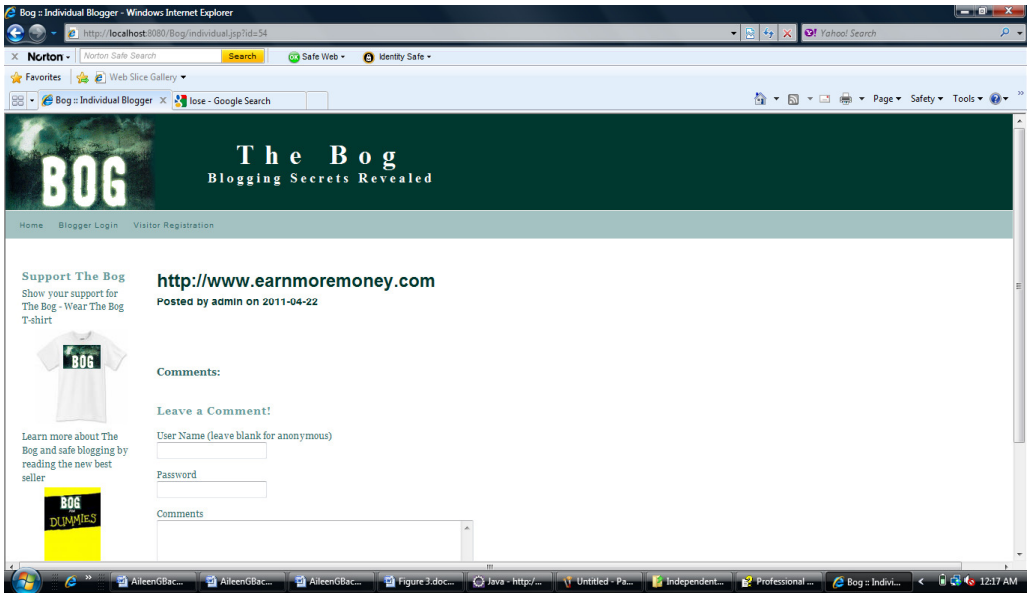

Figure 15. Content of the Site with the URL http://localhost:8080/Bog/individual.jsp?id=54

During the test for XSS vulnerabilities, the application was exploited by the malicious message posted on the site prior to displaying the blog (Figure 14). This vulnerability can be exploited further by using a link rather than an alert message.

Another exploit of this vulnerability was to steal valuable data. This was done by forging the website and embedding content that lures the user to enter sensitive information. This is illustrated in Figure 18.

