

# MALWARE HUNTER

## Analyse – Conception

### PAGE DE SERVICE

Référence : N/A

Plan de classement :

Niveau de confidentialité : corporate

Mises à jour

Version	Date	Auteur	Description du changement
1.1	03/02/2023	NIETO Wylan	Conception <i>Lire un fichier</i>
1.2	10/02/2023	NIETO Wylan	Parcourir les répertoires, Scanner chaque fichier
1.3	18/02/2023	NIETO Wylan	Lire la réponse de l'API
1.4	19/02/2023	NIETO Wylan	Traiter les alarmes, finitions

Validation

Version	Date	Nom	Rôle
---------	------	-----	------

Diffusion

Version	Date	Nom	Rôle
---------	------	-----	------

### SOMMAIRE

#### Table des matières

PAGE DE SERVICE .....	0
SOMMAIRE .....	0
1 CAS N°1 .....	1
1.1 RAPPEL DU CONTEXTE .....	1
1.2 OBJECTIFS .....	1
1.3 HOME PAGE FIREFOX .....	1
2.1 OBJECTIFS .....	2
2.2 VIRUSTOTAL .....	2
2.3 LIRE UN FICHIER .....	3
2.4 PARCOURIR LES REPERTOIRES .....	3
2.5 SCANNER CHAQUE FICHIER .....	4
2.6 LIRE LA REPONSE DE L'API .....	6
2.7 TRAITER LES ALARMES .....	7
2.8 CONCLUSION .....	8
3 CAS N°3 .....	ERREUR ! SIGNET NON DEFINI.
4 BIBLIOGRAPHIE .....	9

## 1 CAS N°1

### 1.1 RAPPEL DU CONTEXTE

Un virus est un logiciel malveillant qui se propage en infectant d'autres ordinateurs ou fichiers. Il peut endommager les données, ralentir les performances et afficher des publicités indésirables.

Pour détecter un virus sur ProcessMonitor, vous pouvez rechercher des activités suspectes telles que des processus qui s'exécutent sans autorisation, des modifications de fichiers importants, des tentatives de connexion réseau inhabituelles, etc. Vous pouvez également utiliser un logiciel antivirus pour scanner votre ordinateur et détecter tout virus présent.


### 1.2 OBJECTIFS

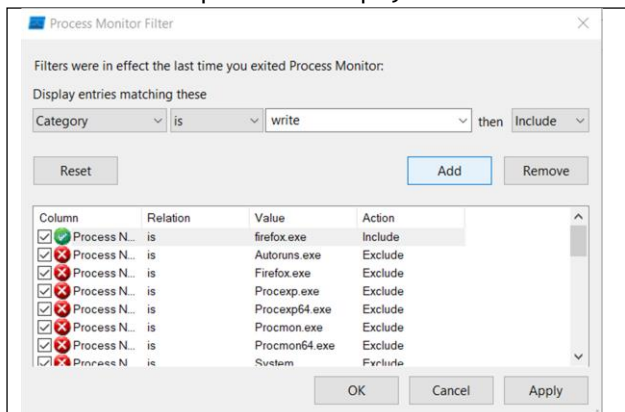
Lors de mon TP sur ProcessMonitor, mon objectif est de mieux comprendre les malwares et de les détecter moi-même en utilisant cet outil de surveillance du système. Je peux suivre les activités du système en temps réel et rechercher des anomalies qui peuvent indiquer la présence d'un malware.

Ce TP a pour but de renforcer mes compétences en matière de sécurité informatique et de m'aider à mieux protéger mon ordinateur contre les menaces en ligne.

