Installation de Systèmes Mac OS Classiques et l'Ajout de Plugins Python dans Autopsy

Etudiant:
BEN OMAR Hamid

Tuteurs :
ROSENBERGER Christophe
GERNOT Tanguy
GIGUET Emmanuel

Contents

Chapter 1

Installation des Systèmes Mac OS 7, 8, 9 avec SheepShaver

1 Installation sous Windows

Cette section explore les étapes nécessaires pour installer Mac OS 8 sur un système Windows. Grâce à l'utilisation de l'émulateur SheepShaver, il est possible de faire tourner des applications Mac OS classiques dans un environnement Windows. Ce guide fournit les instructions détaillées pour configurer SheepShaver et installer Mac OS 8, permettant ainsi une expérience utilisateur fluide et intégrée entre les deux systèmes d'exploitation.

1.1 Installation de SheepShaver

SheepShaver est un environnement d'exécution MacOS pour BeOS et Linux qui permet de faire fonctionner des applications MacOS classiques à l'intérieur de l'environnement multitâche de BeOS/Linux. Cela signifie que les applications BeOS/Linux et MacOS peuvent fonctionner simultanément (généralement dans une fenêtre sur le bureau de BeOS/Linux) et que des données peuvent être échangées entre elles. Si vous utilisez un système basé sur PowerPC, les applications fonctionneront à vitesse native (c'est-à-dire sans émulation). Il y a également un émulateur PowerPC intégré pour les systèmes non PowerPC.

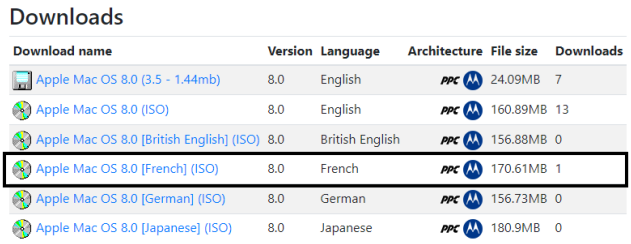
Lien pour installer SheepShaver [Archive](10.4MB), il faut télécharger la dernière version.

Une fois le fichier Zip téléchargé, il faut l'extraire, de préférence le renommer "Mac OS 8". La configuration de l'environement se fera avec l'application SheepShaverGUI qui se trouve dans le répertoire.

Avant de passer à la configuration de SheepShaver, il convient de sélectionner la version de MacOS à installer.

Émulation de MacOS 8

- Page de téléchargement de l'ISO de Mac OS 8 [Archive](160.89MB)



- Un fichier ROM valide : Old World 4mb ROM[Archive](2MB).

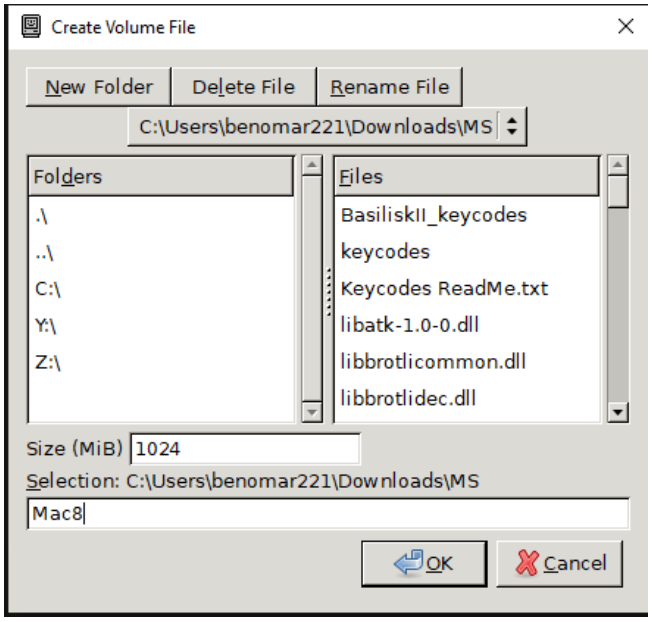
  Note : Avec une ROM Old World, vous pouvez exécuter Mac OS 7.5.3 jusqu'à Mac OS 9.0.4. Avec une ROM New World, vous pouvez exécuter Mac OS 8.5 jusqu'à Mac OS 9.0.4. On

Il est conseillé d'extraire ces fichiers et de les placer dans le repértoire de SheepShaver, ceci facilitera la configuration.

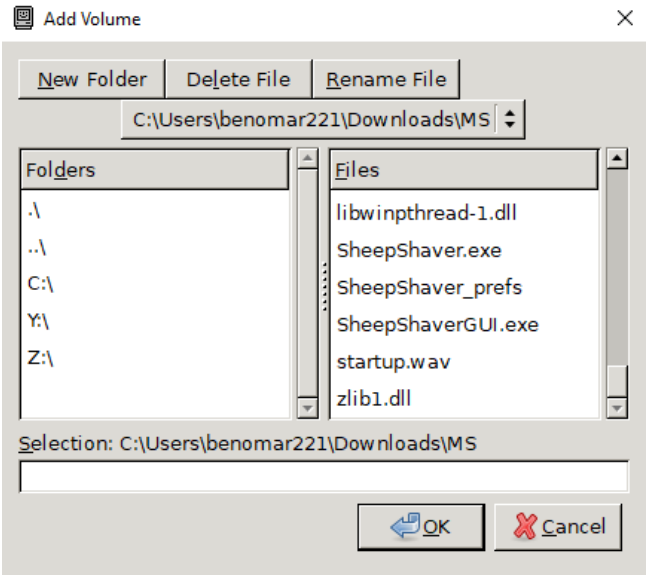
1.2 Configuration de SheepShaver

L'onglet Volume:

Il faut créer un Volume file de taille 1024 Mb en cliquant sur Create, de préférence le nommer "Mac8" :

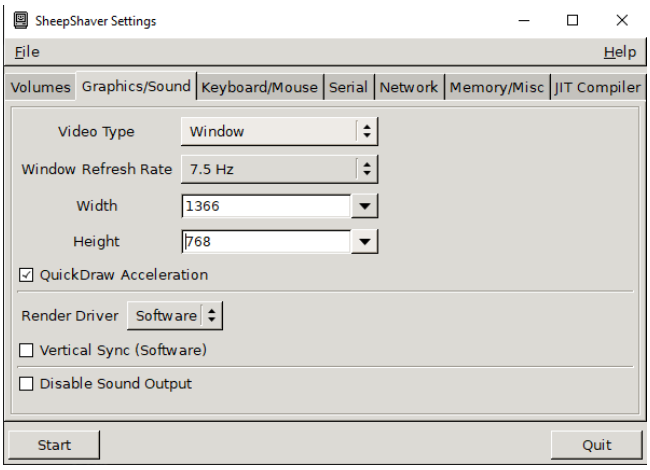


Ensuite on ajoute notre fichier ISO en cliquant sur Add et en choisisant l'iso qui se trouve déja dans le répértoire, il suffit de chercher dans Files



L'onglet Graphics/Sound:

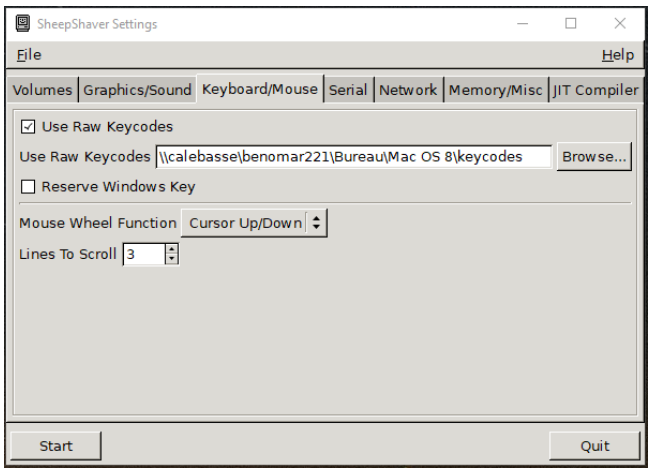
Les paramètres Width et Height doivent être changés en fonction de la taille que vous souhaitez pour la fenêtre. En les mettant a maximum, l'émulateur se lance en plein écran.



