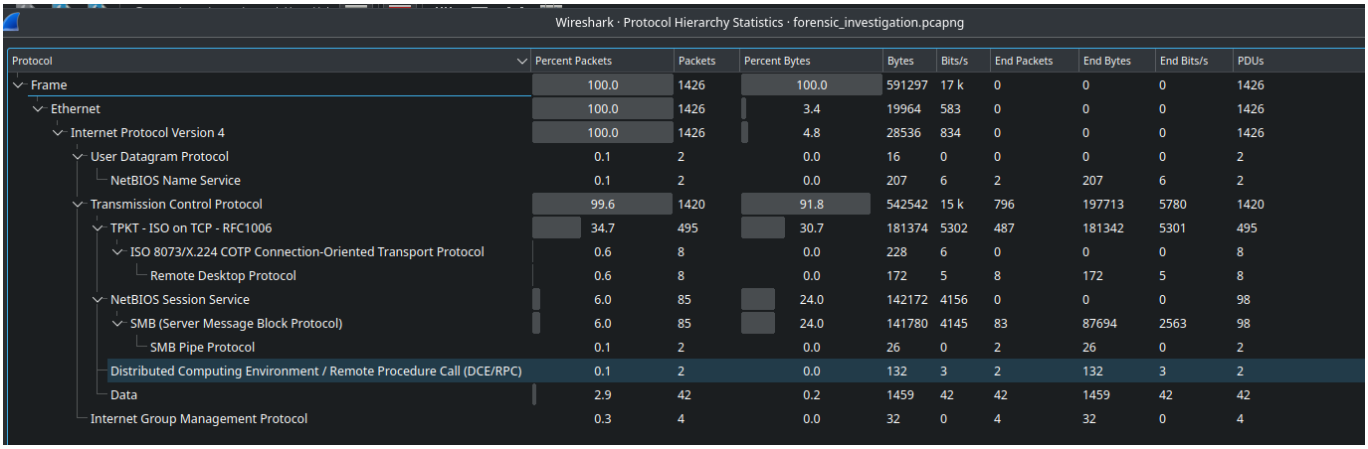
The Skye Can't Be Blue

Auteur: bWlrYQ

1. Analyse de la capture

Pour commencer, nous avons la certitude qu'un exploit a eu lieu, c'est l'énoncé qui nous l'indique. On peut plonger directement dedans. Notre meilleure amie la hiérarchie des protocoles est là pour nous aider à déméler le tout.



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	1426	100.0	591297	17 k	0	0	0	1426
Ethernet	100.0	1426	3.4	19964	583	0	0	0	1426
Internet Protocol Version 4	100.0	1426	4.8	28536	834	0	0	0	1426
User Datagram Protocol	0.1	2	0.0	16	0	0	0	0	2
NetBIOS Name Service	0.1	2	0.0	207	6	2	207	6	2
Transmission Control Protocol	99.6	1420	91.8	542542	15 k	796	197713	5780	1420
TPKT - ISO on TCP - RFC1006	34.7	495	30.7	181374	5302	487	181342	5301	495
ISO 8073/X.224 COTP Connection-Oriented Transport Protocol	0.6	8	0.0	228	6	0	0	0	8
Remote Desktop Protocol	0.6	8	0.0	172	5	8	172	5	8
NetBIOS Session Service	6.0	85	24.0	142172	4156	0	0	0	98
SMB (Server Message Block Protocol)	6.0	85	24.0	141780	4145	83	87694	2563	98
SMB Pipe Protocol	0.1	2	0.0	26	0	2	26	0	2
Distributed Computing Environment / Remote Procedure Call (DCE/RPC)	0.1	2	0.0	132	3	2	132	3	2
Data	2.9	42	0.2	1459	42	42	1459	42	42
Internet Group Management Protocol	0.3	4	0.0	32	0	4	32	0	4

Concrètement on fait face à 3 protocoles, TCP, SMB et TPKT (les deux derniers étant encapsulés dans TCP). Les failles d'exploitations étant généralement sur des protocoles applicatifs il y a deux possibilités quant au protocole utilisé pour compromettre la machine, SMB ou TPKT.