### 1.3 HOME PAGE FIREFOX

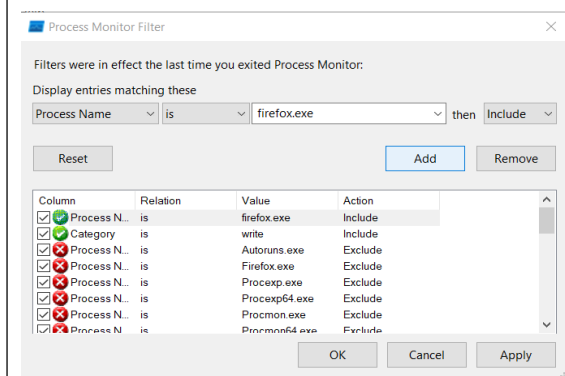
Dans ce cas n°1, nous allons utiliser ProcessMonitor pour analyser le comportement de FireFox.

Dans un premier temps, nous allons créer des filtres sur ProcessMonitor  :



Filtre "category", "write", en "include"

Mettre un filtre en "include" dans ProcessMonitor permet de n'afficher que les entrées spécifiques du journal de surveillance. Cela peut être utile pour se concentrer sur les entrées qui sont pertinentes pour l'analyse en cours et pour ignorer les entrées qui ne sont pas nécessaires ou qui peuvent perturber l'analyse.



Filtre "Process Name", "firefox.exe", en "include"

## 2 CAS N°2

---

### 2.1 OBJECTIFS

L'objectif du TP est de permettre aux apprenants de comprendre comment utiliser l'API de VirusTotal pour analyser des fichiers malveillants à l'aide d'un code en Python. En effet, le fait de pouvoir analyser des fichiers malveillants est devenu crucial pour les entreprises et les particuliers qui cherchent à protéger leurs systèmes informatiques des menaces. À travers ce TP, les apprenants pourront apprendre à créer un code en Python qui leur permettra de scanner des fichiers à l'aide de VirusTotal. Ils apprendront également à interpréter les résultats du scan afin de déterminer si le fichier est malveillant ou non. En fin de TP, les apprenants seront en mesure de créer un script en Python qui leur permettra d'automatiser l'analyse de fichiers en utilisant l'API de VirusTotal.

### 2.2 VIRUSTOTAL

VirusTotal est une plateforme en ligne qui permet de détecter les virus, les logiciels malveillants, les ransomwares, les chevaux de Troie, les adwares et d'autres menaces potentielles en utilisant plusieurs moteurs antivirus. Cette plateforme fournit des résultats de scans complets pour les fichiers, les URL et les adresses IP. Voici une présentation complète des fonctionnalités de Virus Total :

**Analyse de fichiers :** Virus Total permet d'analyser des fichiers en les téléchargeant sur la plateforme. Les fichiers sont scannés par plusieurs moteurs antivirus afin de déterminer s'ils sont infectés ou non. Les résultats sont ensuite présentés sous forme de rapport détaillé, qui indique les noms des moteurs antivirus ayant détecté une menace ainsi que d'autres informations pertinentes.

**Analyse d'URL :** Virus Total permet également d'analyser des URL pour détecter les sites web malveillants. Les résultats sont affichés sous forme de rapport détaillé, indiquant si l'URL est sûre ou non.

**Analyse d'adresses IP :** Virus Total permet d'analyser des adresses IP pour détecter les serveurs malveillants. Les résultats sont présentés sous forme de rapport détaillé, indiquant si l'adresse IP est associée à une activité malveillante ou non.

**Réputation de fichiers :** Virus Total dispose d'une base de données de fichiers malveillants connus. Lorsqu'un fichier est analysé, Virus Total compare sa signature avec celle de la base de données pour déterminer s'il est connu pour être malveillant. Les résultats sont affichés sous forme de rapport détaillé, indiquant si le fichier est considéré comme malveillant ou non.

**Analyse de comportement :** Virus Total offre des fonctionnalités d'analyse de comportement pour détecter les menaces potentielles. Cette fonctionnalité permet d'identifier les activités suspectes des fichiers, des URL ou des adresses IP. Les résultats sont affichés sous forme de rapport détaillé, qui indique les activités suspectes détectées.

**API :** VirusTotal offre une API pour permettre aux développeurs d'intégrer les fonctionnalités de la plateforme dans leurs applications. Cette API permet d'automatiser les analyses de fichiers, d'URL ou d'adresses IP.