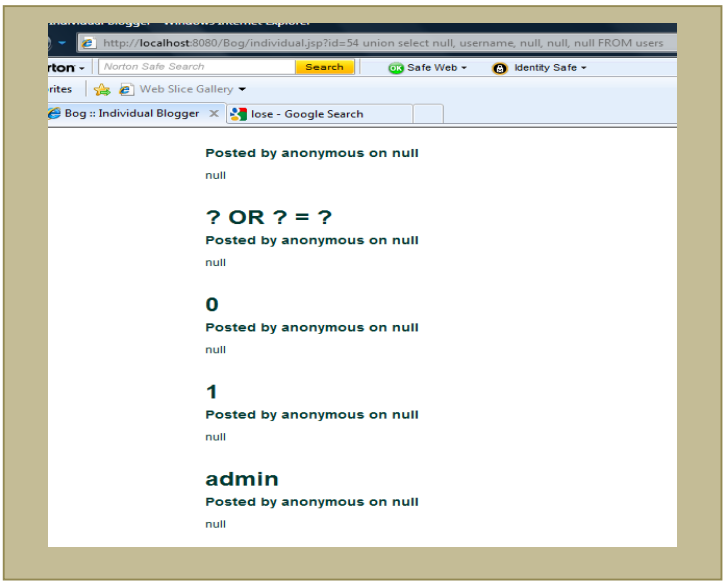

Figure 16. Manipulating URL to Discover User Names

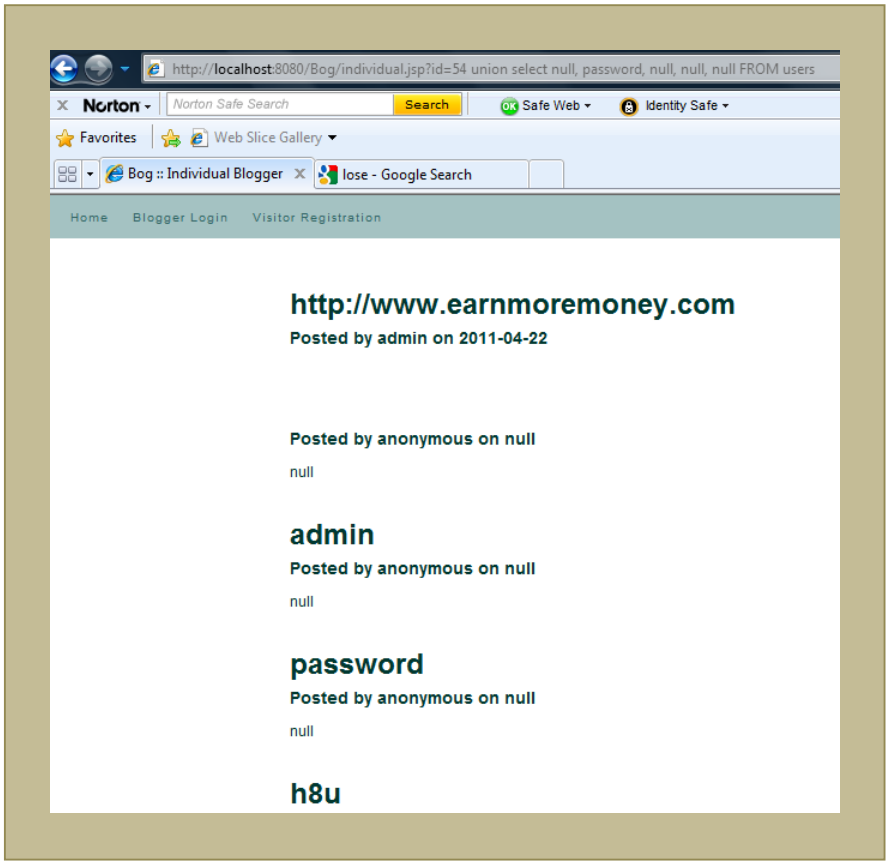

Figure 17. Manipulating URL to Discover Passwords

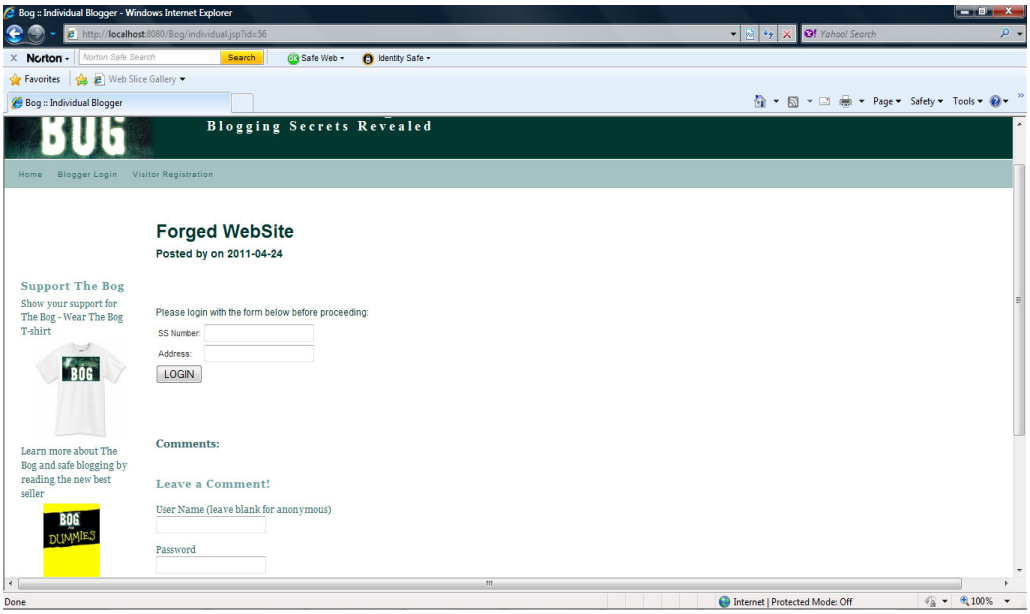

Figure 18. Forged Site Asking the User to Enter Sensitive Information

Exploiting Vulnerabilities in TuneStore Application. Through fuzz testing, it is shown that TuneStore has SQL injection vulnerability. It was observed that, using the SQL injection attack string, "***anyuser' OR 'x'='x***", as input to user name and password, the attacker was logged in as the first user in the database (Figure 10). By input attack string "***anyuser' OR 'x'='x'* and username <>'aileen**", the attacker is logged in as the next user in the database. Using similar approach, the attacker can log in as any user in the database.

An anomalous behaviour of the application is accepting a very large amount in the "add balance" function. Computation with very large value could give incorrect results. Therefore it was necessary to explore all functions that involve computation and find a possible exploit.

After adding a very large value to the current balance, it was found that the balance is not reduced after a song is purchased (Figure 19). This means the attacker is able to purchase a song for free.

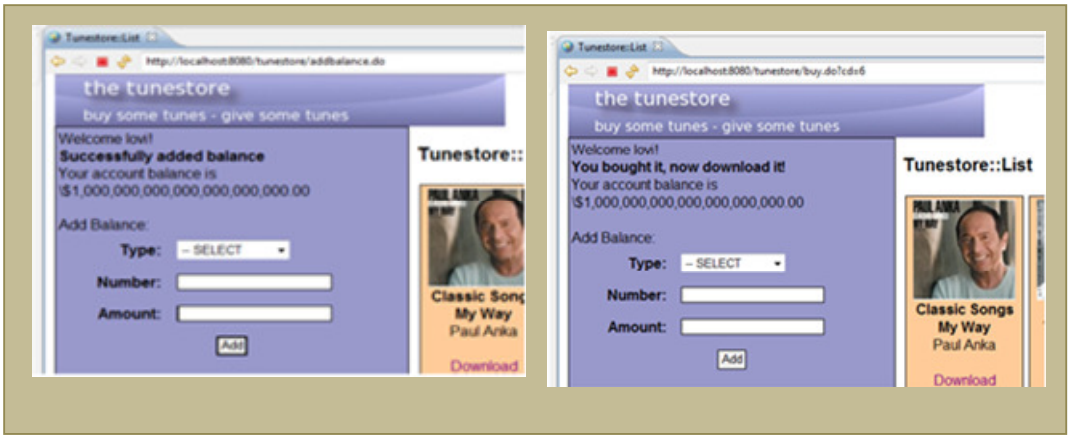

Figure 19. Balance Unchanged after Purchasing a Song

5.4 Test Analysis Phase

This phase is the interface of the results, the testers and the target entity [23]. It is important that the target entity is aware of typical attacker modus operandi, techniques and tools attackers rely on, exploits they use, and any needless exposure of data the target is suffering from.

The reactive and corrective steps that need to be taken to address the problems should be included in this phase. Considering the findings for BOG and TuneStore, one of the remediation strategies could be to sanitize the user's input data before placing it within a SQL query. By requiring that the username and password being passed to the database do not contain any bad characters, the systems could be protected against SQL injection attacks.

6. DISCUSSION

Penetration testing can be an efficient and cost-effective strategy to protect the organization's systems against attacks. If done properly, it helps the organization identify the internal practices that give rise to vulnerabilities and other sources of vulnerabilities. The identified sources enable the organization to remove the vulnerabilities, properly direct the system's security efforts, pressure vendors to improve their products, improve its internal business security practices and prove to customers, shareholders and regulatory agencies that it is making a good faith effort to properly protect critical business data.

Selecting a penetration team is a pertinent factor towards the success of penetration testing process. In evaluating the team, consideration should be given to their qualifications, experience and knowledge, reputation in the e-business community, access to, and use of, state-of-the art tools. A rule of thumb is to eliminate a team who provides the systems to be tested.