L'onglet Keyboard/Mouse

Note: lorsque vous utilisez un clavier non US-EN, vous avez besoin d'un fichier de codes de touches pour mapper la disposition de votre clavier à celle de Mac OS. Un fichier de codes clés.(3KB)

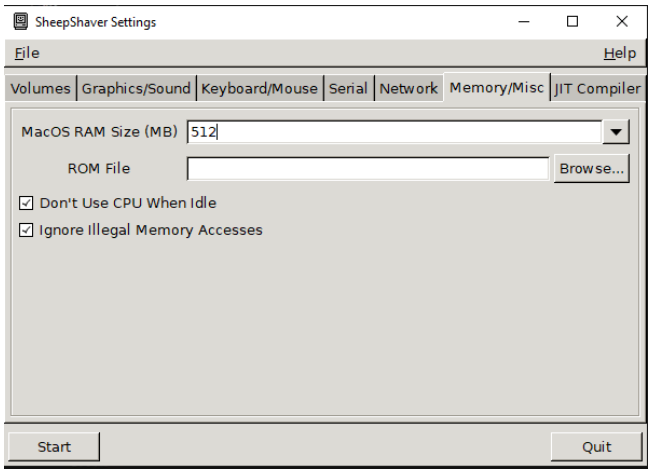
Copier le fichier keycodes contenu dans le fichier zip dans le répertoire courant. Puis sélectionner la case "Use Raw Keycodes" et ajouter le fichier keycodes :



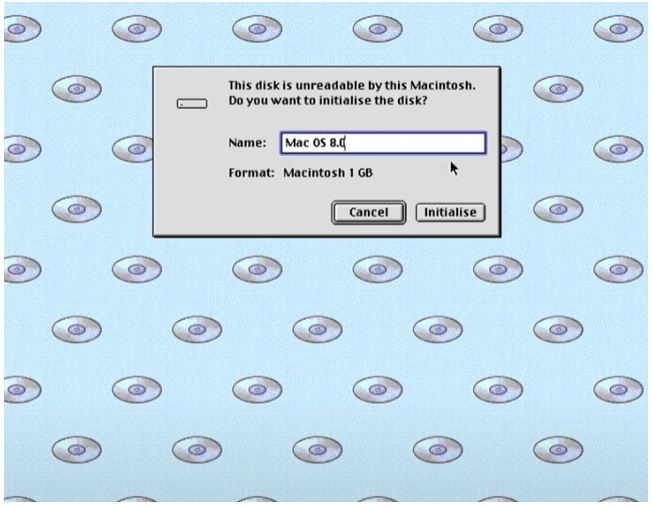
L'onglet Memory/Misc

La valeur de MacOS RAM size doit être 512 Mb.

On ajoute le fichier ROM qu'on a mis dans le répertoire, il suffit de fouiller dans Files

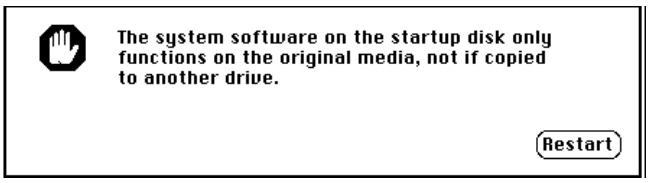


La configuration est terminée, on peut cliquer sur start. Cette interface s'affiche pour finaliser la configuration :

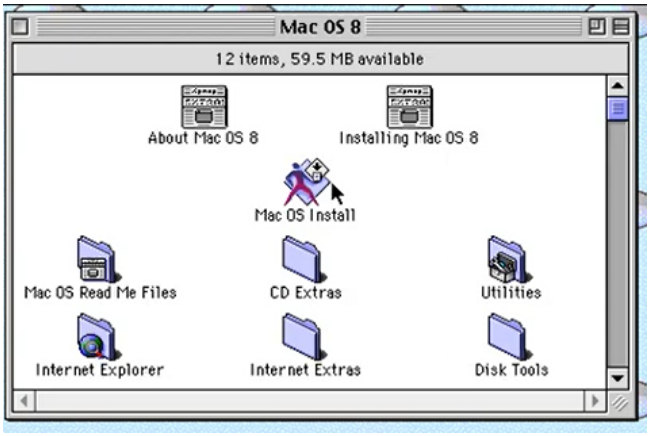


on renomme le disk à "Mac 8" et on clique sur initialise.

si vous rencontrez l'erreur suivante :



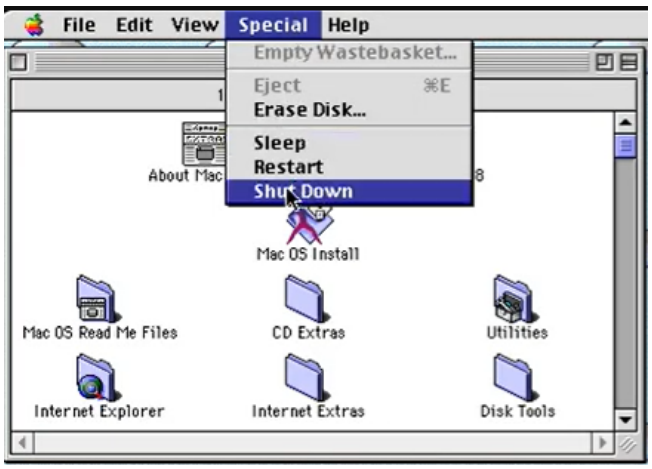
il suffit de mettre le fichier de l'iso en mode "read-only", en allant sur "proriétés" du fichier et cocher l'attribut lecture seule



Aprés l'apparition de cette fenêtre, il suffit de lancer le programme Mac OS install

On change la destination disk à Mac 8 et on clique sur select » continue » continue » agree » start

Quand l'installation est terminée, il faut redémarer le système :



La configuration est terminée, Mac OS 8.0 est prêt à être utilisé.