**Collaboratif :** Virus Total permet aux utilisateurs de signaler des fichiers, des URL ou des adresses IP suspects à la communauté Virus Total. Cette fonctionnalité permet de bénéficier de la collaboration et des connaissances de la communauté pour détecter les menaces potentielles.

**Alertes :** Virus Total permet de configurer des alertes pour être averti lorsqu'un fichier, une URL ou une adresse IP est identifié comme malveillant. Les alertes peuvent être configurées pour être envoyées par courriel ou via une application de messagerie.

En résumé, Virus Total est une plateforme en ligne qui permet de détecter les menaces potentielles en utilisant plusieurs moteurs antivirus.

### 2.3 LIRE UN FICHER

Ce code Python demande à l'utilisateur de saisir un mot et l'enregistre dans deux fichiers différents (IOTest-1.txt et IOTest-2.txt). Ensuite, le code lit le contenu du fichier virus-hunter.cfg et affiche chaque ligne à l'écran.

```
unMot = input('Saisir un mot : ')
print('Votre dernier mot : ', unMot)

#IO write file
#@ToDo documenter la différence avec/sans with
#avec with
with open("F:\\BTS SIO\\Mr Valenti\\B3 - Sécurité & méthodologie\\Projets\\nieto-wylan-
v2\\build\\data\\TestsIO\\IOTest-1.txt", "w") as filout:
    filout.write(unMot)
#sans with
fichierSortie = open("F:\\BTS SIO\\Mr Valenti\\B3 - Sécurité & méthodologie\\Projets\\nieto-wylan-
v2\\build\\data\\TestsIO\\IOTest-2.txt", "a")
fichierSortie.write(unMot)
fichierSortie.close

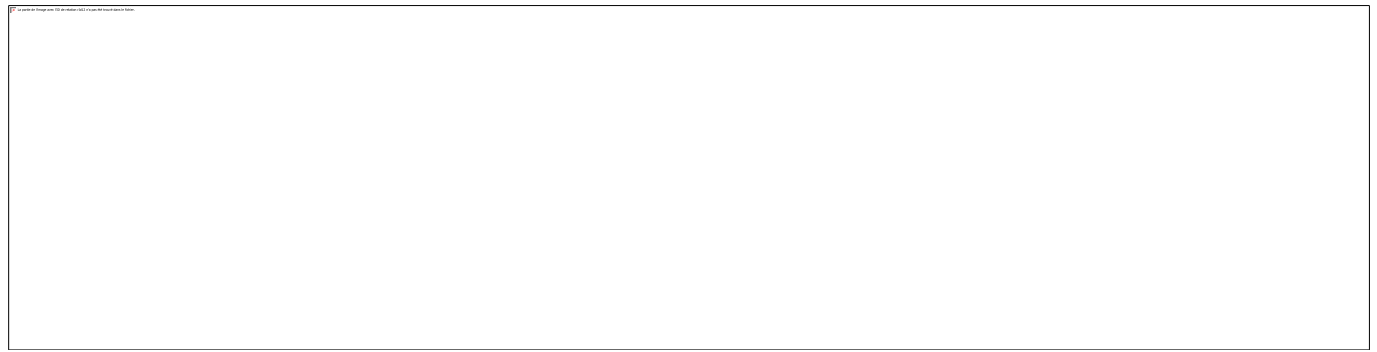
#IO read file
with open("F:\\BTS SIO\\Mr Valenti\\B3 - Sécurité & méthodologie\\Projets\\nieto-wylan-
v2\\build\\data\\TestsIO\\IOTest-2.txt", "r") as filin:
    filin.read()
#IO read file line per line
with open("F:\\BTS SIO\\Mr Valenti\\B3 - Sécurité & méthodologie\\Projets\\nieto-wylan-
v2\\build\\data\\virus-hunter.cfg", "r") as filin:
    for ligne in filin:
        print(ligne)
```