SMB est un protcole généralement lié à l'environnement Windows qui permet le partage de fichiers entre différents hôtes. TPKT est quant à lui utilisé pour la transmission de données, c'est un protocole d'encapsulation. On peut retrouver dans TPKT des paquets RDP, on est donc sûrement toujours dans un environnement Windows. Ils ont pu être générés dans le cadre d'une connexion distante à un hôte Windows.

Si on regarde la structure de la capture on remarque que TPKT n'apparaît qu'à partir de la fin de la capture ce qui signifie que la connexion RDP a dû être réalisée post-exploitation. Le protocole qui est donc responsable de la faille qui a permis l'exploitration est donc SMB, nous pouvons nous concentrer dessus.

2. Analyse de l'attaque

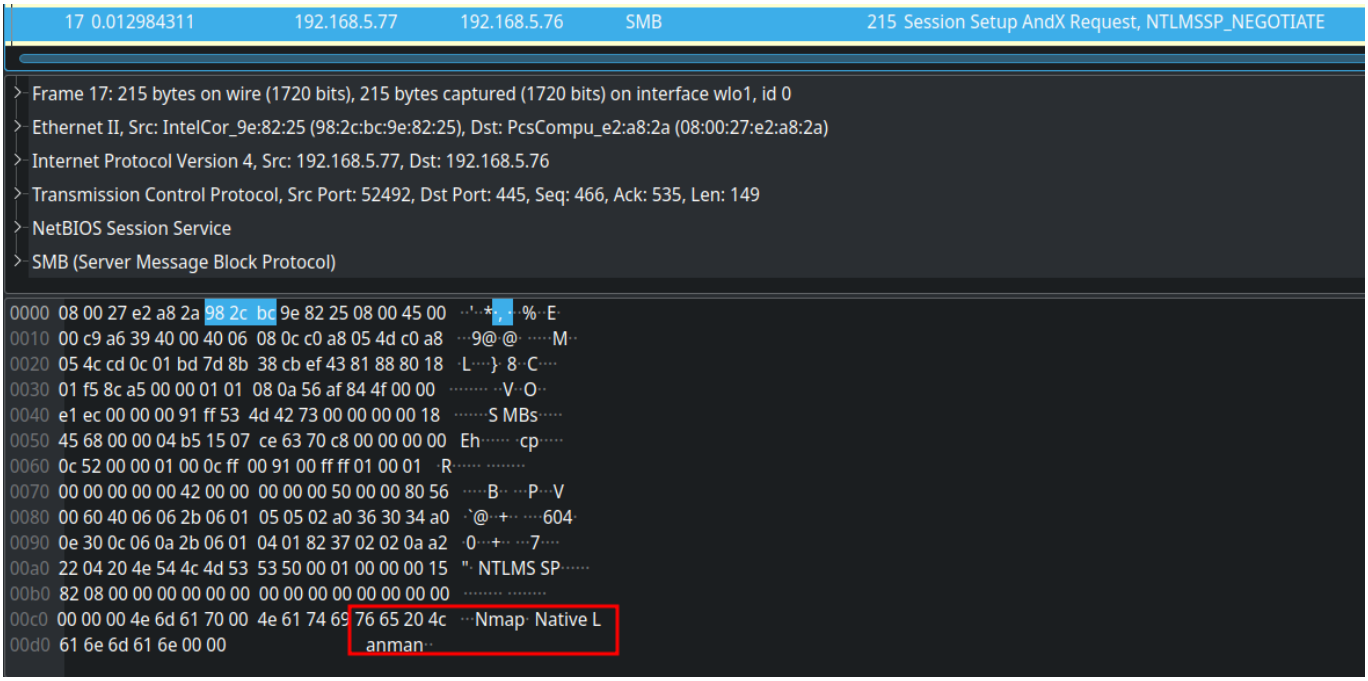
Essayons de séparer la capture en différentes parties, reconaissance, attaque, post-compromission.