2 Installation sous Linux

Pour mieux organiser l'installation et la configuration de SheepShaver, il est préférable de créer le propre dossier de SheepShaver :

```
mkdir -p ~/SheepShaver
cd ~/SheepShaver
sudo sysctl -w vm.mmap_min_addr=0
```

Toutes les commandes doivent être éxecutées dans le dossier /SheepShaver

2.1 Installation de SheepShaver

Téléchargez la dernière version de SheepShaver[Archive](24.8 MB) adaptée à votre distribution Linux (64 bits ou 32 bits) :

```
curl -L -o SheepShaver-x86_64.AppImage https://github.com/
Korkman/macemu-appimage-builder/releases/download/continuous/
SheepShaver-x86_64.AppImage
```

Assurez-vous que l'exécutable a les permissions nécessaires pour être exécuté. Vous pouvez utiliser la commande suivante :

```
chmod +x SheepShaver-x86_64.AppImage
```

2.2 Configuration de SheepShaver sous Linux

Préparations

Tout d'abord, assurez-vous d'avoir téléchargé les fichiers nécessaires :

- Un fichier ROM valide : la ROM Old World[Archive](2MB).

  Note : Avec le fichier ROM Old World, SheepShaver peut exécuter System 7.5.3 jusqu'à Mac OS 9.0.4 ; avec le fichier ROM New World, SheepShaver peut exécuter Mac OS 8.5 jusqu'à 9.0.4.

```
curl -L -o  mac_oldworld_rom4mb.rom.zip
https://smb4.s3.us-west-2.amazonaws.com/sheepshaver/
apple_roms/mac_oldworld_rom4mb.rom.zip

unzip mac_oldworld_rom4mb.rom.zip -d ~/SheepShaver
```

- L'ISO de Mac OS 8[Archive](160.89MB):

```
curl -L -o Apple_Mac_Os_ISO_8.7z https://winworldpc.com/
download/c3990b24-3025-c382-11c3-a6e280947e52/from/
c39ac2af-c381-c2bf-1b25-11c3a4e284a2

sudo apt install p7zip-full

7za x Apple_Mac_Os_ISO_8.7z
```

Configuration de SheepShaver

1. installer l'application SheepShaver avec la commande

```
./SheepShaver-x86_64.AppImage --install
```

2. L'application SheepShaver sera créé dans le menu des applications, après avoir l'ouvert, SheepShaver settings GUI apparaît pour configurer l'émulateur
3. Suivez les étapes de configuration de SheepShaver comme décrit précédemment dans la section 1.2 pour Windows.

Une fois la configuration terminée, vous pouvez démarrer SheepShaver et commencer à utiliser Mac OS 8.0 sur votre système Linux.

Pour les utilisateurs qui préfèrent lancer SheepShaver directement depuis le terminal sans passer par l'interface graphique, il est possible de démarrer l'émulateur avec une seule commande en spécifiant les différents chemins nécessaires pour le disque, l'image ROM et autres fichiers de configuration. Par exemple, la commande suivante permet de démarrer SheepShaver avec Mac OS 8.0:

```
./SheepShaver-x86-64.AppImage --disk "/chemin/vers/votre/disk/Mac8"

--disk "/chemin/vers/votre/image/ISO/MacOS80-FR.iso"

--rom "/chemin/vers/votre/rom/mac-oldworld-rom4mb.rom"

--keycodes true --keycodefile "/chemin/vers/votre/keycodes/keycodes"

--ramsize 536870912 --nogui true
```

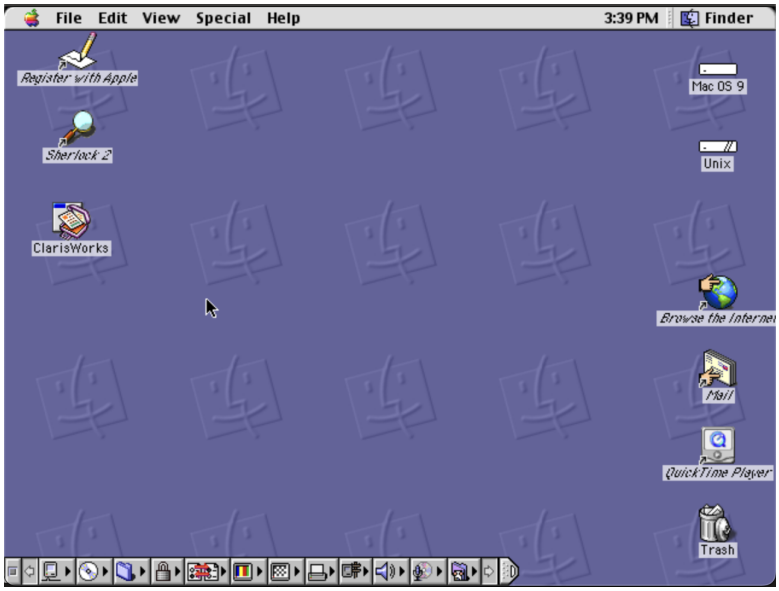
Assurez-vous de remplacer les chemins dans la commande ci-dessus par les chemins corrects correspondant à votre configuration. Cette méthode permet un lancement rapide de l'émulateur en mode console, sans afficher l'interface graphique de configuration

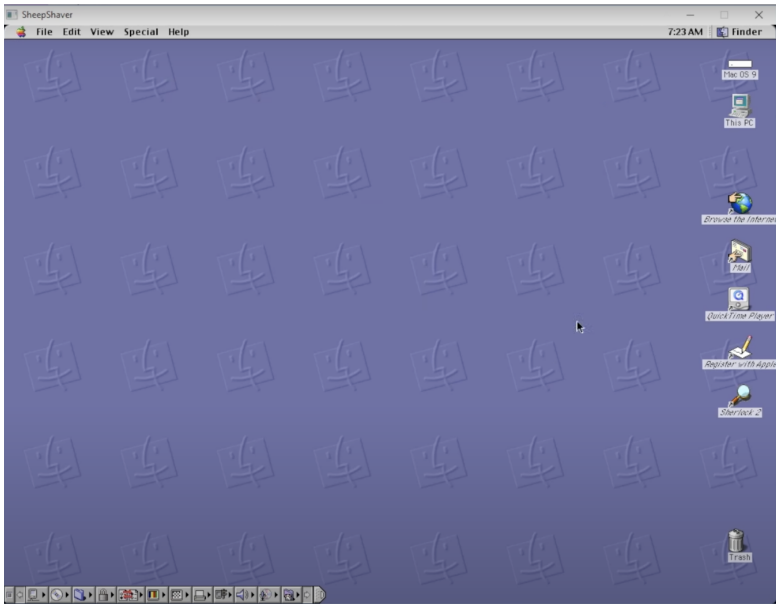
3 Installation pour d'autres versions de Mac OS

Pour les versions de Mac OS allant de 7.5.3 à 9.0.4, la procédure d'installation reste identique à celle décrite pour Mac OS 8.0. Il vous suffit de télécharger l'ISO correspondant à la version souhaitée depuis le site suivant : WinWorldPC - Operating Systems [Archive]. Assurez-vous de suivre les mêmes étapes d'installation pour configurer SheepShaver avec l'image disque de votre choix.

4 Installation d'applications dans SheepShaver

Une fois SheepShaver configuré, un disque UNIX apparaît sous Linux (ou "Ce PC" sous Windows), permettant de partager des dossiers et fichiers avec l'ordinateur hôte.



