Suppressed files
-
Rapport

BELLET Inès
PHOK Amélie

21 mai 2024

Project 2
SMINFO2D

Table des matières

1 Introduction

Ce rapport présente notre projet *Suppressed Files* qui concerne la récupération de données supprimées dans le cadre d'une analyse forensique numérique. Au premier semestre, nous avons fait un état de l'art[1] pour synthétiser et comparer les outils existants utilisés pour récupérer des fichiers effacés. Nous avons ainsi choisi quatre logiciels afin de pouvoir les tester dans la partie analyse pratique de ce projet.
Suivant un protocole scientifique détaillé en 3, nous avons testé les logiciels suivants :

— Autopsy et Encase Forensics pour Amélie
— OSForensics et Recuva pour Inès

2 Objectifs

L'objectif de cette partie du projet est de faire une analyse pratique des quatre logiciels afin de continuer la comparaison commencée dans l'analyse théorique. Cette comparaison permettra, dans l'idéal, d'identifier le plus performant afin de pouvoir l'intégrer au projet G'DIP du GREYC.

3 Protocole expérimental

Nous avons choisi de tester les différents logiciels sur différents types de fichiers et différents scénarios. Nous avons ainsi utilisé trois types de fichiers :

1. fichiers textes (.txt)
2. fichiers images (.png)
3. fichiers cryptés (.pcv)

Avant même de pouvoir commencer à tester, nous avons dû trouver des corpus de données. Pour les fichiers textes, nous avons utilisé des textes issus de la librairie Gutenberg[2]. Pour les images, nous avons trouvé des images exemples issus d'un site gratuit[3]. Pour les fichiers cryptés, nous avons simplement chiffrés les images avec le logiciel open-source Picocrypt.

Le protocole expérimental peut être schématisé de la façon suivante :

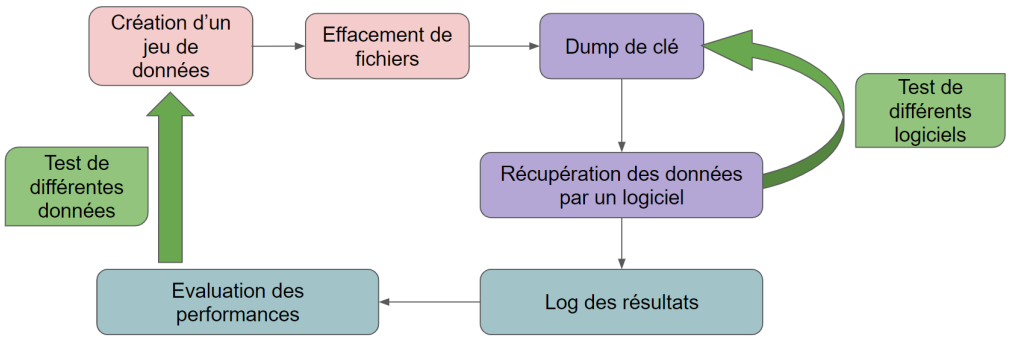

FIGURE 1 – Protocole expérimental

Afin de tester les différents logiciels, nous avons décidé d'appliquer différents scénarios d'effacement sur les clés USB.

1. copier 1.5GB des fichiers sources vers la clé et supprimer 50%
2. copier 1.5GB des fichiers sources vers la clé et supprimer 90%
3. copier 2.5GB des fichiers sources, supprimer 10%, puis recopie 2GB sur la clé
4. copier 2.5GB des fichiers sources, supprimer 70%, puis recopie 2GB sur la clé
5. copier 1.5GB des fichiers sources, supprimer 10%, puis recopie 2GB sur la clé
6. copier 1.5GB des fichiers sources, supprimer 70%, puis recopie 2GB sur la clé

Nous avons utilisé des clés de 3Go pour effectuer nos tests.

Nous avons utilisé l'utilitaire dd pour Windows pour dumper les clés afin de ne pas compromettre les clés lors des manipulations.

Enfin, le langage Python a été utilisé pour la majorité des fonctions telles que les fonctions de copie dans la clé, d'effacement de fichiers et comparaison. Pour automatiser certains processus, Windows Batch a également été utilisé.