Penetration testing cannot be expected to identify all possible security vulnerabilities since it is but just one aspect of testing. The organization should develop an overall security testing strategy that is tailored to its threat models and security policies.

Finally, penetration testing should never be regarded as a one-off-service. It is conducted at a point of time. System changes, threats emerge, business strategies advance, and hacker tools evolve. Fixing or patching the vulnerability detected does not mean an end to your security worries and nightmare, it is just the beginning of a never ending cycle. In addition, a penetration test does not offer any guarantee of absolute security, it is just a measurement of security posture [27].

7. RELATED WORK

The penetration testing process described by Neumann [28] includes four steps: (1) understanding the system, (2) hypothesizing the flaws, (3) testing to confirm or reject hypotheses, (4) extending successful tests to yield other hypotheses. These steps correspond to the methodology described in this paper. Step (1) corresponds to the information gathering step, Steps (2) and (3) correspond to the vulnerability analysis step, and step (4) corresponds to the vulnerability exploits step. Pfleeger et al. [29] described an organized methodology for planning a suite of penetration tests, which includes three steps: (1) identifying sensitive objects, (2) determining points of vulnerability for those objects, (3) testing vulnerabilities to determine the adequacy of controls. This methodology provides a systematic approach for hypothesizing the flaws in Neumann's model, and can be used in the vulnerability analysis step in our methodology. The vulnerability analysis step illustrated in this paper is based on different categories of vulnerabilities. Bishop [30] emphasizes that penetration testing should consider the goal and limits of testing. These should be considered during the test preparation phase in our methodology. Arkin et al. [31] propose that penetration testing should be structured according to perceived risk, and at feature, component or unit, and system level. These could be considered during the test phase of our methodology.

8. CONCLUSION

Penetration testing is a comprehensive method to identify the vulnerabilities in a system. It offers benefits such as prevention of financial loss; compliance to industry regulators, customers and shareholders; preserving corporate image; proactive elimination of identified risks.

The testers can choose from black box, white box, and gray box testing depending on the amount of information available to the user. The testers can also choose from internal and external testing, depending on the specific objectives to be achieved. There are three types of penetration testing: network, application and social engineering.

This paper describes a three-phase methodology consisting of test preparation, test, and test analysis phase. The test phase is done in three steps: information gathering, vulnerability analysis, and vulnerability exploit. This phase can be done manually or using automated tools. This penetration testing process was illustrated on the web applications, BOG and TuneStore.

The testers should follow a comprehensive format to present the test results. One of the most important parts of the test analysis phase is the preparation of remediation which includes all necessary corrective measures for the identified vulnerabilities. The final report needs to have enough detail and substance to allow those doing remediation to simulate and follow the attack pattern and respective findings [23].

REFERENCES

[1] McGraw, G. (2006). *Software Security: Building Security In*, Adison Wesley Professional.

[2] The Canadian Institute of Chartered Accountants Information Technology Advisory Committee, (2003) "Using an Ethical hacking Technique to Assess Information Security Risk", Toronto Canada. http://www.cica.ca/research-and-guidance/documents/it-advisory-committee/item12038.pdf, accessed on Nov. 23, 2011.

[3] Mohanty, D. "Demystifying Penetration Testing HackingSpirits,"

http://www.infosecwriters.com/text_resources/pdf/pen_test2.pdf, accessed on Nov. 23, 2011.

[4] "Penetraion Testing Guide", http://www.penetration-testing.com/

[5] iVolution Security Technologies, "Benefits of Penetration Testing,"

http://www.ivolutionsecurity.com/pen_testing/benefits.php, accessed on Nov. 23, 2011.

[6] Shewmaker, J. (2008). "Introduction to Penetration Testing," http://www.dts.ca.gov/pdf/news_events/SANS_Institute-Introduction_to_Network_Penetration_Testing.pdf, accessed on Nov. 23, 2011.

[7] "Application Penetration Testing," https://www.trustwave.com/apppentest.php, accessed on Nov. 23, 2011.

[8] Mullins, M. (2005) "Choose the Best Penetration Testing Method for your Company," http://www.techrepublic.com/article/choose-the-best-penetration-testing-method-for-your-company/5755555, accessed on Nov. 23, 2011.