Si on regarde le timer de la capture, les paquets 0 à 27 sont réalisés dans un très court laps de temps, sûrement une phase de reconnaissance pour permettre d'établir si un service SMB est en fonction ou non sur la machine (sûrement réalisé à l'aide d'un outil de mapping réseau comme nmap).

On peut voir la connexion au port 445 (port par défaut de SMB) avec une connexion TCP au tout début:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.5.77	192.168.5.76	TCP	74	52488 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1454343234 TSecr=0 WS=128
2	0.000286571	192.168.5.76	192.168.5.77	TCP	74	445 → 52488 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM TSval=57835 TSecr=1454343234
3	0.000304158	192.168.5.77	192.168.5.76	TCP	66	52488 → 445 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1454343234 TSecr=57835

Dans la trame 17, on peut confirmer l'usage de nmap avec la donnée dans la trame, une recherche google ajoute de l'eau à notre moulin.

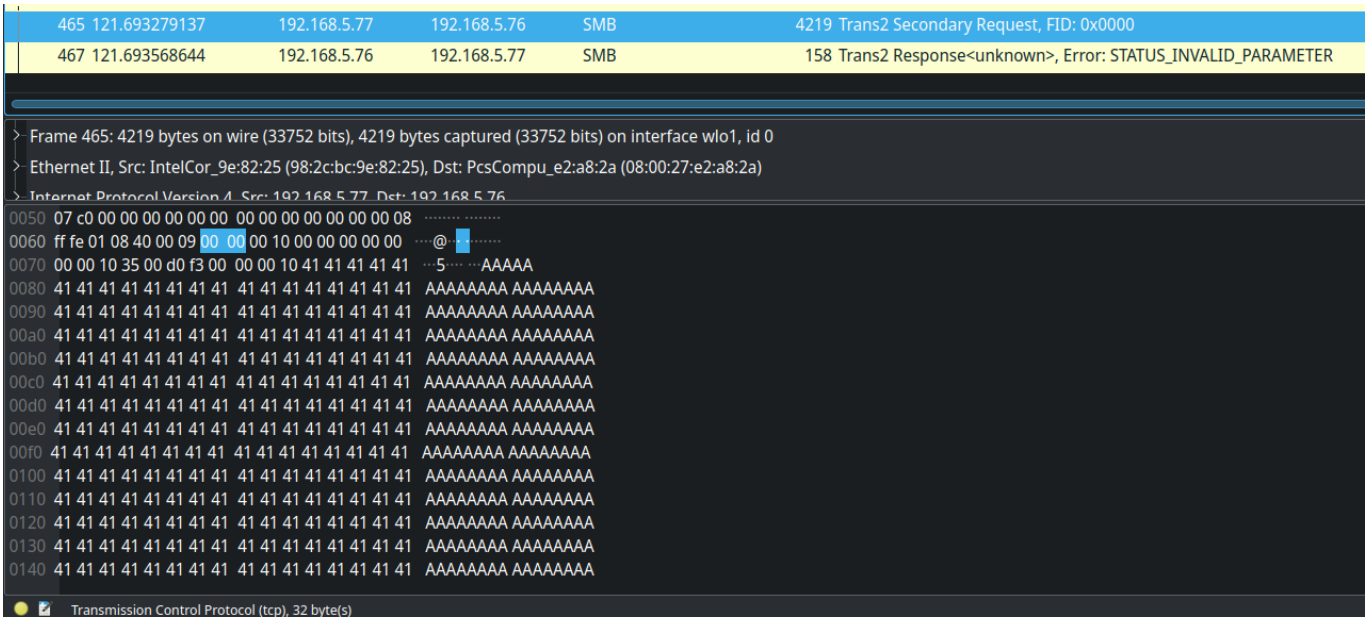


On en déduit donc que l'échange 0 à 27 est un scan du port 445 avec l'usage d'un script de détection pour une potentielle vulnérabilité liée à SMB. Pour localiser le momen de l'exploitation on peut utiliser le filtre wireshark `tcp.port==445`. On voit ensuite que l'exploitation s'est déroulée entre la trame 30 et la trame 607.