4 Éléments Techniques

4.1 Python uniquement

Inès a utilisé seulement des scripts python. Dans un premier temps, la une fonction `copy_files()` en Python a été créer ce qui permet de copier des fichiers tous en respectant la capacité qu'on veut. Pour cette fonction, il faut avoir au minimum une base de donnée égale à la clé usb utilisé.

Require: liste fichier_a_copier, nom clé usb, pourcentage à copie
  while la taille de clé < pourcentage_a_copie*taille_clé_usb **do**
    tirer un fichier de fichier_à_copier au hasard
    if la capacité du fichier + capacité de la clé actuelle < capactié voulue **then**
      copier le fichier sur la clé
      ajout de la capacité du fichier à capcité de la clé actuelle
    end if
    Supprimer le fichier de la liste fichier_à_copier
  end while

Algorithm 1: Fonction de copie

Dans cette algorithme, on copie une seule fois un fichier sur la clé usb et on repéte l'opération jusqu'à atteindre la taille voulue

Require: nom clé usb, pourcentage de supression,fichier log, numéro de test

  Calculer le nombre de fichiers à supprimer dans la clé pour atteindre le pourecentage de supression voulue
  while le nombre de fichier déjà supprimer < nombre fichier à supprimer **do**
    tirer au hasard un fichier dans la clé
    supprimer le fichier dans la clé
    ajouter le fichier dans le log
    incrémenté le nombre de fichier déjà supprimer
  end while

Algorithm 2: Fonction de supression

Cette algorithme permet de supprimer les fichiers dans la clé usb au

hasard jusqu'à la capacité voulue. On garde les noms des fichiers supprimer dans un log.

Require: chemin de fichiers_a_copier ,chemin de fichiers_recuperer, numero test, logiciel, type de données
Creer deux listes contenant les fichiers de fichiers_a_copier puis de fichiers_recupere
for chaque fichier f de fichiers_recupere **do**
comapre f à ceului dans fichiers_a_copier octet à octet
ecrit dans le log si ils ont identiques ou non
end for

Algorithm 3: Fonction de comparaison

Ce troisiéme algorithme permet de comparer les fichiers récuperer aux fichiers originaux. On ecrit également dans un log si les fichiers récuperer sont identiques ou non.

Inès n'a pas utilisé de script bash. Les differentes étapes qu'elle a réalisé pour effectuer un dump d'une clé sont :

- Formatage de la clé USB par la commande format d : /V :Cle realiser deux fois
- Copie des fichiers avec la fonction copy_files()
- Supression des fichiers avec la fonction del_files()
- Recopie des fichiers avec la fonction copy_files() si besoin
- Dump de la clé avec la commande ./dd

Les étapes copie,suppression,recopie a pris chaqu'une à peu près 1 heure.

4.2 Python + Batch Script

Amélie a utilisé un mélange de Python et de Batch Script. Pour copier les fichiers, une fonction `copy_files_to_usb()` en Python a été crée pour copier les fichiers jusqu'à ce qu'une certaine taille a été atteinte.

```
Require: dossier source, nom de la clé, taille en gb
 1: if la clé est de la taille voulue then
 2:     ne rien faire
 3: else
 4:     while la taille voulue n'est pas atteinte do
 5:         for chaque fichier du dossier source do
 6:             copier le fichier dans la clé USB
 7:             if le fichier a déja été copié then
 8:                 modifier le nom pour indiquer que c'est une copie
 9:             end if
10:         end for
11:     end while
12: end if
```

Algorithm 4: Fonction de copie

Ce premier algorithme 4 permet de copier les fichiers d'un dossier source, potentiellement plusieurs fois, jusqu'à atteindre la taille voulue dans le dossier destination, ici la clé USB.

```
Require: dossier source, pourcentage à supprimer, fichier log, numéro de
    test
 1: on prend une liste des fichiers disponibles
 2: on choisit dans la liste des fichiers au hasard jusqu'à un certain pourcen-
    tage
 3: for chaque fichier choisi à supprimer do
 4:     ajouter le nom du fichier dans un log
 5:     supprimer le fichier de la clé
 6: end for
```

Algorithm 5: Fonction d'effacement

Ce second algorithme 5 nous permet de supprimer, au hasard, les fichiers dans un dossier source, ici, la clé USB. Il nous permet également de garder une trace des fichiers supprimés dans un log pour pouvoir comparer les résultats des logiciels.

