

# Investigation numérique

---

## 2. ACQUISITION DES PREUVES

# Sommaire

01

Généralités

02

Méthodologie globale

03

Collecte totale (*full disk*)

04

Collecte de mémoire vive

05

Collecte partielle de données

# 01

## Généralités

# Définition et enjeux

La phase d'acquisition, ou de collecte, est la phase durant laquelle sont récupérées toutes les données nécessaires à l'investigation.

- Activité non bornée dans le temps : elle a lieu au début de l'investigation mais il est tout à fait possible de devoir refaire des collectes tout au long de celle-ci
- La collecte est une étape primordiale pour l'investigation et doit être effectuée avec minutie et rigueur => toute l'analyse découle des données collectées
- Le choix des données à collecter est également très important. Il est possible de passer à côté d'un constat important par oubli de collecte de certaines preuves
- Dans tous les cas, l'analyste est tributaire des données disponibles et collectées
- En fonction du contexte, des prérequis spécifiques peuvent s'appliquer (autorisation d'accéder aux données, huissier, ...)

Les rétentions variables des équipements sont un élément important à prendre en compte : les données disparaissent avec le temps !

# Précautions et bonnes pratiques

- Documenter au maximum toute les étapes de la collecte, mais aussi ce qui n'a pas pu être collecté
- Capturer une image du système / de l'infrastructure la plus proche possible de l'état actuel de celui-ci
- Les moyens et outils de collecte doivent interférer le moins possible avec le système. Utiliser des outils dédiés et maîtrisés. Le principe de Locard s'applique autant aux criminels qu'aux enquêteurs...
- Mettre en place et respecter une chaîne de traçabilité (*chain of custody*)
- Les preuves collectées ne doivent subir aucune altération après leur collecte. Un *hash* doit systématiquement être réalisé après la collecte pour comparaison et une copie doit systématiquement être réalisée sur un disque accessible en lecture seule
- Si un choix se pose entre analyser et collecter, toujours collecter AVANT d'analyser

“

*Nul ne peut agir avec l'intensité que suppose l'action criminelle sans laisser des marques multiples de son passage.*

”

*Edmond Locard*

# 02

## Méthodologie globale

# Méthodologie générale

Globalement, la collecte est menée à bien en réalisant les tâches suivantes :



Dresser la liste des éléments à collecter en fonction  
du besoin et du contexte



Définir une procédure de collecte pour chaque élément



Récupérer les données depuis les équipements



Vérifier l'intégrité des données

Encore une fois, ces tâches peuvent être effectuées à plusieurs reprises au cours de l'investigation.

# Dresser la listes des éléments à collecter

La liste des éléments à récupérer est évidemment très variable en fonction du contexte et du type d'investigation.

Pas de recette miracle ici, il faut essayer de penser à tout ce qui pourrait être utile, quitte à collecter trop de choses et ne pas s'en servir.

Quelques éléments à garder en tête néanmoins :

- Mieux vaut trop que pas assez
- Prendre en compte la rétention des équipements et collecter en priorité ceux avec la plus faible rétention
- Penser aux preuves indirectes si les preuves directes ne sont pas disponibles
- La compréhension du SI est un facteur clé : si vous comprenez comment le SI fonctionne et quels équipements le compose, vous savez quelles données vous pouvez récupérer

Attention, même si trop vaut mieux que pas assez, l'idée n'est pas non plus de récupérer une collecte de tous les serveurs du SI...



# Définir les procédures de collecte

Une fois la liste des éléments à collecter, une procédure de collecte adaptée doit être définie pour chacun de ces éléments définissant au minimum les points suivants :

- Le **type d'équipement** sur lequel la collecte va avoir lieu : serveur physique ou virtuel, poste de travail, routeur, pare-feu, ...
- La **version de l'équipement** en question pour adapter la procédure
- Le **matériel nécessaire** pour accéder à l'équipement et réaliser la collecte : câbles spécifiques, disque dur externe, *write blocker*, outils divers, ...
- Le logiciel adapté à la collecte en question : script maison, opération manuelle, CLI de l'équipement, ...
- Les **besoins en termes d'accès** à l'équipement : conditions d'accès au site physique, compte sur les solutions (Office365, VPN, EDR, ...), compte AD avec des droits suffisant, ...
- Le **type de données attendues** et les **moyens permettant de vérifier l'intégrité et la complétude** des données : somme de contrôle, présence de tous les fichiers attendus, ...