Disclaimer: pour cette partie, je ne me considère pas comme techniquement assez bon dans tout ce qui est lié à l'exploitation de ce genre de vulnérabilités concernant les protocoles applicatifs, je vais donc éviter d'entrer dans les détails afin de ne pas raconter des choses qui seraient fausses.

Pour limiter la recherche (car 570 trames, ça fait beaucoup), on va éliminer toutes les trames TCP et filtrer par `smb`, pour essayer de voir si des réponses "inattendues" surviennent dans l'échanges, quelque chose qui sort de l'ordinaire quant à la réaction produite par le serveur (le plus proche de la fin, la où on identifie la fin de l'exploitation et le début de la RCE).

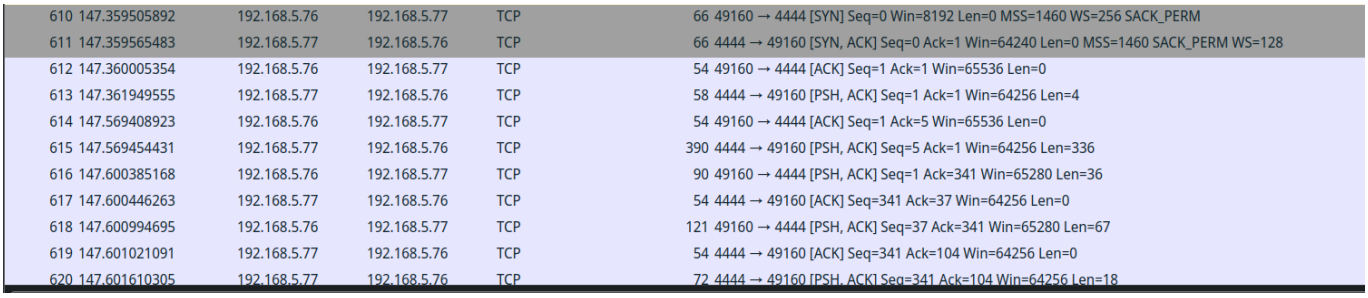
On peut voir que nos deux dernières trames sont une requête du client avec une taille de 4000 (assez anormal pour une requête), et une réponse avec une erreur "Trans 2 Response [...], Error: STATUS_INVALID_PARAMTER". Le payload envoyé par le client est une suite de 'AAAAA[...]AAAAA', payload courant lors d'une exploitation liée aux failles de type buffer overflow.



C'est donc à cet endroit que l'exploit vraissemblablement fonctioné, on peut étudier la suite de l'échange pour déterminer ce qu'il s'est passé.

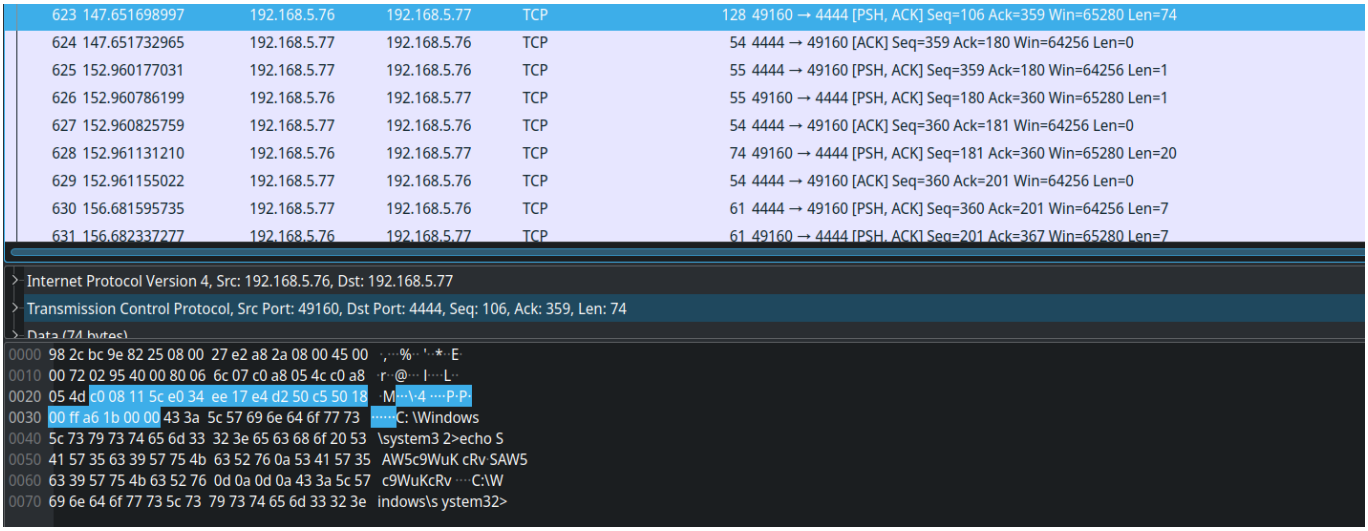
Reprise de "je sais à peu près de quoi je parle 😃"