Require: dossier 1, dossier 2, numéro de test
1: on crée deux listes pour chaque dossier
2: **for** chaque fichier de dossier 1 **do**
3: on crée un pattern avec le nom du fichier 1
4: **for** chaque fichier de dossier 2 **do**
5: **if** le pattern correspond **then**
6: on compare bit à bit si les fichiers sont identiques
7: on ecrit dans un fichier log s'ils le sont.
8: **end if**
9: **end for**
10: **end for**

Algorithm 6: Fonction de comparaison

Ce dernier algorithme 6 nous permet de comparer bit à bit les fichiers entre deux dossiers. Cela nous permet d'évaluer la récupération de fichiers.

Listing 1 – Creation des images pour tests sur png

```
@echo off
setlocal enabledelayedexpansion

@REM Set source folder for copying
set SOURCE_FOLDER=.\sample
@REM Set destination folder (USB drive)
set DESTINATION_FOLDER=E:
@REM Set maximum size to copy (1.5GB)
set MAX_SIZE_GB=1.5
@REM Initial value for DELETE_PERCENTAGE
set DELETE_PERCENTAGE=50

echo PART 1 of 3

@REM Loop through each test
for /l %%i in (20,1,21) do (
    echo Running test number %%i...

    echo Formatting drive ...
    format /Y /P:2 e:

    echo Copying files ...
    "D:/Program Files/Python/python.exe" copy_until_size.py %
        SOURCE_FOLDER% %DESTINATION_FOLDER% !MAX_SIZE_GB!

```

```
    @REM Delete 50% of the files from the USB drive
    echo Deleting files ...
    "D:/Program Files/Python/python.exe" random_del.py %DESTINATION_FOLDER% !DELETE_PERCENTAGE! .\logs\log_%%i.log %%i
    echo Dumping drive to img ...
    .\dd if=\\.\e: of=.\disks\usb_%%i.img --progress
    @REM Increase DELETE_PERCENTAGE by 40 for the next iteration
    set /a DELETE_PERCENTAGE+=40
)

echo PART 2 of 3
set DELETE_PERCENTAGE=10

for /l %%i in (22,1,23) do (
    echo Running test number %%i...
    set MAX_SIZE_GB=2.5

    echo Formatting drive
    format /Y /P:2 e:

    echo Copying files
    "D:/Program Files/Python/python.exe" copy_until_size.py %SOURCE_FOLDER% %DESTINATION_FOLDER% !MAX_SIZE_GB!

    @REM Delete 50% of the files from the USB drive

    echo Deleting files
    "D:/Program Files/Python/python.exe" random_del.py %DESTINATION_FOLDER% !DELETE_PERCENTAGE! .\logs\log_%%i.log %%i

    set MAX_SIZE_GB=2
    echo Copying back files
    "D:/Program Files/Python/python.exe" copy_until_size.py %SOURCE_FOLDER% %DESTINATION_FOLDER% !MAX_SIZE_GB!

    echo Dumping drive to img
    .\dd if=\\.\e: of=.\disks\usb_%%i.img --progress

    @REM Increase DELETE_PERCENTAGE by 60 for the next iteration
    set /a DELETE_PERCENTAGE+=60
```

```
)

echo PART 3 of 3
set DELETE_PERCENTAGE=10


for /l %%i in (24,1,25) do (
    echo Running test number %%i...
    set MAX_SIZE_GB=1.5

    echo Formatting drive
    format /Y /P:2 e:

    echo Copying files
    "D:/Program Files/Python/python.exe" copy_until_size.py %
        SOURCE_FOLDER% %DESTINATION_FOLDER% !MAX_SIZE_GB!

    @REM Delete 50% of the files from the USB drive

    echo Deleting files
    "D:/Program Files/Python/python.exe" random_del.py %
        DESTINATION_FOLDER% !DELETE_PERCENTAGE! .\logs\log_%%i.
        log %%i

    set MAX_SIZE_GB=2
    echo Copying back files
    "D:/Program Files/Python/python.exe" copy_until_size.py %
        SOURCE_FOLDER% %DESTINATION_FOLDER% !MAX_SIZE_GB!

    echo Dumping drive to img
    .\dd if=\\.\e: of=.\disks\usb_%%i.img --progress

    @REM Increase DELETE_PERCENTAGE by 60 for the next iteration
    set /a DELETE_PERCENTAGE+=60
)
```