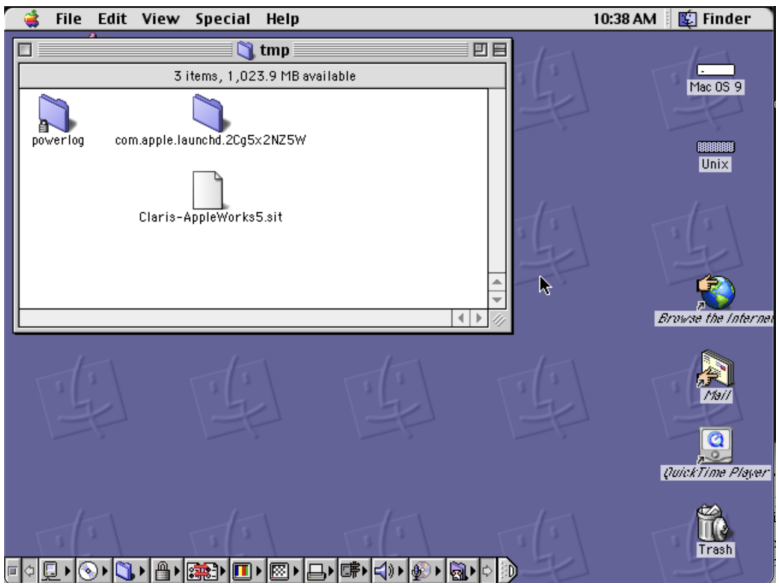


Pour installer des applications telles que ClarisWorks ou MacWrite, les étapes suivantes doivent être suivies :

Téléchargement des fichiers d'installation : Les fichiers d'installation des applications désirées, généralement sous forme de fichiers avec l'extension .sit, peuvent être téléchargés à partir de sites comme Macintosh Repository. [Archive]

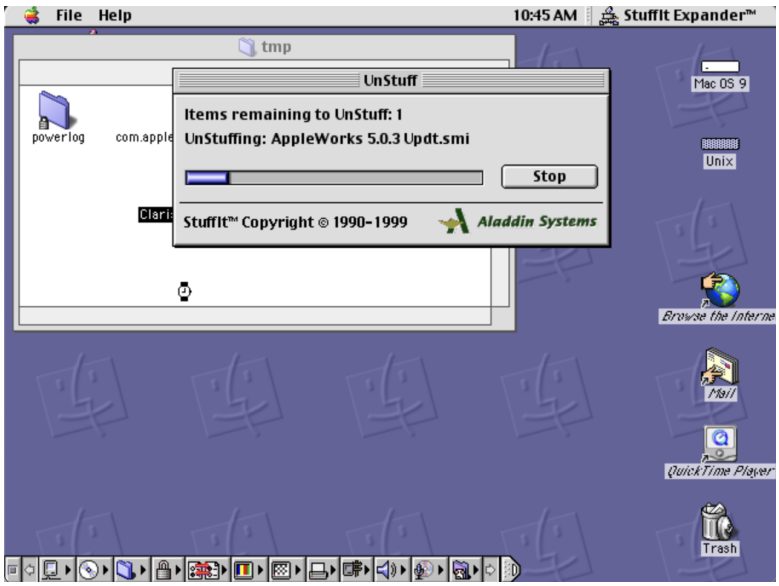
- Claris Works [Archive](7.31 MB)
- MacWrite [Archive] (5.68 MB)
- MS Word [Archive](4.62 MB)

Transfert vers SheepShaver : Les fichiers .sit téléchargés doivent être placés dans un dossier accessible via le disque UNIX. Cela permet de transférer facilement les fichiers de l'ordinateur hôte vers l'environnement émulé.



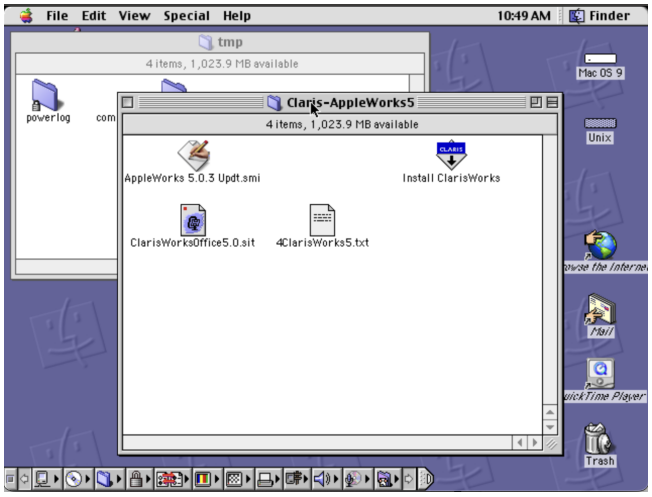
Installation via SheepShaver : Accès au fichier d'installation : Une fois SheepShaver lancé, l'utilisateur peut accéder au disque UNIX depuis l'environnement émulé. Le fichier .sit doit être localisé dans ce disque.

Décompression et installation : Le fichier .sit est ouvert à l'aide de StuffIt Expander, une application couramment utilisée dans l'environnement MacOS émulé, qui est par défaut déjà installé.



StuffIt Expander décompresse le fichier et crée un dossier dans le même répertoire. Ce dossier contient l'application décompressée ou son installateur.

Finalisation de l'installation L'application est alors prête à être utilisée. Si un installateur est présent, il devra être exécuté pour terminer l'installation.



Sinon, l'application peut être lancée directement depuis le dossier créé.

En suivant ces étapes, l'installation d'applications dans SheepShaver est simplifiée, permettant d'enrichir l'environnement émulé avec des logiciels historiques compatibles avec MacOS classique.

Chapter 2

Installation de Mac OS 6 avec Mini vMac

Mini vMac est un émulateur permettant de faire tourner les systèmes Mac OS versions 7.5 et antérieures, comme Mac OS 6 et les versions précédentes.

1 Configuration de mini vMac

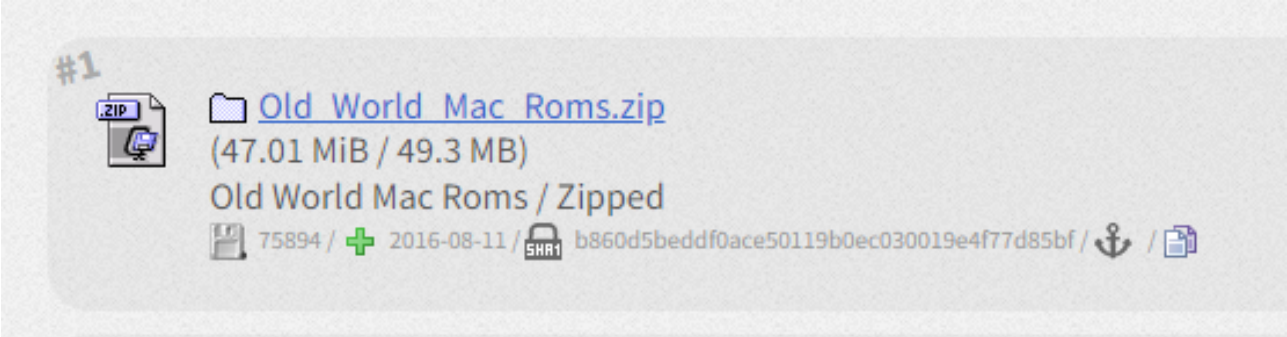
Téléchargez la dernière version de mini vMac adaptée à votre distribution sur le site officiel : Mini vMac Downloads[Archive](59 KB).

Il est préférable de créer un dossier propre pour mini vMac, et de placer tous les téléchargements associés dedans.