Voici une explication détaillée du code :

- La première ligne (`unMot = input('Saisir un mot : ')`) demande à l'utilisateur de saisir un mot, qui est stocké dans la variable `unMot`.
- La deuxième ligne (`print('Votre dernier mot : ', unMot)`) affiche le mot saisi par l'utilisateur à l'écran.
- Le bloc suivant de code utilise la méthode `with` pour ouvrir le fichier `IOTest-1.txt` en mode écriture ("`w`") et y écrire le mot saisi par l'utilisateur à l'aide de la méthode `write`.
- Ensuite, le code ouvre le fichier `IOTest-2.txt` en mode ajout ("`a`") à l'aide de la fonction `open`, et y écrit le mot saisi par l'utilisateur à l'aide de la méthode `write`. Puis, le code ferme ce fichier à l'aide de la méthode `close`.
- Le bloc suivant de code utilise la méthode `with` pour ouvrir le fichier `IOTest-2.txt` en mode lecture ("`r`") et lire son contenu à l'aide de la méthode `read`. Toutefois, le contenu n'est pas stocké dans une variable ou affiché à l'écran, donc cette opération est inutile.
- Enfin, le code ouvre le fichier `virus-hunter.cfg` en mode lecture ("`r`") à l'aide de la fonction `open` et de la méthode `with`, puis utilise une boucle `for` pour parcourir le contenu du fichier ligne par ligne. À chaque itération de la boucle, la variable `ligne` contient la ligne actuelle, qui est affichée à l'écran à l'aide de la fonction `print`.

### 2.4 PARCOURIR LES REPERTOIRES

Ce code utilise la bibliothèque `os` et `glob` pour effectuer des opérations sur des fichiers et répertoires.



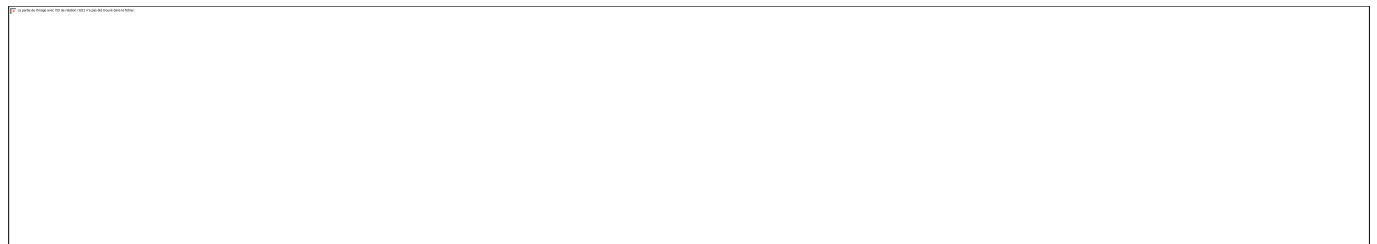
Voici une explication ligne par ligne (hors note #) :

- La première ligne importe la bibliothèque **os** et **glob** pour manipuler les fichiers et les répertoires.
- **aDirectory** contient le chemin d'accès du répertoire à parcourir.
- **listFiles** utilise la méthode **listdir** de la bibliothèque **os** pour lister tous les fichiers et répertoires présents dans le répertoire spécifié par **aDirectory**.
- La liste des fichiers est affichée à l'aide de la fonction **print**.
- Pour chaque fichier dans la liste, la taille du fichier est récupérée en utilisant la méthode **os.path.getsize** et affichée à l'aide de la fonction **print**.
- **glob.iglob** est une fonction permettant de parcourir un répertoire et ses sous-répertoires pour obtenir une liste de fichiers correspondant à un modèle de nom de fichier spécifié. Ici, le modèle est 'F:\BTS SIO\Mr Valenti\B3 - Sécurité & méthodologie\Projets\nieto-wylan-v2\build\data\TestsIO\IOTest-1.txt'.
- La liste des fichiers correspondant au modèle de nom de fichier est affichée, avec leur taille récupérée en utilisant la méthode **os.path.getsize**.