Enfin, ce dernier script est un exemple de fichier batch qui a permis à Amélie d'automatiser le processus de création des dumps de clé. En effet, ce processus a pris un certain temps (~8h pour tous les scnéarios pour un type de fichier) et il a été judicieux de l'automatiser au maximum. Le script est constitué de trois boucles for qui permettent de créer les six scénarios choisis.

Cependant, les trois boucles sont construites sur le même format :

— formatage de la clé USB

— copie des fichiers sources avec la fonction de copie Python

— suppression des fichiers avec la fonction d'effacement Python
— recopie possible selon le scénario
— dump de la clé avec l'utilitaire dd for Windows

5 Expérimentations

5.1 OSForensics

Ce logiciel a une version gratuit mais nous avons comme même demander une licence temporaire. Il n'y a pas de ligne de commande donc toute les manipulations sont faite manuellement. C'est un logiciel intuitif avec des icônes et des noms clair. De plus, on a une documentation ouvrable sur la page de l'outil correspondant.

Pour chaque scénarios, Inès a crée une nouvelle affaire dans le logiciel et ainsi monté un nouveau disque de l'image du scénario correspondante en lecture seulement. Cette opération est proposé par le logiciel. Puis elle a lancé la recherche de fichiers supprimer avec les parametres scan files index records et mis la minimum quality à All files et l'option enable carving. Ce logiciel ne donne pas le temp que la recherche a durée mais il est très rapide généralement entre 4-6 secondes. Avec cette configuration, des entrèes NTFS $I30 (attribué à chaque répertoire) apparait dans la récupération ce qui permet de prouver l'existant d'un fichier mais il n'est pas exploitable, la qualité est de 0. Ces entrées n'ont pas été prise en compte dans les résultats. Pour les png, l'option enable carving peux être utiliser et donne des fragments et la localisation en hex sur la clé. Dans la recuperation, Inès a mis cette option mais pas utiliser ces fichiers dans le test de comparaison.

En moyenne, OSForensisc atteint un taux de 42.02% de fichiers intacts lors de la récupération.

5.2 Recuva

Recuva est un logiciel avec une version gratuit que nous avons utilisé mais il n'est pas considéré comme un outil de criminalistique numérique. Néamoins dans les tests effectués, il a été rapide et plutôt efficace. Les deux points négatifs de se logiciel sont son interface et un manque d'outil. Son interface se présente comme des boîtes de dialogue et on peux pas enregistrer des "affaires" comme avec osforensics donc il faut configurer pour chaque

essais. De plus, recuva ne propose pas de monter une image de disque.

Les paramétres utilisaient pour la récupération de données sont l'option enable deep scan, la précision de la location de la recherche (dans la clé usb) et la sélection de tous les types de fichiers. Enable deep scan permet de faire une recherche dans les clusters non alloués par leurs en-têtes et pieds de page. Avec ce paramétre coché pour les test des fichiers png, on trouve des captchas sur la clé, elle a été mal formater. Le logiciel nous dit aussi par quelles fichiers le captcha a été écrasé. Donc pour les tests de png, le temps indiqué est pour toute l'analyse de la clé avec les captachas. Mais seulement les fichiers samples sont utilisé pour la comparaison et les résultats pour le nombre de fichiers suprimmer récupérer.

En moyenne, Recuva donne un taux de 43.54% de fichiers intacts lors de la récupération.

5.3 Autopsy

D'après notre état de l'art du premier semestre, Autopsy est un logiciel open source, il a été donc très facile de l'installer et commencer à l'utiliser. Nous avons également vu qu'il était possible d'utiliser le logiciel en ligne de commande.