Dans la majorité des cas, les mêmes procédures sont appliquées d'une investigation à une autre et cette étape devient quasiment automatique. Attention toutefois à garder ces idées en tête pour ne pas se faire surprendre par un cas particulier.

# Mener la collecte

Les procédures établies, vient le moment de les mettre en application.

De manière générale, on pourra classer les collectes en quatre types :

- Les copies bit-à-bit, ou *full disk*
- Les collectes partielles de fichiers du système (à froid ou à chaud)
- Les collectes de mémoire vive et de données volatiles (collectes à chaud)
- Les autres collectes (équipement spécifique, *cloud*, capture réseau, ...)

Les trois premiers seront détaillés dans les prochaines parties.

Le dernier est une catégorie fourre-tout trop vaste pour être détaillée. Certains exemples seront néanmoins abordés dans la suite du cours et pendant les TPs.



A froid : système éteint, données figées.

A chaud : le système est en cours d'exécution pendant la collecte

# Vérifier l'intégrité des données

Dans l'idéal, on aimerait pouvoir s'assurer que les données collectées sont bien, à l'identique, les données présentes sur le système. On effectue donc un contrôle d'intégrité.

Plusieurs problématiques se posent alors pour de nombreux éléments :

- Que signifie « à l'identique » pour la RAM ? Elle va forcément évoluer pendant la collecte. Pire, la collecte elle-même va modifier la RAM. Quel sens peut donc avoir un contrôle d'intégrité dans ce cadre là ?
- De même pour les fichiers qui sont écrits très régulièrement (base de registre ou \$MFT par exemple). Si je fais un *hash* de ce fichier à un instant T et que je réalise sa collecte à l'instant T+1, il est très probable que je n'ai pas le même *hash* à la fin de la collecte.
- Dans le cas des fichiers collectés par le client, comment vérifier leur intégrité ?

La réponse à toutes ces questions est : **on s'adapte et on fait au mieux.**

L'intégrité des données récupérées ne pourra être vraiment vérifiée que dans les cas suivants :

- La donnée est figée et n'évolue pas entre le premier calcul d'intégrité et la collecte (collecte à froid : *full disk* notamment. Les utilitaires de copie de disque calculent systématiquement un *hash* avant et après la collecte)
- La collecte est réalisée sous contrôle d'un huissier de justice qui atteste qu'elle provient bien du système et n'a pas été altérée lorsque le *hash* après collecte est calculé.

# Vérifier l'intégrité des données

Dans les autres cas, il s'agira donc plus d'un contrôle de complétude que d'un contrôle d'intégrité :

- Pour la RAM, on vérifie qu'on a un fichier qui correspond à la taille attendue
- Pour les collectes à chaud de données volatiles ou de fichiers systèmes, on vérifie que tous les fichiers sont bien récupérés et qu'ils ne sont pas corrompus.

Dans tous les cas, on calcule toujours un *hash* après collecte, mais plus pour maintenir la chaîne de preuve que pour attester de l'intégrité lors de la collecte.

# 03

Collecte totale (*full disk*)

# Collecte *full disk*

Pas toujours à privilégier en raison de la taille du fichier résultant et du temps de collecte, la collecte bit-à-bit est parfois indispensable pour mener à bien une investigation.

La procédure est globalement la même quelque soit l'équipement :

1. Démontage de l'appareil si le disque n'est pas immédiatement accessible
2. Noter le numéro de série du disque et le consigner
3. Brancher le disque sur la station de copie en utilisant un *write blocker* type Tableau (vérifier qu'il est bien configuré en mode lecture seule !)
4. Réaliser la copie avec des outils appropriés comme **dc3dd**, **FTK Imager** ou **EnCase**
5. Vérifier l'intégrité du disque une fois la copie terminée (les outils dédiés le font automatiquement)
6. Consigner le disque original dans un coffre ET NE PLUS Y TOUCHER
7. Placer la copie du disque sur un serveur accessible en lecture seule uniquement et ne travailler que sur cette copie (certains préconisent même de faire une deuxième copie comme copie de travail).