A la trame 610 on peut voir un comportement particulier, le serveur SMB initie une connexion vers l'hôte qui exploitait une vulnérabilité sur le port 4444, sûrement le retour de l'exploitation avec un reverse shell.



610	147.359505892	192.168.5.76	192.168.5.77	TCP	66 49160 → 4444 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
611	147.359565483	192.168.5.77	192.168.5.76	TCP	66 4444 → 49160 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
612	147.360005354	192.168.5.76	192.168.5.77	TCP	54 49160 → 4444 [ACK] Seq=1 Ack=1 Win=65536 Len=0
613	147.361949555	192.168.5.77	192.168.5.76	TCP	58 4444 → 49160 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=4
614	147.569408923	192.168.5.76	192.168.5.77	TCP	54 49160 → 4444 [ACK] Seq=1 Ack=5 Win=65536 Len=0
615	147.569454431	192.168.5.77	192.168.5.76	TCP	390 4444 → 49160 [PSH, ACK] Seq=5 Ack=1 Win=64256 Len=336
616	147.600385168	192.168.5.76	192.168.5.77	TCP	90 49160 → 4444 [PSH, ACK] Seq=1 Ack=341 Win=65280 Len=36
617	147.600446263	192.168.5.77	192.168.5.76	TCP	54 4444 → 49160 [ACK] Seq=341 Ack=37 Win=64256 Len=0
618	147.600994695	192.168.5.76	192.168.5.77	TCP	121 49160 → 4444 [PSH, ACK] Seq=37 Ack=341 Win=65280 Len=67
619	147.601021091	192.168.5.77	192.168.5.76	TCP	54 4444 → 49160 [ACK] Seq=341 Ack=104 Win=64256 Len=0
620	147.601610305	192.168.5.77	192.168.5.76	TCP	72 4444 → 49160 [PSH, ACK] Seq=341 Ack=104 Win=64256 Len=18