Amélie a utilisé un batch script pour créer les dossiers pour chaque scénarios en ajoutant comme source les images de clé USB crées précedemment et pour chaque dossier, il ne reste plus qu'à exécuter les modules, ce qui analyse l'image disque et détecte les fichiers supprimés.

La seule chose qu'il faut faire manuellement a été d'exporter les résultats voulus pour la comparaison. Avec Autopsy, les fichiers supprimés sont classés dans leur propre catégorie et il est facile de les exporter.

La performance en temps d'Autopsy est assez consistante entre 1 et 3 minutes même en changeant le type de fichiers et la réecriture.

Le logiciel parvient a récupérer des fichiers intacts dans la plupart des tests mais si ce n'et pas le cas, il arrive à donner des fragments de fichiers de texte, incomplets cependant. Un simple effacement renvoie la totalité des fichiers effacés peu importe le type de fichiers mais c'est pour la réécriture que cela se complique. Pour les textes, Autopsy parvient à récupérer des fragments lisibles mais pour les images, les fragments sont illisibles. Pour les fichiers cryptés, on parvient à récupérer entre 8 et 17% s'il y a eu réécriture.

En moyenne, Autopsy atteint un taux de 53.54% de fichiers intacts lors de la récupération.

5.4 Encase Forensics

Ce logiciel est commercial et il a donc fallu demander une licence temporaire afin de popuvoir l'utiliser. Contrairement à Autopsy, il n'est pas possible de l'utiliser en ligne de commande et il a donc fallu à Amélie de faire tous les tests manuellemnt, ce qui prend un temps considérable.

Comme les autres logiciel, on peut créer un dossier d'enquête et y ajouter des preuves, ici une image de disque. On peut ensuite procéder à l'analyse des preuves. Il a fallu un certain temps pour comprendre comment ce logiciel fonctionnait puisqu'il n'est pas intuitif du tout. Le manuel n'est pas extrêmement clair et il n'y pas de vraie communauté en ligne.

Encase est un logiciel très rapide dans l'ensemble, il faut simplement ccomprendre quelles manipulations sont à faire pour avoir le résultat attendu. Par exemple, il y a des scripts pré-établi en Enscript pour trouver des documents donc de type txt,docx, ... ou encore des images de type png, jpg, ...

La performance est pltôt bonne lorsque les fichiers ont seulement été effacés et non pas écrasés par une réécriture. Dans le cas d'une réecriture, Encase ne parvient pas à récupérer les fichiers.

En moyenne, Encase atteint un taux de 45.76% de fichiers intacts lors de la récupération.

6 Résultats

Nous avons décidé de numéroter les différents tests selon le format suivant :

— tests 5 à 11 concernent les textes
— tests 20 à 25 concernent les images
— tests 30 à 35 concernent les fichiers cryptés.

Chaque batterie de test incluent les 6 différent scénarios mentionnés 3

Ayant effectué les tests choisis pour chaque logiciel, nous avons rassemblé nos résultats dans une feuille Excel. A partir de ces données, nous pouvons en tirer les graphiques suivants pour visualiser les données.