Et pour les disques virtuels ? Copier / coller / vérifier l'intégrité. Facile.

Collecte *full disk*

Démo

# Collecte *full disk*

Exemple avec dc3dd :

```
forensics@forensics:~$ sudo dc3dd if=/dev/sdb hof=USB.raw hash=sha1 hlog=USB_copy.log bufsz=$((512*256))
```

```
dc3dd 7.2.646 started at 2024-10-22 21:09:17 +0200
```

```
compiled options:
```

```
command line dc3dd if=/dev/sdb hof=USB.raw hash=sha1 hlog=USB_copy.log bufsz=131072
```

```
device size: 15630336 sectors (probed), 8,002,732,032 bytes
```

```
sector size: 512 bytes (probed)
```

```
8002732032 bytes ( 7.5 G ) copied ( 100% ), 1041 s, 7.3 M/s
```

```
8002732032 bytes ( 7.5 G ) hashed ( 100% ), 60 s, 128 M/s
```

```
input results for device `/dev/sdb':
```

```
15630336 sectors in
```

```
0 bad sectors replaced by zeros
```


```
9effabfc333033698b32bdca1ef42818472f8786 (sha1)
```

```
output results for file `USB.raw':
```

```
15630336 sectors out
```

```
[ok] 9effabfc333033698b32bdca1ef42818472f8786 (sha1)
```

```
dc3dd completed at 2024-10-22 21:26:38 +0200
```



Pensez à utiliser l'option `hof` pour que dc3dd calcule le *hash* avant et après la copie automatiquement.



# Collecte *full disk*

## Exemple avec ewfacquire :

```
forensics@forensics:~$ sudo ewfacquire -t USB.E01 /dev/sdb
[sudo] password for forensics:
ewfacquire 20140814
```

```
Device information:
Bus type:          USB
Vendor:           SanDisk
Model:            Cruzor Blade
Serial:           4C531001340523107085
```

```
Storage media information:
Type:             Device
Media type:       Removable
Media size:       8.0 GB (8002732032 bytes)
Bytes per sector: 512
```

```
Acquiry parameters required, please provide the necessary input
Case number: 1
Description: Simple copy of USB
Evidence number: 1
```

```
Written: 7.4 GiB (8002732220 bytes) in 12 minute(s) and 31 second(s)
MD5 hash calculated over data:      1652c9864c6d2925dc8f3404b905
ewfacquire: SUCCESS
```



```
forensics@forensics:~$ ls USB*
USB.E01  USB.E02  USB.E03  USB.E04  USB.E05  USB.E06
```

```
forensics@forensics:~$ ewfinfo USB.E01
ewfinfo 20140814
```

```
Acquiry information
Case number:          1
Description:          Simple copy of USB
Examiner name:       Roman
Evidence number:      1
Acquisition date:    Tue Oct 22 21:47:23 2024
System date:         Tue Oct 22 21:47:23 2024
Operating system used: Linux
Software version used: 20140814
Password:            N/A
Model:               Cruzor Blade
Serial number:       4C531001340523107085
```

```
EWf information
File format:          EnCase 6
Sectors per chunk:    128
Error granularity:    128
Compression method:   deflate
Compression level:    no compression
```

```
Media information
Media type:           removable disk
Is physical:         yes
Bytes per sector:     512
Number of sectors:    15630336
Media size:           7.4 GiB (8002732032 bytes)
```

```
Digest hash information
MD5:                  1652c9864c6d2925dc8f3404b9051f70
```

L'intérêt de ewfacquire est d'obtenir un fichier au format EnCase, permettant d'utiliser tous les outils associés (ewfmount, ewfinfo, ewfverify, ...)