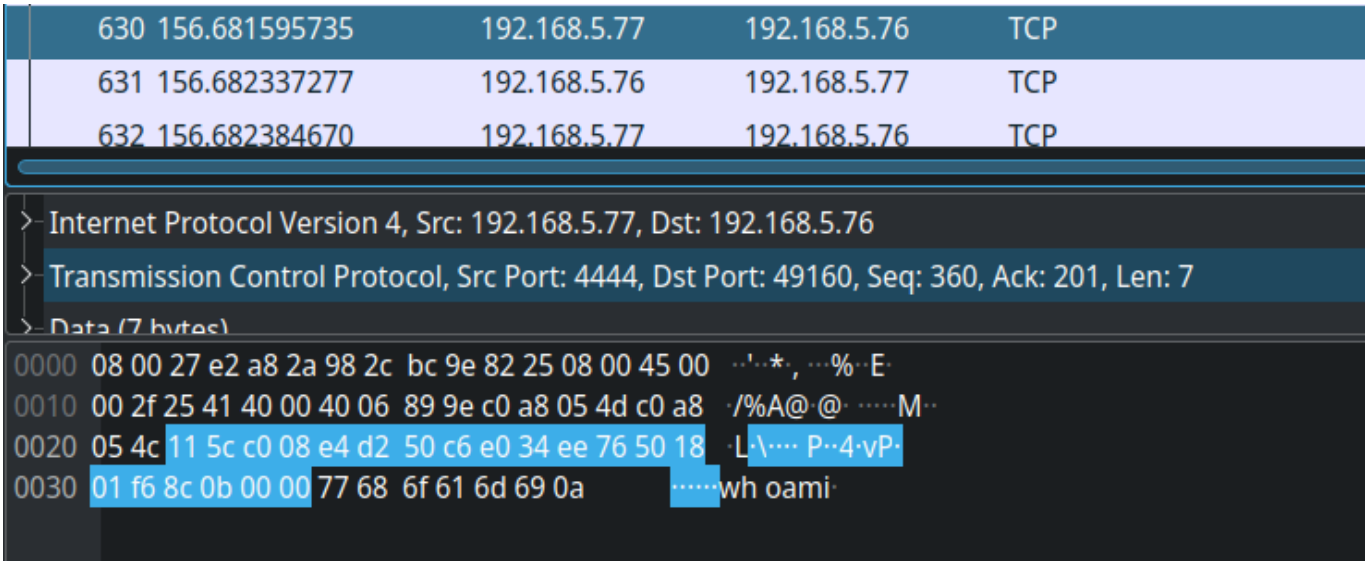
Quand on observe les données contenues dans les trames, on peut y voir quelque chose qui ressemble à un shell fourni pour la machine attaquante.



623	147.651698997	192.168.5.76	192.168.5.77	TCP	128 49160 → 4444 [PSH, ACK] Seq=106 Ack=359 Win=65280 Len=74
624	147.651732965	192.168.5.77	192.168.5.76	TCP	54 4444 → 49160 [ACK] Seq=359 Ack=180 Win=64256 Len=0
625	152.960177031	192.168.5.77	192.168.5.76	TCP	55 4444 → 49160 [PSH, ACK] Seq=359 Ack=180 Win=64256 Len=1
626	152.960786199	192.168.5.76	192.168.5.77	TCP	55 49160 → 4444 [PSH, ACK] Seq=180 Ack=360 Win=65280 Len=1
627	152.960825759	192.168.5.77	192.168.5.76	TCP	54 4444 → 49160 [ACK] Seq=360 Ack=181 Win=64256 Len=0
628	152.961131210	192.168.5.76	192.168.5.77	TCP	74 49160 → 4444 [PSH, ACK] Seq=181 Ack=360 Win=65280 Len=20
629	152.961155022	192.168.5.77	192.168.5.76	TCP	54 4444 → 49160 [ACK] Seq=360 Ack=201 Win=64256 Len=0
630	156.681595735	192.168.5.77	192.168.5.76	TCP	61 4444 → 49160 [PSH, ACK] Seq=360 Ack=201 Win=64256 Len=7
631	156.682337277	192.168.5.76	192.168.5.77	TCP	61 49160 → 4444 [PSH, ACK] Seq=201 Ack=367 Win=65280 Len=7