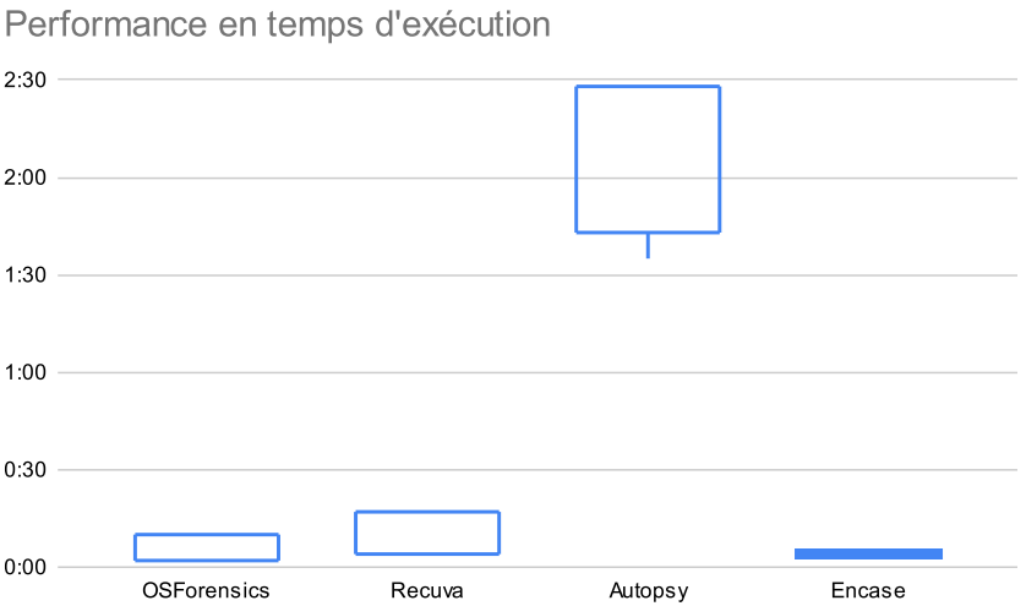

FIGURE 2 – Performance en temps

Ce premier graphique 2 en boite à moustaches montre la disparité temporelle de chaque logiciel. On peut voir que chaque logiciel est plutôt consistant peu importe le test et rapide sauf Autopsy qui prend plus de temps que les autres.

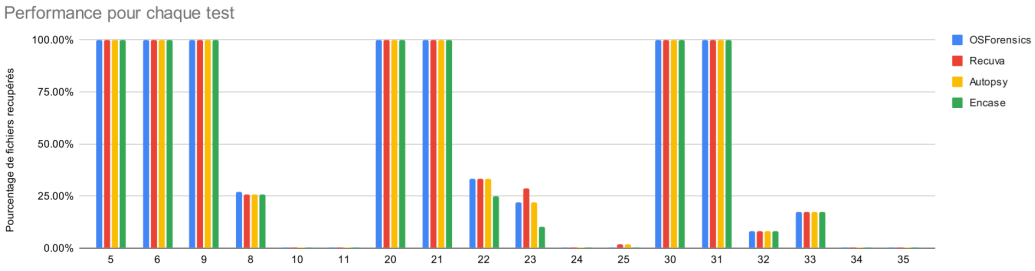

FIGURE 3 – Fichiers récupérés

Ce second graphique 3 est un histogramme qui montre le taux en pourcentage de fichiers récupérés pour chaque test. On peut en déduire que chaque logiciel est efficace pour récupérer des fichiers effacés mais pas écrasés. Dans ce dernier cas, plus les nouveaux fichiers écrasent les traces des fichiers effacés, plus il est compliqué de recupérer les fichiers intacts. On peut parfois

récuperer des fragments cependant. OSForensics est particulièrement efficace pour cela.

	5	6	9	8	10	11	20	21	22	23	24	25	30	31	32	33	34	35
OSForensics	90.00%	88.46%	95.36%	21.07%	0.00%	0.00%	75.00%	68.18%	0.00%	31.57%	0.00%	0.00%	100.00%	100.00%	0.00%	86.67%	0.00%	0.00%
Recuva	100.00%	100.00%	100.00%	29.86%	0.00%	0.00%	75.00%	68.18%	0.00%	24.00%	0.00%	0%	100.00%	100.00%	0.00%	86.67%	0.00%	0.00%
Autopsy	100.00%	100.00%	100.00%	22.99%	0.00%	0.00%	100.00%	100.00%	0.00%	47.37%	0.00%	50.00%	100.00%	100.00%	50.00%	93.33%	0.00%	0.00%
Encase	100.00%	100.00%	100.00%	100.00%	0.00%	0.00%	50.00%	50.00%	50.00%	73.68%	0.00%	0.00%	50.00%	50.00%	50.00%	50.00%	0.00%	0.00%

FIGURE 4 – Fichiers intacts

Ce tableau 4 est coloré pour montrer le pourcentage de fichiers intacts que chaque logiciel parvient à obtenir lors de la récupération. A l'aide d'un format conditionnel, on peut voir que les taux en dessous de 50% sont en rouge car trop faible ; les taux entre 50 et 90% sont en orange ; et les meilleurs taux sont en vert car acceptables.