# Collecte *full disk*

Information for C:\Users\Romian\Desktop\USB\_Copy:

Physical Evidentiary Item (Source) Information:

[Device Info]

Source Type: Physical

[Drive Geometry]

Cylinders: 972

Tracks per Cylinder: 255

Sectors per Track: 63

Bytes per Sector: 512

Sector Count: 15 630 336

[Physical Drive Information]

Drive Model: SanDisk Cruzer Blade USB Device

Drive Serial Number: 4C531001340523107085

Drive Interface Type: USB

Removable drive: True

Source data size: 7632 MB

Sector count: 15630336

[Computed Hashes]

MD5 checksum: 22e632233c7d25d6fa0d3c71536e79bd

SHA1 checksum: 847f0a9420299c529f163a66170f2e81ba719a16

Image Information:

Acquisition started: Wed Sep 13 18:01:27 2023

Acquisition finished: Wed Sep 13 18:05:05 2023

Segment list:

C:\Users\Romian\Desktop\USB\_Copy.E01

Image Verification Results:

Verification started: Wed Sep 13 18:05:05 2023

Verification finished: Wed Sep 13 18:05:24 2023

MD5 checksum: 22e632233c7d25d6fa0d3c71536e79bd : verified



# 04

Collecte de mémoire vive

# Collecte de mémoire vive

Contrairement à ce qu'on pourrait penser, la collecte de la mémoire vive n'est pas forcément systématique et ce pour plusieurs raisons :

- La quantité de données à collecter peut être importante (plusieurs dizaines de Go par machine)
- Processus parfois laborieux
- Les données contenues dans la mémoire peuvent souvent être obtenues par d'autres moyens (récupération des fichiers sur le disque, commandes système, ...)
- L'analyse peut être très fastidieuse et coûteuse en temps

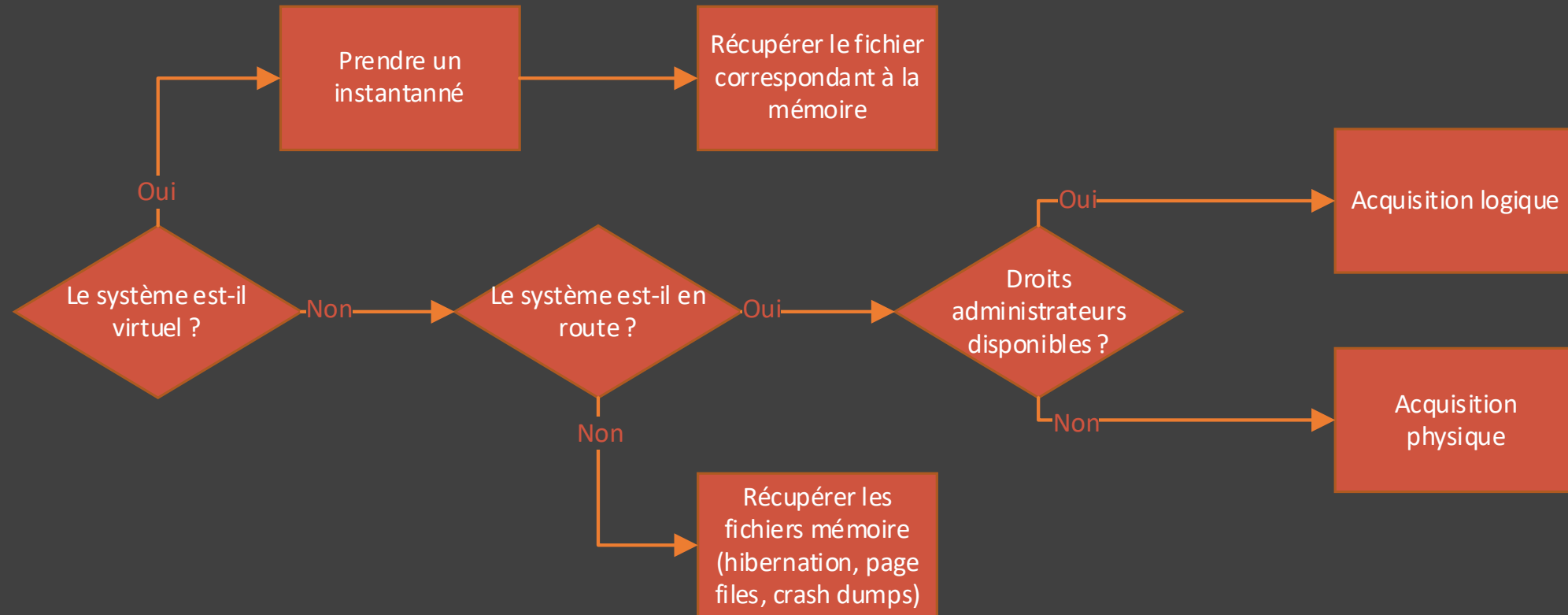
La collecte de la mémoire se justifie donc pleinement dans des cas très spécifiques :