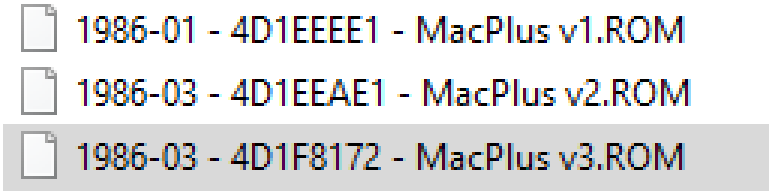
Extrayez le fichier Mini vMac téléchargé, et lancez l'exécutable, vous obteinderai le message

```
Unable to locate ROM image
```

La ROM du Mac Plus est nécessaire pour faire fonctionner Mini vMac. il est téléchargeable depuis le site Macintosh ROMs[Archive](49.3 KB)

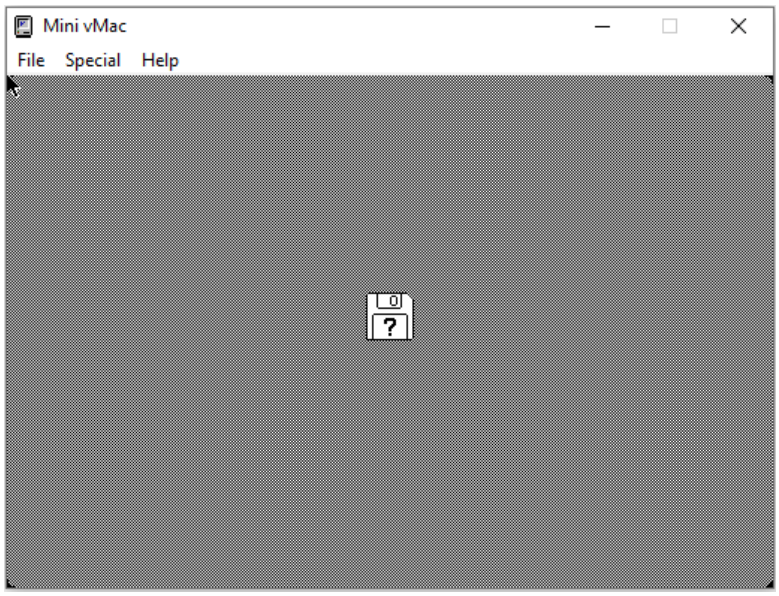


Ce zip contient une centaine de fichiers ROMs, mais on s'interesse au ROMS se trouvant dans le dossier 128KB ROMs, on s'interesse particulièrement à la version 3 :

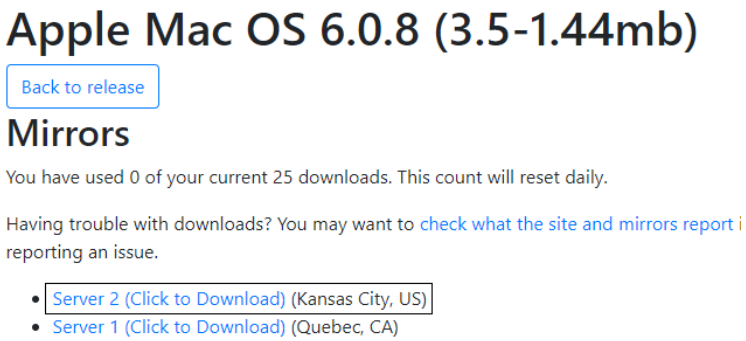


On copie ce fichier dans le dossier contenant l'exécutable mini vMac, il est nécessaire de le renommer "vMac.ROM".

Mini vMac n'affichera plus le message **"Unable to locate ROM image"**, et l'interface ressemble a ceci :

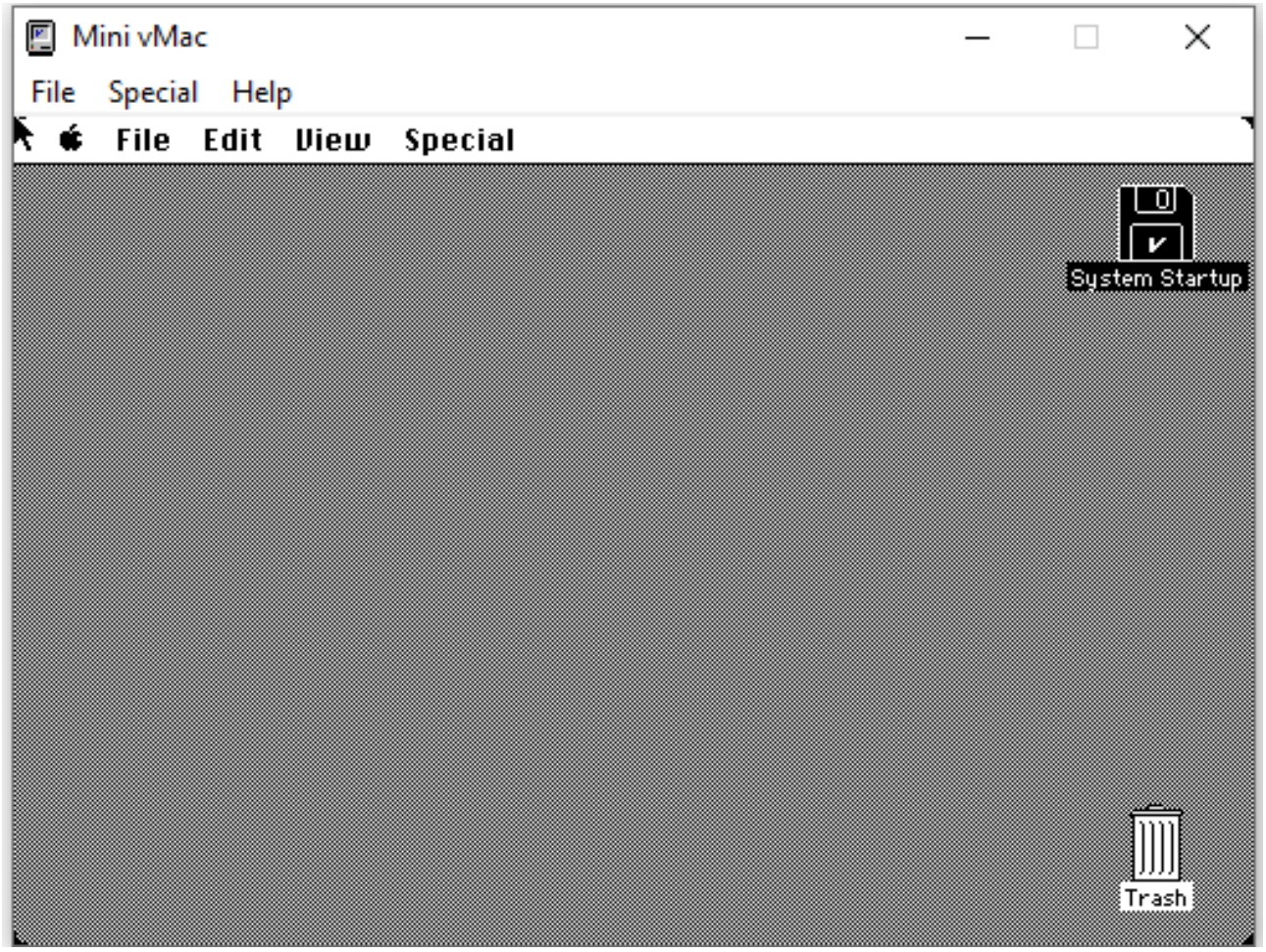


Les fichiers d'installation du système Mac OS 6 sont nécessaires, il sont téléchargeables depuis le site WinWorld Mac OS[Archive]



Une fois le fichier téléchargé extrait, les deux fichiers d'extension ".img" sont ceux qui permettent l'installation du système Mac Os 6.