[9] Saindane, M. "Penetration Testing – A Systematic Approach," http://www.infosecwriters.com/text_resources/pdf/PenTest_MSaindane.pdf, accessed on Nov. 23, 2011.

[10] "Nmap – Free Security Scanner for Network Explorer, http://nmap.org/, accessed on Nov. 23, 2011.

[11] Sanfilippo, S. "Hping – Active Network Security Tool," http://www.hping.org/, accessed on Nov. 23, 2011.

[12] Superscan, http://www.mcafee.com/us/downloads/free-tools/superscan.aspx, accessed on Nov. 23, 2011.

[13] Xprobe2, http://www.net-security.org/software.php?id=231, accessed on Nov. 23, 2011.

[14] P0f, http://www.net-security.org/software.php?id=164, accessed on Nov. 23, 2011.

[15] Httprint, http://net-square.com/httprint/, accessed on Nov. 23, 2011.

[16] Nessus, http://www.tenable.com/products/nessus, accessed on Nov. 23, 2011.

[17] Shadow Security Scanner, http://www.safety-lab.com/en/download.htm, accessed on Nov. 23, 2011.

[18] Iss Scanner, http://shareme.com/showtop/freeware/iss-scanner.html, accessed on Nov. 23, 2011.

[19] GFI LAN guard, http://www.gfi.com/network-security-vulnerability-scanner, accessed on Nov. 23, 2011.

[20] Brutus, http://download.cnet.com/Brutus/3000-2344_4-10455770.html, accessed on Nov. 23, 2011.

[21] MetaSploit, http://www.metasploit.com/, accessed on Nov. 23, 2011.

[22] Skoudis, E. "Powerful Payloads: The Evolution of Exploit Frameworks," (2005). http://searchsecurity.techtarget.com/news/1135581/Powerful-payloads-The-evolution-of-exploit-frameworks, accessed on Nov. 23, 2011.

[23] Andreu, A. (2006). *Professional Pen Testing for Web Applications*. Wrox publisher, 1st edition.

[24] OWASP. "Web Application Penetration Testing," http://www.owasp.org/index.php/Web_Application_Penetration_Testing, accessed on Nov. 23, 2011.

[25] Fiddler, http://www.fiddler2.com/fiddler2, accessed on Nov. 23, 2011.

[26] Stuttard, D. and Pinto, M. (2008) *The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws,*, Wiley. 1st edition.

[27] "White Paper on Penetration Testing," http://www.docstoc.com/docs/70280500/White-Paper-on-Penetration-Testing, accessed on Nov. 23, 2011.

[28] Neumann, P. (1977) "Computer System Security Evaluation," *Proceedings of AFIPS 1977 Natl. Computer Conf.*, Vol. 46, pp. 1087-1095.

[29] Pfleeger, C. P., Pfleeger, S. L., and Theofanos, M. F. (1989) "A Methodology for Penetration Testing," *Computers &Security*, 8(1989) pp. 613-620.

[30] Bishop, M. (2007) "About Penetration Testing," *IEEE Security & Privacy*, November/December 2007, pp. 84-87.

[31] Arkin, B., Stender, S., and McGraw, G. "Software Penetration Testing," *IEEE Security & Privacy*, January/Feburary 2005, pp. 32-35.

Authors

Aileen G. Bacudio

Aileen Guira Bacudio received her Masters degree in computer science from North Carolina A&T State University, USA. She worked as a programmer, a project officer of SPI Technology, Inc., and an instructor of computer science at the University of the Philippines Los Banos (UPLB) Laguna.

Xiaohong Yuan

Xiaohong Yuan is an associate professor in the Department of Computer Science, North Carolina A&T State University, USA. Her research interests include computer science education, information assurance, software engineering and visualization.

Bei-Tseng Bill Chu

Bei-Tseng Bill Chu is a Professor and the Chair of the Department of Software and Information Systems in University of North Carolina at Charlotte, USA. His research interests include secure software development, enterprise integration and security, and information technology education.

Monique F. Jones

Monique Jones is a Senior Computer Science Student at North Carolina A&T State University, USA.