- Suspicion d'un programme masquant sa présence au système
- Récupération d'un malware *fileless*
- Nécessité d'accéder à des données qu'on sait n'être contenues qu'en mémoire (contenu des fenêtres, données manipulées par tous les programmes, ...)
- Récupération de la version dépackée d'un malware
- ...

Encore une fois tout est donc question de contexte. Dans le cas d'une compromission étatique, on prendra bien évidemment la peine de tout regarder, ce qui se justifiera nettement moins pour une intervention chez un client par exemple.

# Collecte de mémoire vive

Que collecter exactement, et quand ?



# Collecte de mémoire vive

## Machines virtuelles :

- La RAM est stockée dans un fichier dont le format dépend de l'hyperviseur
- Le nom du fichier et la conversion de ce dernier en format exploitable dépend du fabricant
- Les plus classiques :
  - .vmss, .vmsn, .vmem pour VMWare, à convertir avec vmss2core
  - .bin et .sav pour Hyper-V, utiliser LiveKd

## Fichiers mémoire :

- *hiberfil.sys* : enregistre l'état du système pour les opérations d'hibernation. Doit être converti en un format exploitable (Volatility sait le faire)
- *pagefile.sys* et les autres fichiers de pagination : emplacements mémoire stockés sur le disque pour étendre la taille de la RAM
- *Crash dumps* : enregistrés lors d'un BSOD, par défaut sous %SystemRoot%\MEMORY.DMP
- *Fichiers WER (Windows Error Reporting)*: enregistrés lors du crash des programmes.
- *Volume Shadow Copy* : contient également des données volatiles qui peuvent être utiles.

## Collecte logique :

- Exécution d'un logiciel sur le poste pour collecter la RAM. Nécessite les droits administrateur
- WinPMEM ou FTKImager pour Windows, LiME ou fmem pour Linux, entre autres.

## Collecte physique :

- Possible dans certains cas mais souvent limités (port FireWire, limité à 4Go, ...)
- A réserver au cas extrêmes.

Démo

# Collecte de mémoire vive

Exemple avec WinPmem :

```
C:\Users\Romian\Desktop>winpmem_mini_x64_rc2.exe ram.dmp
WinPmem64
Extracting driver to C:\Users\Romian\AppData\Local\Temp\pme153B.tmp
Driver Unloaded.
Loaded Driver C:\Users\Romian\AppData\Local\Temp\pme153B.tmp.
Deleting C:\Users\Romian\AppData\Local\Temp\pme153B.tmp
The system time is: 08:25:34
Will generate a RAW image
- buffer_size_: 0x1000
CR3: 0x00001AE000
4 memory ranges:
Start 0x00001000 - Length 0x0009E000
Start 0x00100000 - Length 0x697DB000
Start 0x6FBFF000 - Length 0x00001000
Start 0x100000000 - Length 0x37F800000
max_physical_memory_ 0x47f800000
Acquisition mode PTE Remapping
Padding from 0x00000000 to 0x00001000
pad
- length: 0x1000

...

78% 0x38A000000 .....
83% 0x3BC000000 .....
87% 0x3EE000000 .....
91% 0x420000000 .....X.....X.....
96% 0x452000000 .....XX.X.....
The system time is: 08:27:32
Driver Unloaded.
```

ram.dmp	09/10/2023 10:27	Fichier DMP	18 866 176 ...
winpmem_mini_x64_rc2.exe	09/10/2023 09:14	Application	516 Ko