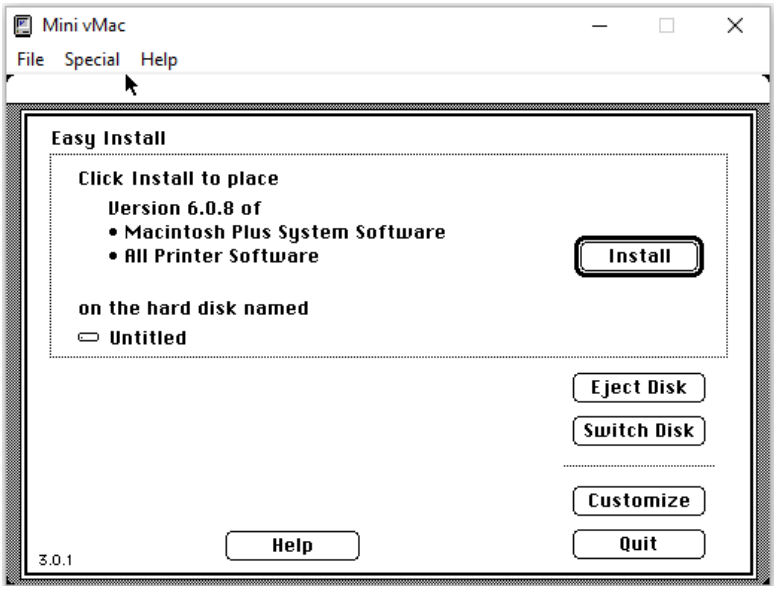
Lancez Mini vMac et faites glisser "System Startup.img" dans la fenêtre de l'émulateur. Cela démarrera le système.



Ouvrez System Startup et le lancez le "installer"

On aura besoin des images de disques vides, téléchargeables sur le site Gryphel Blanks[Archive](54 Kb), le zip téléchargé contient plusieurs disk images de tailles différentes, on prend par exemple celle de 224Mb qui se trouve dans le dossier extrait » M » 224M.zip, on extrait le fichier 224M.dsk et on le met dans le dossier contenant mini vMac.

Faites glisser 224M.dsk dans la fenêtre de l'émulateur déja ouverte, et cliquez sur ok.



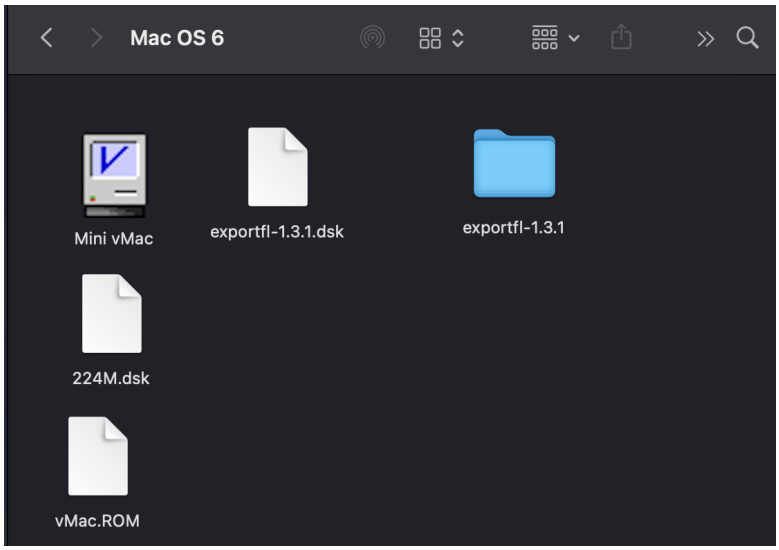
Appuyez sur le bouton 'Installer' pour lancer l'installation de Mac OS 6.

À un certain moment, Mini vMac demandera l'insertion de System Additions.img. Faites glisser ce fichier dans la fenêtre de Mini vMac pour continuer l'installation.

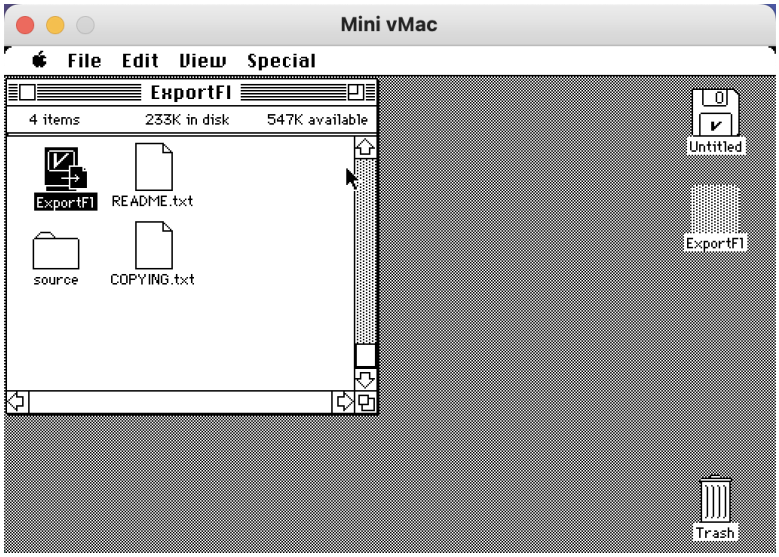
L'installation est terminée et Mac OS 6 est prêt à être utilisé, à chaque fois que l'on souhaite le lancer, il suffit de lancer mini vMac et de faire glisser 224M.dsk dans la fenêtre de l'émulateur.

2 Installation d'applications dans mini vMac

Le site Gryphel propose des applications utils pour mini vMac, il suffit de chercher par catégorie sur ce site et d'installer le zip correspondant. L'application ExportFl est un exemple :

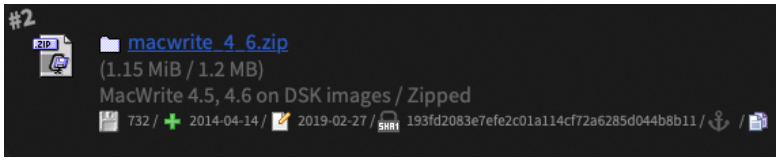


Une fois mini vMac lancé, il suffit de faire glisser le fichier exportfl-1.3.1.dsk dans la fenetre de mini vMac.



L'application est déjà installé, il suffit de la faire glisser sur le bureau pour l'utiliser.

le Site Macintosh Repository propose parfois des applications compatibles avec mini vMac mais il faut trouver un fichier image DSK



Chapter 3

Constitution de la base de Hash

1 Pour Mac Os 7-8-9

Afin de constituer une base de hashs MD5 des applications et fichiers téléchargés par défaut par SheepShaver, il faut copier tous les fichiers existants dans un dossier créé, que l'on récupère sur l'ordinateur hôte via le dossier partagé "unix" sous Linux ou "Ce PC" sous Windows (section 4 Transfert vers SheepShaver).