Ce code permet de lister les fichiers et dossiers présents dans un répertoire spécifique, d'afficher leur nom et leur taille. Il utilise les modules **os** et **glob**.

Par exemple, si on souhaite lister les fichiers et dossiers présents dans le répertoire "F:\BTS SIO\Mr Valenti\B3 - Sécurité & méthodologie\Projets\nieto-wylan-v2\build\data\TestsIO\" et afficher leur nom et leur taille, on peut utiliser ce code. Il suffit de remplacer le chemin d'accès par celui correspondant à notre propre répertoire.

Résultat obtenu :



On peut apercevoir les fichiers contenus dans le répertoire "F:\BTS SIO\Mr Valenti\B3 - Sécurité & méthodologie\Projets\nieto-wylan-v2\build\data\TestsIO\" et leur taille.

## 2.5 SCANNER CHAQUE FICHIER

Ce code utilise la librairie **vt** pour Python qui permet de communiquer avec l'API de VirusTotal, un service en ligne pour l'analyse de fichiers et d'URLs à la recherche de virus et de malwares.

```
import vt

#API-REST VirusTotal
client = vt.Client("be1308d1c02a6d469ef536a253116945b6c19235720ccb38b0330e8f2b894490")

# file scan
with open("F:\\BTS SIO\\Mr Valenti\\B3 - Sécurité & méthodologie\\Projets\\nieto-wylan-
v2\\build\\data\\test-1\\EICAR-positive-test-1.txt", "rb") as f:
    analysis = client.scan_file(f, wait_for_completion=True)
    print("Date : ", analysis.date)
    print('Statut : ', analysis.status)
    print('Stats : ', analysis.stats)
    print("Résultats : ", analysis.results)
```

Le code commence par importer le module **vt**. Ensuite, il crée une instance de la classe **vt.Client** en passant en argument la clé API de VirusTotal, qui permet d'accéder à l'API.

Il ouvre ensuite un fichier en lecture binaire avec **open()** et appelle la méthode **client.scan\_file()** pour analyser le fichier avec VirusTotal. Cette méthode prend en argument un objet de type **file** et un booléen **wait\_for\_completion** qui indique si l'analyse doit se faire de manière synchrone ou asynchrone. Ici, **wait\_for\_completion** est à **True**, ce qui signifie que le code attend que l'analyse soit terminée avant de continuer.

Le résultat de l'analyse est stocké dans un objet **vt.ObjectAnalysis** nommé **analysis**. Le code affiche ensuite quelques informations sur l'analyse comme la date de l'analyse, le statut, les statistiques et les résultats.

Un exemple d'utilisation de ce code serait de l'utiliser pour analyser un fichier potentiellement malveillant téléchargé depuis internet. On peut appeler ce code depuis une application qui télécharge automatiquement des fichiers et l'analyse avec VirusTotal pour s'assurer qu'ils sont sûrs avant de les ouvrir.

## 2.6 LIRE LA REPONSE DE L'API

Ce code est utilisé pour manipuler des données JSON en Python. La bibliothèque Python json est utilisée pour sérialiser et désérialiser des objets JSON.

```
import json
from io import StringIO
import ast

#JSON to String
data = json.dumps({"harmless": 0, "type-unsupported": 16, "suspicious": 0, "confirmed-timeout": 0,
"timeout": 3, "failure": 1, "malicious": 3, "undetected": 51}, sort_keys=True, indent=4)
print(data)

io = StringIO()
object = json.dump({"harmless": 0, "type-unsupported": 16, "suspicious": 0, "confirmed-timeout": 0,
"timeout": 3, "failure": 1, "malicious": 3, "undetected": 51}, io)
print(io.getvalue())

#JSON to Object
lesObjetsStr1 = '{"harmless": 0, "type-unsupported": 16, "suspicious": 0, "confirmed-timeout": 0,
"timeout": 3, "failure": 1, "malicious": 3, "undetected": 51}'
lesObjetsJSON1 = json.loads(lesObjetsStr1)
print(lesObjetsJSON1)
print(lesObjetsJSON1['malicious'])

# simple quote
lesObjetsStr2 = '{"harmless": 0, "type-unsupported": 16, "suspicious": 0, "confirmed-timeout": 0,
"timeout": 3, "failure": 1, "malicious": 3, "undetected": 51}'
data2 = ast.literal_eval(lesObjetsStr2)
print(data2)
print(data2['malicious'])
```