# Collecte de mémoire vive

Exemple avec fmem sous Linux :

```
forensics@forensics:~/Tools/fmem$ sudo ./run.sh
Module: insmod fmem.ko a1=0xfffffffffa7d0d250 : OK
Device: /dev/fmem
----Memory areas: ----
cat: /proc/mtrr: Input/output error
-----
!!! Don't forget add "count=" to dd !!!
forensics@forensics:~/Tools/fmem$ sudo dd if=/dev/fmem of=/home/forensics/mem.raw bs=1M count=8192
8192+0 records in
8192+0 records out
8589934592 bytes (8.6 GB, 8.0 GiB) copied, 79.2063 s, 108 MB/s
```

# 05

Collecte partielle de données

# Collecte partielle de données

## Avantages et inconvénients de la méthode

- **Collecte rapide** (15 – 20 min) et **quantité de données obtenue faible** (100 Mo / 1 Go) => il est possible de collecter et de transférer un grand nombre de preuves depuis de nombreuses machines en peu de temps
  - On récupère directement les **artefacts dont on a besoin** et qu'il aurait de toute façon fallu récupérer dans un deuxième temps avec une copie complète
  - Possibilité **d'automatiser** la collecte des données avec des outils dédiés
  - **Pas besoin d'un expert** pour réaliser la collecte, la procédure peut être confiée à un administrateur ou un technicien chez le client => pas de déplacement sur place
- On ne récupère pas tout, et il est donc possible d'avoir besoin de faire une **deuxième collecte** à un moment donné (pour récupérer un binaire pour analyse par exemple). Il est possible que **l'état du système ait changé** entre temps : serveur réinstallé / chiffré, fichiers supprimés, ...
  - Il est donc important **d'anticiper au maximum**, mais on ne peut pas pour autant déterminer par avance tout ce dont on pourrait avoir besoin

En fonction du contexte, cette méthode pourrait donc ne pas être indiquée (police, suspicion de destruction de données au démarrage du poste, ...). Cependant, ses nombreux avantages en font la méthode par défaut pour la majorité des réponses à incident en entreprise (en tout cas la mienne...)

# Collecte partielle de données

## A froid ou à chaud ?

### Avantages de la collecte à chaud :

- Ne force pas l'arrêt d'un système critique pour le client (trop d'impact métier, serveurs d'hôpitaux, ...)
- Permet d'être discret et de maintenir tous les serveurs opérationnels pour observer les actions de l'attaquant. Ceci peut être particulièrement important dans certains contextes (espionnage, attaque étatique, ...)
- Autorise la récupération des données volatiles (en respectant l'ordre de volatilité)

En revanche, la collecte à chaud expose au risque d'altération ou de destruction des données par l'attaquant (chiffrement du SI, détection de la collecte) ainsi qu'à une réaction de sa part s'il surveille l'activité sur les serveurs (déclenchement de la phase suivante de l'attaque, suppression de ses traces, ...).

En pratique, ce risque est assez faible pour les attaques classiques, mais dans un cadre plus sensible il faut le garder en tête.

### Recommandation à garder en tête lors d'une collecte à chaud :

- Sauf cas de force majeure, ne pas éteindre le système avant la fin de la collecte. De nombreux éléments pourraient disparaître et la routine d'extinction a pu être modifiée par l'attaquant
- Ne pas utiliser les outils présents sur le système (hors commandes natives) mais des outils tiers approuvés et maîtrisés
- Ne pas utiliser d'outils qui modifient la date de dernier accès au fichier comme `tar` ou `xcopy`



### Ordre de volatilité :

1. Date du système
2. Registres CPU et caches (bon courage pour les récupérer...)
3. Table de routage, connexions actives, cache ARP, liste des processus et des services, mémoire vive
4. Systèmes de fichiers temporaires
5. Fichiers et disques

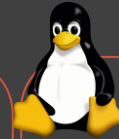
# Collectes partielle de données

Liste non exhaustive de données collectées :



- Informations sur le systèmes
- Données volatiles
- Bases de registre
- Journaux d'événements
- \$MFT des disques
- \$UsnJrnl des disques
- Tâches planifiées
- Historiques des navigateurs et WebCache
- Fichiers de Prefetch
- Historique des commandes PowerShell
- Base SRUM
- Données persistantes de WMI
- Répertoires Recents et Startup
- Fichiers de l'AmCache
- ...