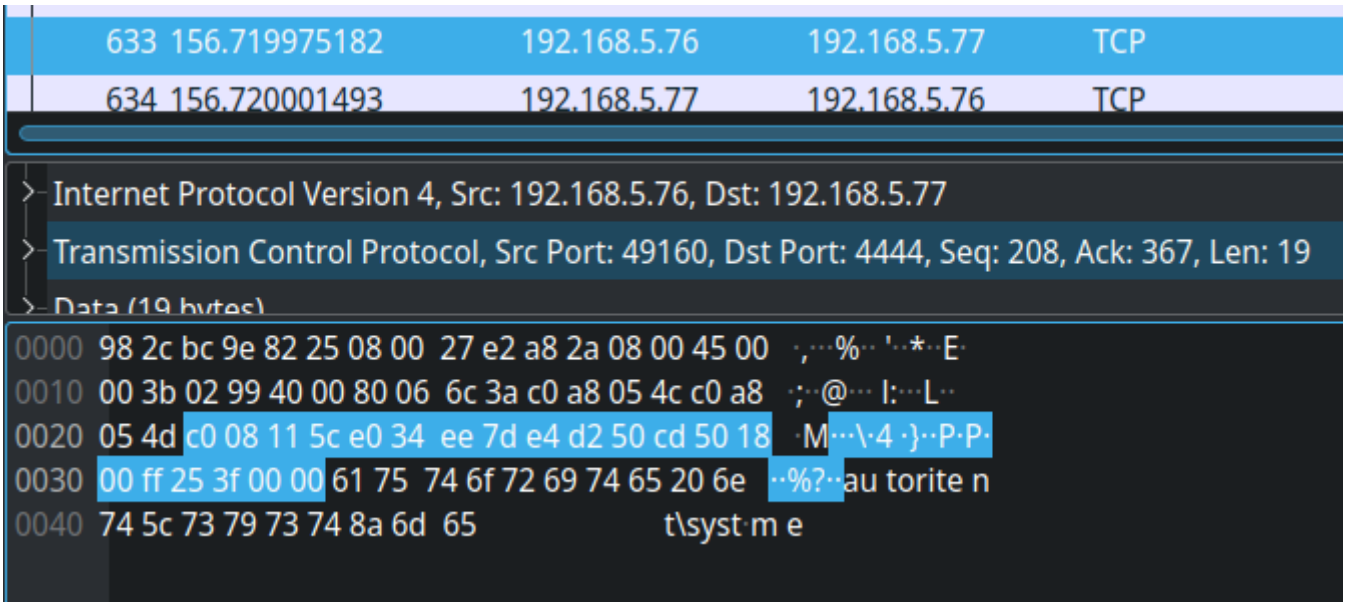
```
Internet Protocol Version 4, Src: 192.168.5.76, Dst: 192.168.5.77
Transmission Control Protocol, Src Port: 49160, Dst Port: 4444, Seq: 106, Ack: 359, Len: 74
Data (74 bytes)
0000  98 2c bc 9e 82 25 08 00  27 e2 a8 2a 08 00 45 00   ·,···%·· '··*··E·
0010  00 72 02 95 40 00 80 06  6c 07 c0 a8 05 4c c0 a8   ·r··@··· l····L··
0020  05 4d c0 08 11 5c e0 34  ee 17 e4 d2 50 c5 50 18   ·M···\·4 ····P·P·
0030  00 ff a6 1b 00 00 43 3a  5c 57 69 6e 64 6f 77 73   ····C: \Windows
0040  5c 73 79 73 74 65 6d 33  32 3e 65 63 68 6f 20 53   \system3 2>echo S
0050  41 57 35 63 39 57 75 4b  63 52 76 0a 53 41 57 35   AW5c9WuK cRv·SAW5
0060  63 39 57 75 4b 63 52 76  0d 0a 0d 0a 43 3a 5c 57   c9WuKcRv ····C:\W
0070  69 6e 64 6f 77 73 5c 73  79 73 74 65 6d 33 32 3e   indows\s ystem32>
```

On confirme un petit plus loin avec l'usage de la commande `whoami`, et la réponse qui suit

