1 2 9 0

UNIVERSIDADE D
COIMBRA

Cyber Security Assessment and Management

Security Assessment and Improvement

Francisco Catarino Mendes - 2019222823
Department of Informatics Engineering
University of Coimbra

Introduction

Nowadays, to ensure a system is as secure as possible, security assessments and improvements must be frequently done. The objective of this practical assignment is to understand the process, advantages, and disadvantages of using automated tools in tasks of security assessment, as well as applying attack surface analysis.

This assignment is composed of three parts. Before starting the process, one had to study the setup given and think about tools and methods to use in order to accomplish the objectives. The first part corresponds to the first security assessment, where the system is assessed without modifying, updating, or improving it. Then, there comes the security improvement phase, where the output of each step of the previous phase is analyzed and from that, a set of security measures is gathered with the aim of improving security. Finally, the first step is repeated, which is called a security reassessment, to determine if the solutions implemented in step 2 were effective or not.

0 - Information Gathering - Tools used

After studying what has to be done and configuring the web server, this was the main list of tools considered useful to reach the goal of this project:

1. Nmap - a tool for network discovery and security auditing;
2. Nikto - a web server scanner that performs comprehensive tests against web servers for multiple items;
3. Metasploit - an open-source project with resources for system hardening when used with good purposes;
4. CIS Debian Linux Benchmark - a set of guidelines and best practices for securing Debian Linux systems.

It must be said that these were not the only ones, as various command-line utility tools were also used during the realization of this assignment.

1 - Security Assessment

a) Measurement of simplified attack surface

The first step of the security assessment is the measurement of the attack surface, to discover possible entry points in the targeted system, which in this case is a web application. Because of efficiency purposes, the analysis was limited to the variables listed in Table 1:

Open sockets	Open RPC endpoints	Services	Enabled accounts
Open TPC ports	Open named pipes	Services running by default	Enabled accounts in admin group
Open UDP Ports	Dynamic web pages	Services running as SYSTEM	Guest accounts enabled

Table 1: Variables of the attack surface measured

For each variable, it was determined the number of avenues of attack, as well as a bias number for their potential danger of becoming pathways to the discovery of vulnerabilities and exploits. Then, by multiplying both scores, one gets the resulting bias-applied values. The sum of these values is what is called the **relative attack surface quotient.** The results of the measurement of the attack surface are shown in Table 2:

Avenues of Attack	Bias	Identified Avenues of Attack	Resulting Bias-Applied Values	Commands
Open Sockets	1	13	13	netstat -tuln netstat -tuln | grep -c 'LISTEN'
Open TCP ports	1	6	6	nmap -sT -p- -v <target IP>
Open UDP ports	1	6	6	sudo nmap -sU -p- -v <target IP>
Open RPC endpoints	0.9	8	7.2	rpcinfo -p 192.168.56.101 | grep -v 'program' | wc -l
Services	0.2	33	6.6	sudo service --status-all | wc -l
Services running by default	0.8	17	13.6	sudo service --status-all | grep +
Services running as SYSTEM	0.9	1	0.9	sudo service --status-all | grep 'SYSTEM|root'
Enabled accounts	0.7	33	23.1	getent passwd
Enabled accounts in admin group	0.9	1	0.9	getent group admin
Guest accounts enabled	0.9	1	0.9	getent passwd | grep -i 'guest|nobody'
Relative Attack Surface Quotient	**78,2**			

Table 2: Results of the measurement of the attack surface

It must be said that two of the variables were not tested successfully, those being the open-named pipes and the dynamic web pages. For the dynamic web pages, Burp Suite was being used, but an error was preventing it from analyzing the HTTP processes, even though it is believed that everything was configured accordingly.

b) Check the system for outdated software and famous vulnerabilities

Some checks were made to inquire if the system was running with outdated software, which would lead to many potential vulnerabilities and paths for exploitation. These were the software deemed outdated:

1. OpenSSH;
2. Apache2;
3. PHP;
4. MySQL.

Then, a search was made for vulnerabilities in the system, using Nikto and Metasploit. The results are as follows:

1. Outdated Apache version;
2. Outdated PHP version;
3. HTTP TRACE Method is enabled (XST - Cross Site Tracing) - can be used to steal HTTP cookies and authentication data in some configurations;
4. Revealing PHP errors - can reveal sensitive information about the server configuration;
5. Existence of robots.txt - could potentially reveal the presence of directories that the webmaster wishes to hide;
6. Directory Indexing - allows anyone to view the entire directory contents via a web browser;
7. Existence of ”interesting” directories and files for the tool;

c) Creation of an exploit for one of the vulnerabilities found in b)

From the list of vulnerabilities found in b), it was decided that, using Metasploit, the outdated Apache version was going to be exploited.

Firstly, the db_nmap utility was used to perform a scan. With the results, information became available regarding the identified system, where the Apache vulnerability is also present matching with section b).