On peut ainsi voir qu'Autopsy est le logiciel qui parvient à obtenir le taux global le plus haut à 53.54% alors que les autres restent autour de 43à5%.

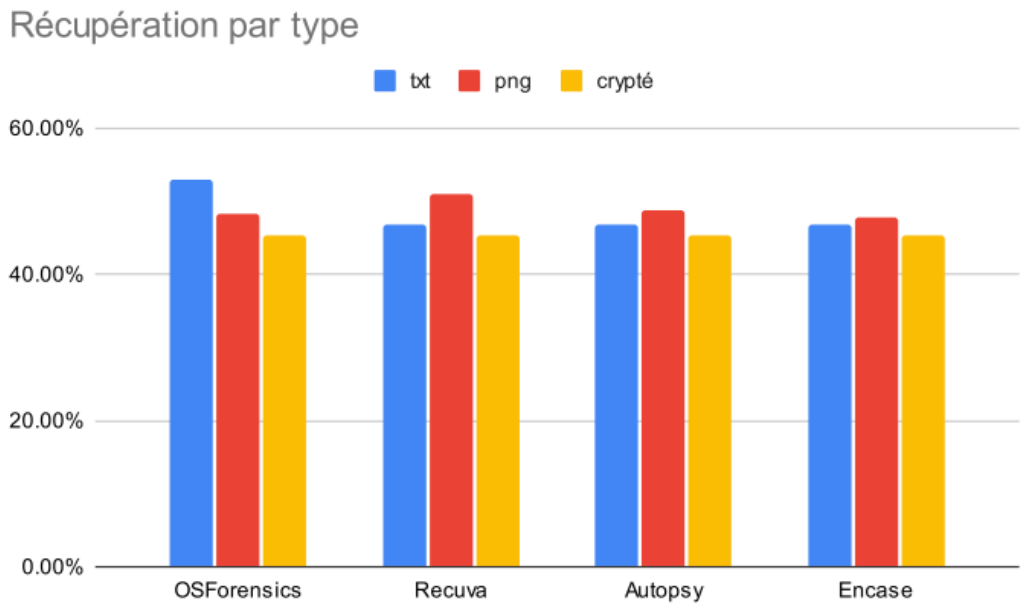

FIGURE 5 –

Cet histogramme 5 montre le taux de récupération par type de chaque logiciel testé. On peut ainsi voir que globalement tous les logiciels parvienent à récupérer les fichiers textes et qu'OSForensics est un peu plus efficace que

les autres. Recuva est particulièrement capable avec les images et ils ont tous le même taux pour les fichiers cryptés.