Ensuite, il suffit de lancer ce script Python en spécifiant le chemin correct vers le dossier récupéré et en choisissant l'algorithme MD5 comme algorithme de hachage.

```python
import os
import hashlib

def calculate_hash(file_path, algorithm='sha256'):
    """Calculate the hash of a file."""
    hash_func = hashlib.new(algorithm)
    with open(file_path, 'rb') as f:
        while chunk := f.read(8192):
            hash_func.update(chunk)
    return hash_func.hexdigest()

def get_all_files(directory):
    """Recursively get all files in the given directory, ignoring hidden directories."""
    file_paths = []
    for root, dirs, files in os.walk(directory):
        # Ignore hidden directories
        dirs[:] = [d for d in dirs if not d.startswith('.')]
        for file in files:
            file_paths.append(os.path.join(root, file))
    return file_paths

def format_output(files, root_directory, algorithm='md5'):
    """Format the output to include directory structure and file hashes.
    """
    output_lines = []
    for file_path in files:
        relative_path = os.path.relpath(file_path, root_directory)
        file_hash = calculate_hash(file_path, algorithm)
```

```
28          output_lines.append(f"{relative_path} : {file_hash}")
29      return output_lines
30
31  def save_output(output_lines, output_file='file_hashes.txt'):
32      """Save the formatted output to a file."""
33      with open(output_file, 'w', encoding='utf-8') as f:
34          for line in output_lines:
35              f.write(line + "\n")
36
37  def main(directory, algorithm='md5', output_file='file_hashes.txt'):
38      """Calculate and save the hash of each file in the directory."""
39      files = get_all_files(directory)
40      output_lines = format_output(files, directory, algorithm)
41      save_output(output_lines, output_file)
42
43  if __name__ == "__main__":
44      directory = input("Enter the directory path: ")
45      algorithm = input("Enter the hash algorithm (default is md5): ") or '
            md5'
46      output_file = input("Enter the output file name (default is
            file_hashes.txt): ") or 'file_hashes.txt'
47      main(directory, algorithm, output_file)
```

2 Pour Mac OS 6

L'émulateur Mini vMac ne permet pas d'échanger les fichiers donc l'automatisation de calcul des Hashs s'avère difficile. Pour cela il faut télécharger l'application ExportFl qui permet d'exporter des fichiers depuis mini vMac vers l'ordinateur hôte.

Une fois tous les fichiers et applications installés par Mini vMac copiés dans un dossier créé sur l'ordinateur hôte, il suffit de lancer le script python de la section 1

Chapter 4

Ajout des plugins pythons dans Autopsy

1 Environnement de Travail

L'utilisation d'Autopsy pour l'analyse forensique numérique implique des étapes spécifiques pour configurer l'environnement de travail en fonction du système d'exploitation utilisé. Les procédures varient entre Windows, macOS et Linux.

Autopsy utilise "jython" pour permettre le "python scripting". le code écrit en python se convertit en Java byte code et s'éxecute sur la JVM. Jython a quelques limitations:

- limité à la version 2.7 de python
- n'utilise pas les librairies Python qui contiennent du code natif (Numpy,SciPy, ...)

1.1 Travailler sous windows

Rien n'est plus simple que d'installer Autopsy sur windows, il suffit de télécharger la dernière version depuis le site officiel : Download Autopsy

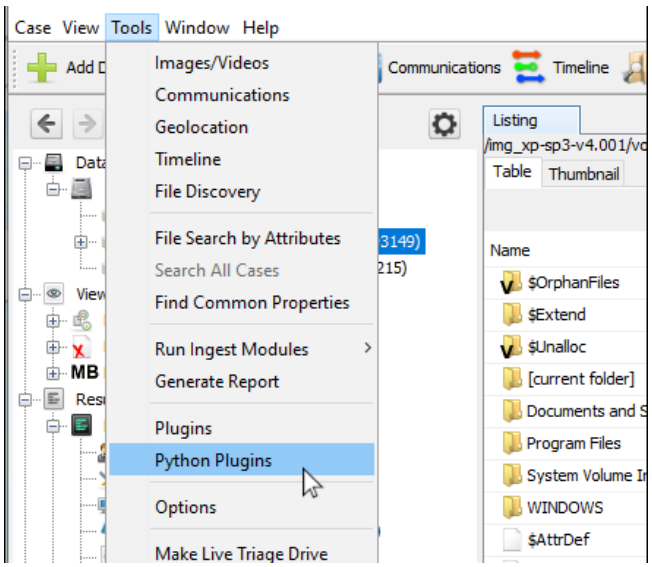
Assurez-vous de télécharger un version de Python supérieure à 2.7 pour éviter les conflits de versions.

1.2 Travailler sous Mac OS/Linux

Malheuresement pour ces distributions, l'installation n'est pas aussi facile que pour windows, il faut suivre les instructions disponibles sur le gitlab d'Autopsy.

2 Ajout de plug-in python

Pour ajouter un plugin Python, il suffit d'ouvrir le dossier des plugins en allant sur Tools » Python Plugins depuis le menu.



Dans le dossier qui apparaît, créez un nouveau dossier en le nommant en fonction de ce que votre plugin fera.

Chaque plugin doit avoir son propre dossier. Ceci est pour éviter les conflits de noms des fichiers d'extension py.

Une fois le fichier python est placé dans le dossier, on peut le lancer depuis le menu en allant à Tools » Run Ingest Modules et en choisissant la data source sur laquelle on veut appliquer le plug-in.

Un tutoriel plus détaillé est disponible sur sleuthkit.

Plusieurs plug-ins sont disponible sur le github de markmckinnon

3 Créer son propre plug-in

il existe trois types de plug-in à developper :

- file ingest module tutoriel
- data source ingest module tutoriel
- report module tutoriel

File Ingest Module :

Le module d'ingestion de fichiers (File Ingest Module) dans Autopsy est utilisé pour importer et analyser des fichiers spécifiques ou des répertoires à partir d'une source de données. Ce module permet à l'utilisateur d'extraire des fichiers d'intérêt à partir d'un ensemble de données plus large, en filtrant les types de fichiers ou en utilisant des critères spécifiques. Par exemple, vous pourriez utiliser ce module pour extraire uniquement des images, des documents ou d'autres types de fichiers à des fins d'examen plus approfondi.

Data Source Ingest Module :

Le module d'ingestion de sources de données (Data Source Ingest Module) est plus large que le module d'ingestion de fichiers. Il permet à Autopsy d'importer et d'examiner des sources de données entières, telles que des disques durs, des images de disques, des systèmes de fichiers ou des appareils

mobiles. Ce module est utilisé pour importer des ensembles de données complets dans Autopsy afin de mener des analyses forensiques sur des systèmes de fichiers entiers ou des dispositifs de stockage.

Report Module :

Le module de génération de rapports (Report Module) est utilisé pour créer des rapports détaillés à partir des résultats d'analyse effectués dans Autopsy. Une fois qu'Autopsy a terminé l'examen d'une source de données (grâce aux modules d'ingestion mentionnés ci-dessus et à d'autres fonctionnalités d'analyse), le module de génération de rapports permet à l'utilisateur de compiler les conclusions, les résultats d'extraction de données, les métadonnées et d'autres informations pertinentes dans un rapport structuré. Ces rapports peuvent être utilisés pour documenter les résultats de l'examen et sont souvent nécessaires dans les enquêtes judiciaires ou lors de la communication des résultats à d'autres parties prenantes.