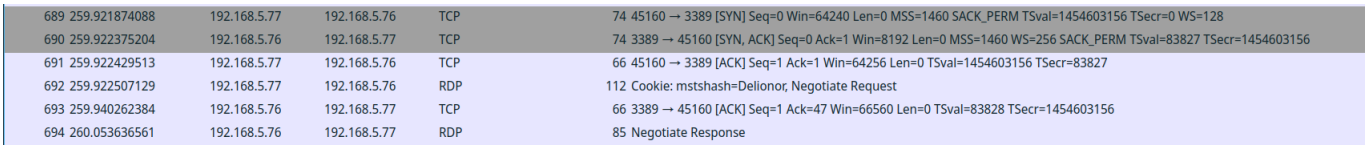


630	156.681595735	192.168.5.77	192.168.5.76	TCP
631	156.682337277	192.168.5.76	192.168.5.77	TCP
632	156.682384670	192.168.5.77	192.168.5.76	TCP

```
Internet Protocol Version 4, Src: 192.168.5.77, Dst: 192.168.5.76
Transmission Control Protocol, Src Port: 4444, Dst Port: 49160, Seq: 360, Ack: 201, Len: 7
Data (7 bytes)
0000  08 00 27 e2 a8 2a 98 2c  bc 9e 82 25 08 00 45 00   ··'··*·, ···%··E·
0010  00 2f 25 41 40 00 40 06  89 9e c0 a8 05 4d c0 a8   ·/%A@·@· ·····M··
0020  05 4c 11 5c c0 08 e4 d2  50 c6 e0 34 ee 76 50 18   ·L·\···· P··4·vP·
0030  01 f6 8c 0b 00 00 77 68  6f 61 6d 69 0a            ····wh oami·
```



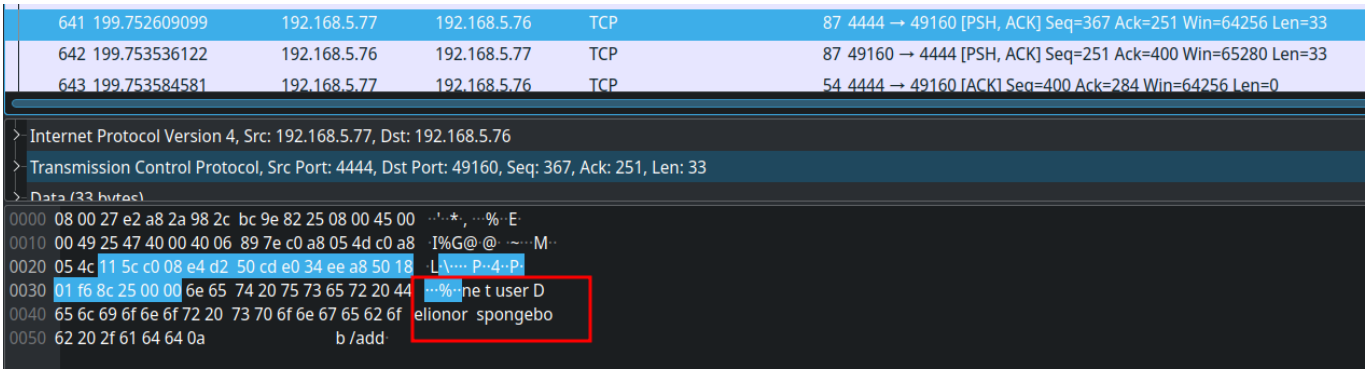
Cet échange continue jusqu'à ce qu'un session RDP soit ouverte entre l'attaquant et le serveur.



3. Flags

On nous demande 2 choses à l'issue de l'analyse de cette capture, retrouver la CVE associée à cette vulnérabilité ainsi que le pseudo de l'attaquant.

Pour le pseudo de l'attauant, on peut remarquer cette commande:



Et lors de la connexion via RDP l'utilisateur utilisé est Delionor. C'est donc le pseudo utilisé par l'attaquant. Pour trouver la CVE associée, avec les informations récoltées on peut faire cette recherche `trans 2 status invalid parameter exploitation smb`. La première page de Google est plutôt bien fournie, le résultat qui revient le plus concerne Eternal Blue, on peut donc pencher vers cette piste et se documenter.

Je recommande vivement de lire ces deux articles: - https://research.checkpoint.com/2017/eternalblue-everything-know/ - https://www.hackers-arise.com/post/2018/11/30/network-forensics-part-2-packet-level-analysis-of-the-eternalblue-exploit Ils expliquent bien mieux que moi la vulnérabilité (bien qu'en anglais). Tant d'un côté réseau que d'un côté système.

On a donc notre pseudo, flag: FMCTF{Delionor}.

Notre vulnérabilité est "EternalBlue", on peut retrouver le numéro de CVE associé avec une recherche Google, à savoir CVE-2017-0144. Flag: FMCTF{CVE-2017-0144}. En vérité il y a plusieurs CVE associées CVE-2017-0143 à CVE-2017-0148 (on peut le voir ici).