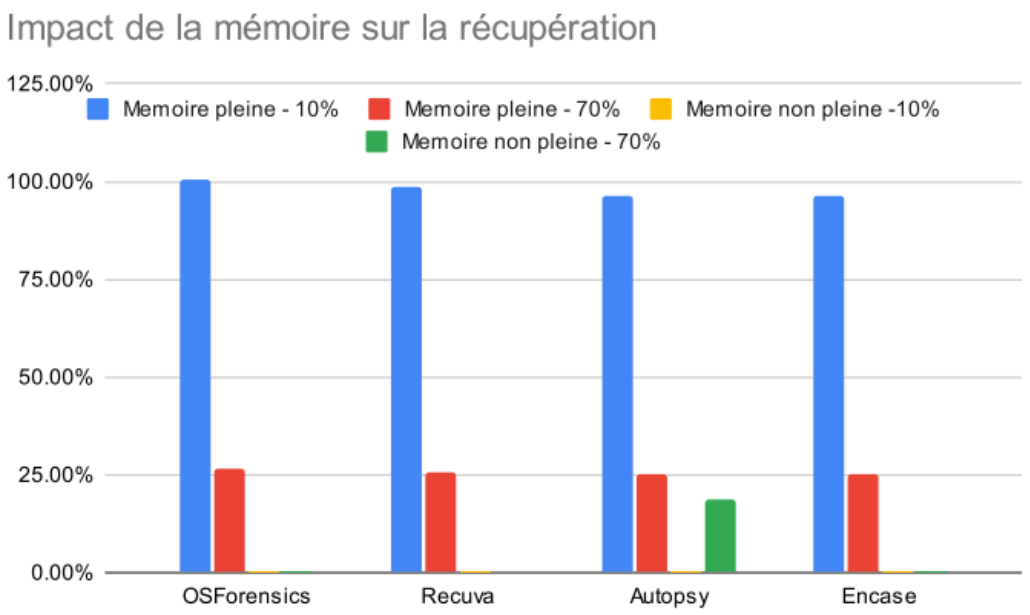

FIGURE 6 –

Ce dernier histogramme 6 montre l'impact de la mémoire sur la récupération. En effet, plus on remplit une clé USB, moins il est possible d'écrire dessus sans écraser des données effacées. L'histogramme montre cependant que dans les test où la mémoire n'était pas pleine, les logiciels n'ont pas réussi à récupérer les données, surtout quand les données effacées étaient moindres (10%).

Les quatre logiciels ont une performance similaire sur cet aspect sauf Autopsy, qui est le seul qui est parvenu à récupérer des données lorsque la mémoire était plein et qu'on avait effacé 70% des données avant recopie.

7 Conclusion

À la suite de cette étude comparative, nous pouvons tirer plusieurs conclusions importantes sur l'efficacité des logiciels de récupération de fichiers supprimés dans différents scénarios. Voici les principaux points à retenir :

— OSForensics s'est avéré très performant, en particulier pour la récupération de fragments de fichiers après réécriture. Son interface est intuitive et son efficacité est notable malgré l'absence de ligne de commande.
— Recuva, bien que facile à utiliser, montre ses limites dans des scénarios complexes. Son interface peu ergonomique et l'absence de certaines fonctionnalités avancées le rendent moins compétitif.
— Autopsy offre une grande fiabilité et une bonne capacité de récupération de fichiers intacts. Cependant, son temps d'exécution plus long peut être un inconvénient dans des situations nécessitant une analyse rapide.
— Encase Forensics est rapide et efficace pour des fichiers simplement supprimés, mais peine à récupérer des fichiers après réécriture. Son interface peu intuitive et la nécessité de manipulations manuelles complexes peuvent poser des défis supplémentaires.

En termes de performance globale, OSForensics et Autopsy se détachent comme les outils les plus complets et efficaces pour la récupération de fichiers supprimés, avec une préférence pour OSForensics en raison de sa rapidité et de son efficacité à récupérer des fragments de fichiers.

Pour le projet G'DIP du GREYC, nous recommandons d'intégrer OSForensics comme outil principal de récupération de fichiers supprimés, avec Autopsy comme alternative, surtout si une analyse approfondie est requise et que le temps d'exécution n'est pas un facteur critique.

8 Propositions d'Améliorations

Pour améliorer cette étude et les futures analyses, nous proposons les actions suivantes :

— Automatisation accrue : Utiliser davantage de scripts pour automatiser les processus dans Autopsy et Encase, ce qui permettrait de réduire le temps et les erreurs humaines.
— Augmenter le corpus de données : Tester avec des fichiers de différents types et tailles, incluant des vidéos, des archives, etc., pour une analyse plus exhaustive.
— Analyse en conditions réelles : Mettre en place des scénarios plus variés et réalistes, comme des manipulations de fichiers par des logiciels malveillants.

— Comparaison de coûts : Intégrer une analyse coût-bénéfice des logiciels, surtout pour les outils commerciaux comme Encase Forensics.
— Formation et documentation : Fournir une formation adéquate et des documents détaillés pour faciliter l'utilisation des logiciels et la reproduction des tests.

Ces améliorations permettraient non seulement d'affiner les résultats obtenus mais également d'élargir le champ d'application de cette étude comparative.

Références

[1] PHOK Amélie BELLET Inès. “Suppressed files - State of the Art”. In : (2023).

[2] Antonis Michalas. *Text files from Gutenberg database*. 2023. URL : https://zenodo.org/records/3360392.

[3] *Sample PNG*. 2023. URL : https://sample-videos.com/download-sample-png-image.php.