Nous allons par la suite se concentrer sur le file ingest module, qui est le type de plug-in le plus utilisé.

Pour developper son propre plug-in, une structure est imposée pour que le plug-in soit fonctionnel

Le patron de conception *Factory* est mis en place, la première class Factory définie le nom et les détails du module, ainsi elle permet à Autopsy de créer des instances du module pour faire l'analyse

La deuxième class est celle où la logique du module est définie, trois méthodes sont necéssaires pour assurer le fonctionnment du module

- startUp
- process
- shutDown

La méthode **startUp** est appelée lors de l'initialisation du module d'ingestion de fichiers. C'est généralement là que le module met en place ses ressources et configure ses composants nécessaires pour démarrer le processus d'ingestion.

La méthode **process** est le cœur du module d'ingestion de fichiers. Elle est appelée pour traiter les tâches d'ingestion spécifiques qui ont été soumises au module. Pendant l'exécution de cette méthode, le module travaille sur chaque tâche d'ingestion en séquence ou de manière parallèle, en utilisant ses analyseurs de fichiers pour examiner et extraire des informations à partir des fichiers importés.

La méthode **shutdown** est appelée lorsque le module d'ingestion de fichiers est prêt à se terminer ou à nettoyer ses ressources après avoir terminé le traitement des tâches d'ingestion. Cette méthode est responsable de la libération des ressources allouées et de la finalisation du module.

Le débogage dans Autopsy peut être complexe en raison de la nature des analyses forensiques et des volumes de données souvent importants. Pour aider les développeurs à diagnostiquer et résoudre les problèmes, Autopsy utilise des outils de journalisation (logging) via un système de logs. Pour déboguer ou voir le résultat d'une variable, on utilise la méthode log définie dans la classe IngestModule

3.1 plug-in paramètrable

Afin de créer un plug-in paramètrable, il faut intéragir avec l'utilisateur à travers l'interface graphique d'Autopsy. L'interface graphique utilise la libraire javax.swing, et peut se composer de : checkbox, boutton, text field, panel ...

Une troisième classe doit être implémentée. Il est recommandé de la nommer avec le même nom que la classe factory, en remplaçant "factory" par "SettingsPanel". C'est ici que l'initialisation des éléments de l'interface graphique se fait. Une méthode qui crée une instance de cette classe doit être implémentée dans la classe factory.

l'ensemble de ces méthodes est détaillé dans cet exemple

4 GDIP Indoor-Outdoor

Pour l'intégration du filtre d'images indoor/outdoor dans Autopsy, deux approches distinctes ont été explorées.

La première approche consiste à utiliser directement les scripts Python fournis. Ces scripts sont exécutés depuis le module de traitement des fichiers (file ingest module) d'Autopsy. Lorsqu'une image est ingérée, le module exécute le script correspondant pour déterminer si l'image a été prise en intérieur ou en extérieur. Les résultats de cette analyse sont ensuite récupérés et utilisés pour classer les images dans le blackboard d'Autopsy. Cette méthode s'est avérée plus rapide lors des tests sur la même base de données d'images, car l'exécution directe des scripts évite les surcharges supplémentaires introduites par la transformation en exécutable. Cependant, un inconvénient notable est que le code source des scripts est accessible, ce qui pose des problèmes de sécurité potentiels, et elle nécessite que toutes les dépendances Python soient correctement installées sur le système.

La deuxième approche implique la transformation des scripts Python en exécutables à l'aide de PyInstaller. Ces exécutables sont ensuite intégrés dans le module de traitement des fichiers d'Autopsy. À chaque fois qu'une image est ingérée, Autopsy lance l'exécutable correspondant, qui analyse l'image pour déterminer si elle a été prise en intérieur ou en extérieur. Les résultats de l'analyse sont récupérés après l'exécution de l'exécutable et utilisés pour classer les images dans le blackboard. Cette méthode offre une meilleure sécurité car le code source est encapsulé dans l'exécutable, réduisant ainsi le risque de modification non autorisée. De plus, les dépendances Python peuvent être emballées dans l'exécutable, évitant ainsi les problèmes de configuration de l'environnement Python sur le système cible. Cependant, cette approche est généralement plus lente, car le lancement d'un exécutable introduit une surcharge supplémentaire, et les exécutables peuvent être volumineux en termes d'espace mémoire, surtout s'ils incluent toutes les dépendances nécessaires.

4.1 Evaluation du temps d'exécution

Pour évaluer les performances des deux approches développées pour le filtre d'images indoor/outdoor dans Autopsy, des tests ont été réalisés sous Windows et Linux en mesurant le temps d'exécution pour traiter 1911 images sur une machine 8 Go de RAM.

Sous Windows, l'approche utilisant des l'exécutable a pris 1 heure et 50 minutes pour traiter les 1911 images, tandis que l'exécution directe des scripts Python (code natif) a pris 1 heure et 25 minutes. Sous Linux, les temps d'exécution étaient plus courts pour les deux approches : 1 heure et 33 minutes pour l'exécutable et 1 heure et 15 minutes pour le code natif.

L'écart de performance est plus prononcé sous Windows, où le temps supplémentaire requis pour les exécutables est plus significatif par rapport à Linux. Cela peut être attribué à la surcharge introduite par le lancement des exécutables et à la taille des fichiers d'exécution, qui incluent toutes les dépendances nécessaires. En revanche, le code natif, bien que nécessitant une configuration appropriée de l'environnement Python, s'avère plus efficient en termes de temps d'exécution.

Ainsi, si la priorité est la rapidité d'exécution, l'approche basée sur l'exécution directe des scripts Python est préférable. Cependant, si la sécurité et la facilité de déploiement sont des critères prioritaires, l'approche utilisant des exécutables pourrait être plus appropriée, malgré le compromis sur la performance.

4.2 Évaluation de l'Empreinte Finale des Fichiers Exécutables

En plus de l'évaluation des temps d'exécution, l'empreinte finale des fichiers exécutables a été analysée. Voici les aspects pris en compte :

Taille des Fichiers Exécutables :

Sous Windows, le fichier exécutable généré avec PyInstaller a une taille de 370 MB. Sous Linux, le fichier exécutable généré a une taille de 3 GB !!!!!

Dépendances Incluses :

Les exécutables incluent toutes les bibliothèques Python nécessaires au fonctionnement du script, telles que OpenCV, NumPy et d'autres dépendances spécifiques.

Consommation de Mémoire :

Lors de l'exécution, les exécutables sous Windows consomment en moyenne 150 MB de RAM, tandis que sous Linux, la consommation moyenne est de 130 MB.

Performance d'Exécution :

En termes de ressources CPU, les exécutables montrent une utilisation moyenne de 25 pourcent.

Ces analyses montrent que, bien que les exécutables offrent une meilleure sécurité et une facilité de déploiement, ils introduisent une empreinte significative en termes de taille et de ressources système par rapport à l'exécution directe des scripts Python. Cela doit être pris en compte lors du choix de la méthode à utiliser en fonction des contraintes et des priorités du projet.