Des outils comme **DFIR-ORC** (ANSSI) ou **Kape** sont fréquemment utilisés.



- Liste des fichiers avec timestamps MACB
- Informations sur le système
- Informations réseaux
- Processus en cours d'exécution
- Journaux dans /var/log
- Configuration des services
- Paquets installés
- Répertoire /tmp
- Crontab
- Liste des sudoers
- Historiques de commandes
- Clefs SSH et known hosts
- ...

**Problématique principale : Linux n'est pas uniformisé, et il est donc difficile d'avoir une méthodologie générique.**

Pas vraiment d'outils reconnus.

Démo

# Collectes partielle de données

## Exemple avec Kape:

Total execution time: 93,9334 seconds

KAPE version 1.3.0.2, Author: Eric Zimmerman, Contact: <https://www.kroll.com/kape> (kape@kroll.com)

KAPE directory: C:\Users\Romian\SynologyDrive\Formation\Forensic\Outils\kape

Command line: --tsource C: --tdest C:\Users\Romian\Desktop\Collect --target !BasicCollection --gui

System info: Machine name: DESKTOP-NE9TOUQ, 64-bit: True, User: Romian OS: Windows10 (10.0.19045)

Using Target operations

Found 19 targets. Expanding targets to file list...

Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!

Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!

Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!

Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!

Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!

Target WindowsIndexSearch with Id 9828b927-f955-464a-80fb-a48ce0101236 already processed. Skipping!

Found 1,374 files in 8.873 seconds. Beginning copy...

Deferring C:\Windows\System32\winevt\logs\Application.evtx due to IOException...

Deferring C:\Windows\System32\winevt\logs\Microsoft-Windows-Windows Defender\Support\MPDeviceCont...

Deferring C:\Windows\System32\winevt\logs\Microsoft-Windows-Windows Defender\Support\MPLog-202409...

Deferring C:\Windows\System32\winevt\logs\Microsoft-Windows-Windows Defender\Support\MPScanSkip-2...

Deferring C:\Windows\System32\winevt\logs\Microsoft-Windows-Windows Defender\Support\MpWppCoreTra...

Deferring C:\Windows\System32\winevt\logs\Microsoft-Windows-Windows Defender\Support\MpWppTracing...

Deferring C:\Windows\System32\winevt\logs\HardwareEvents.evtx due to IOException...

Deferring C:\Windows\System32\winevt\logs\Internet Explorer.evtx due to IOException...

Deferring C:\Windows\System32\winevt\logs\Key Management Service.evtx due to IOException...

Collect > C >

Nom	Modifié le	Type	Taille
\$Extend	22/10/2024 22:35	Dossier de fichiers	
\$Recycle.Bin	22/10/2024 22:34	Dossier de fichiers	
Program Files (x86)	22/10/2024 22:34	Dossier de fichiers	
ProgramData	22/10/2024 22:34	Dossier de fichiers	
Users	22/10/2024 22:34	Dossier de fichiers	
Windows	22/10/2024 22:35	Dossier de fichiers	
\$Boot	22/10/2024 22:35	Fichier	8 Ko
\$LogFile	22/10/2024 22:35	Fichier	65 536 Ko
\$MFT	01/01/2020 00:49	Fichier	1 479 680 Ko
\$Secure_\$SDS	01/01/2020 00:49	Fichier	4 019 Ko

# Des questions ?

## Quelques sources :

- *File System Forensic Analysis* de Brian Carrier
- *The Art of Memory Forensics* de Michael Hale Ligh, Andrew Case, Jamie Levy, Aaron Walters
- [RFC 3227 : Guidelines for Evidence Collection and Archiving](#)
- [Exemple de méthodologie d'analyse forensique](#)

## Liens vers les outils :

- [KAPE](#)
- [WinPmem](#)
- [fmem](#)
- [FTK-Imager](#)