Following that, a search for exploits in the Metasploit framework was done. 85 options appeared from the search result. The exploit selected was the Apache Continuum Arbitrary Command Execution exploit.

d) Analysis according to CIS Benchmarks

In this part, the focus will be on determining how compliant the system is with the recommendation points 3, 4, and 5 present in the CIS Debian Linux Benchmark.

Secure Boot Settings

1. Set User/Group Owner on bootloader config: Compliant;
2. Set Permissions on bootloader config: Not Compliant;
3. Set Boot Loader Password: Not Compliant;
4. Require Authentication for Single-User Mode: Compliant.

Additional Process Hardening

1. Restrict Core Dumps: Half Compliant;
2. Enable XD/NX Support on 32-bit x86 Systems: Compliant;
3. Enable Randomized Virtual Memory Region Placement: Compliant;
4. Disable Prelink: Compliant;
5. Activate AppArmor: Not Compliant.

OS Services

1. Ensure NIS is not installed: Compliant;
2. Ensure rsh server is not enabled: Compliant;
3. Ensure rsh client is not installed: Compliant;
4. Ensure talk server is not enabled: Compliant;
5. Ensure talk client is not installed: Compliant;
6. Ensure telnet server is not enabled: Compliant;
7. Ensure tftp-server is not enabled: Compliant;
8. Ensure xinetd is not enabled: Compliant;
9. Ensure chargen is not enabled: Compliant;
10. Ensure daytime is not enabled: Compliant;
11. Ensure echo is not enabled: Compliant;

12. Ensure discard is not enabled: Compliant;
13. Ensure time is not enabled: Compliant.

e) External analysis of the web application

As for external analysis, a manual investigation of the website corresponding to the application is needed. The tool-assisted search has already been done in b), where Nikto and Metasploit were used.

From the OWASP Testing Guide, one can grab many examples of how to test SQL Injection. If we insert the following username and password values *$username = 1' or '1' =* '1 and *$password = 1' or '1' = '1* in the targeted website's inputs for login page, an exploit is immediately made, as the login is successful, breaching the account of someone named Andreea Radu. Figures 1 and 2 show this process happening:

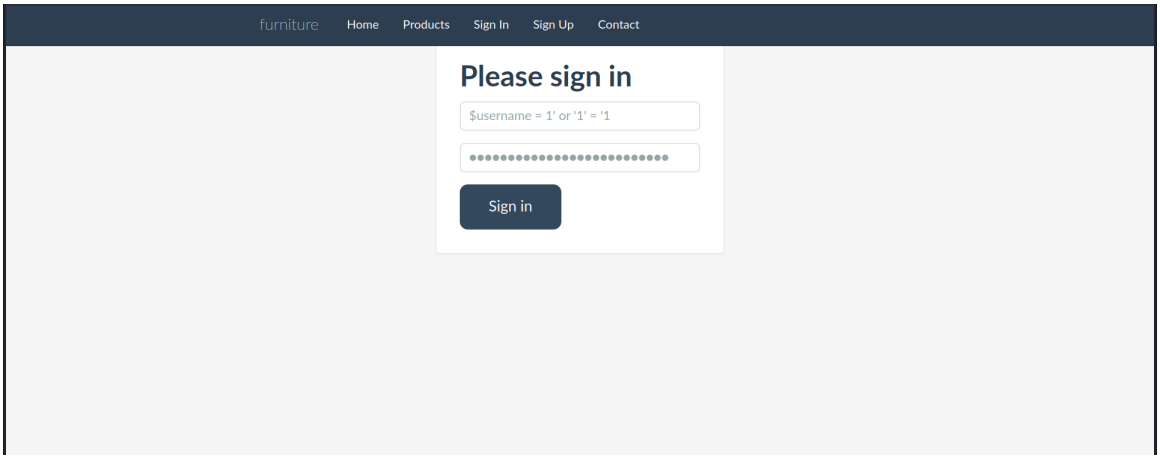

Figure 1: Login inputs

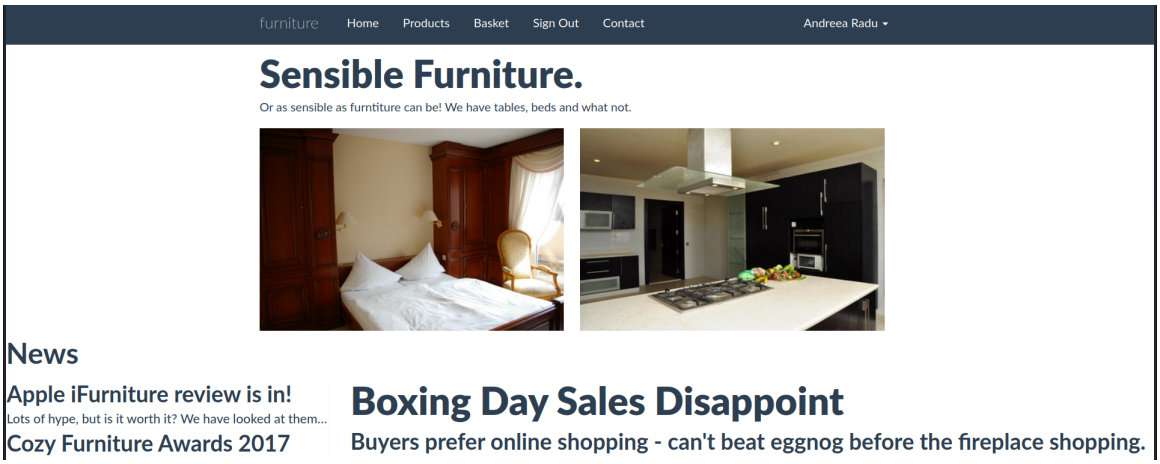

Figure 2: Unauthorized login successful

2 - Security Improvement

Regarding security improvements, with the help of all the information gathered until now, in order to fight and resolve the vulnerabilities the following actions were taken:

1. Updating Apache2;
2. Updating PHP;
3. Disabling the HTTP TRACE Method;
4. Configuring PHP to not display errors to users;
5. Securing robots.txt;
6. Disabling Directory Indexing;

As for attack avenues, these were the controls applied, shown in Table 3:

Controls	Commands
Closing Sockets	sudo fuser -k -n <udp/tcp> <port_number>
Closing TCP ports	sudo fuser -k -n <udp/tcp> <port_number>
Closing UDP ports	sudo fuser -k -n <udp/tcp> <port_number>
Closing RPC endpoints	sudo service rpcbind stop
Reviewing and stopping unnecessary services	sudo service [service_name] stop
Disabling any accounts that are not in use	sudo usermod -L [username] sudo passwd -l [username]
Removing unnecessary users from the admin group	sudo gpasswd -d [username] admin
Disabling guest accounts	sudo usermod -L guest sudo passwd -l guest

Table 3: Security controls on the attack surface variables

Also, regarding the settings that were not in compliance with the CISC Debian Linux Benchmark, all the remediation steps were executed with the aim of making the system more secure.

Other vulnerabilities like SQL Injection cannot be resolved without a white box approach. Code review is needed to assess the bad practices used during the programming of the web application.

3 - Security Reassessment

To finalize this assignment, point a) of the first section, Security Assessment, must be realized again. This is what is called a security reassessment, which analyzes whether the security controls implemented were successful or not.

Table 4 shows the results obtained:

Avenues of Attack	Bias	Identified Avenues of Attack	Resulting Bias-Applied Values	Comments
Open Sockets	1	0	0	All sockets were closed.
Open TCP ports	1	1	1	All TCP ports were closed, except the one used for the SSH session.
Open UDP ports	1	0	0	All UDP ports were closed.
Open RPC endpoints	0.9	0	0	All open RPC endpoints were closed.
Services	0.2	33	6.6	Services were left alone, as there was not enough info about which ones should be stopped.
Services running by default	0.8	17	13.6	Services were left alone, as there was not enough info about which ones should be stopped.
Services running as SYSTEM	0.9	1	0.9	Services were left alone, as there was not enough info about which ones should be stopped.
Enabled accounts	0.7	33	23.1	Accounts were left alone as well.
Enabled accounts in admin group	0.9	1	0.9	Accounts were left alone as well.
Guest accounts enabled	0.9	0	0	Guest account was removed.
Relative Attack Surface Quotient	**46,1**			

Table 2: Results of the re-measurement of the attack surface

As one can see, the controls resulted in a decrease of 32,1 in terms of relative attack surface quotient. This can be seen as the security methods put in place being somewhat efficient concerning the exposure of the attack surface.

Concerning the materials used, Nmap was considered to be a very strong tool, mainly in the detection phases, as it could show a lot of network concepts of the web server. Metasploit was also considered to be a very strong tool, but for the exploitation phase. The fact it has a database of exploits that can be done against systems is scary. It can also help in vulnerability detection, but it is not its strongest attribute, unlike Nikto, for example.

Finally, the CIS Debian Linux Benchmark was also pretty helpful, as it contained a handful of security measures and guidelines. It is a document composed of numerous security checks that if followed carefully, can turn any system's degree of safety higher.

Conclusion

This assignment provided insight into the importance of the security assessment of any system, but in particular of a web server. All the possible security strategies must be in place, and for that to happen these types of assessments need to be performed regularly (monthly perhaps). There are many tools to perform attacks on organizations' systems, and as we could see in this project the avenues of attack that can be discovered do not have a limit. Furthermore, this type of work must be complemented with white-box approaches, as, for example, SQL Injections can only be dealt with using a white-box approach to security.

As for the work done, some parts were lacking, like the exploitation section, and others could have been more detailed. More security mechanisms could have been brought up. More manual testing with the website could have been done. All in all, this work left room for improvement, but it is also believed that, in general, what was made has some degree of quality. For future work, complementing this project is definitely a goal, and like it was said before, this type of assignment must be constantly assembled and improved on to ensure the maximum security possible.