La première partie du code utilise la méthode dumps de la bibliothèque json pour sérialiser un objet JSON en une chaîne JSON. La chaîne JSON est ensuite affichée à l'écran. Le paramètre sort\_keys est utilisé pour trier les clés dans l'ordre alphabétique et le paramètre indent est utilisé pour afficher les données JSON avec une indentation.

La deuxième partie du code utilise la méthode dump de la bibliothèque json pour écrire un objet JSON sérialisé dans un objet StringIO. L'objet StringIO est ensuite lu à l'aide de la méthode getvalue() pour afficher la chaîne JSON.

La troisième partie du code utilise la méthode loads de la bibliothèque json pour désérialiser une chaîne JSON en un objet Python. L'objet Python est ensuite affiché à l'écran, suivi d'un accès à une valeur spécifique de l'objet à l'aide de la notation de clé.

La quatrième partie du code utilise la fonction ast.literal\_eval pour évaluer une chaîne JSON qui n'est pas une chaîne JSON valide. Cette fonction est généralement utilisée pour évaluer des chaînes qui ressemblent à du code Python en objets Python. L'objet Python est ensuite affiché à l'écran, suivi d'un accès à une valeur spécifique de l'objet à l'aide de la notation de clé.

Voici un exemple d'utilisation :

Supposons que nous avons une application qui stocke les notes de tous les étudiants d'une classe. Les notes sont stockées sous forme d'un dictionnaire Python et doivent être converties en JSON pour être stockées dans une base de données.

```
import json
notes = { "Alice": [12, 14, 18], "Bob": [9, 11, 13], "Eve": [15, 17, 16] }
notes_json = json.dumps(notes)
print(notes_json)
```

Dans cet exemple, nous avons créé un dictionnaire contenant les notes de trois étudiants, puis converti ce dictionnaire en une chaîne JSON à l'aide de la méthode json.dumps(). Nous avons ensuite imprimé la chaîne JSON. Cela nous donne :

```
{"Alice": [12, 14, 18], "Bob": [9, 11, 13], "Eve": [15, 17, 16]}
```

## 2.7 TRAITER LES ALARMES

Voici un exemple de code qui place un fichier infecté dans un dossier de quarantaine spécifié:

```
import os
import shutil

#chemin du fichier infecté détecté
file_path = r'F:\BTS SIO\Mr Valenti\B3 - Sécurité & méthodologie\Projets\nieto-wylan-
v2\build\data\test-1\EICAR-negative-test.txt'

#dossier de quarantaine
quarantine_dir = r'F:\BTS SIO\Mr Valenti\B3 - Sécurité & méthodologie\Projets\nieto-wylan-
v2\build\quarantine'

#Création du dossier de quarantaine s'il n'existe pas
if not os.path.exists(quarantine_dir):
    os.makedirs(quarantine_dir)

#Copie du fichier infecté dans le dossier de quarantaine
quarantined_file = os.path.join(quarantine_dir, os.path.basename(file_path))
shutil.copy(file_path, quarantined_file)

#Suppression du fichier infecté original
os.remove(file_path)

#Enregistrement de l'opération dans un fichier de log
with open('virus_log.txt', 'a') as f:
    f.write(f'File {file_path} has been quarantined and moved to {quarantined_file}\n')
```

Ce code déplace le fichier infecté "EICAR-negative-test.txt" dans le dossier de quarantaine spécifié "F:\BTS SIO\Mr Valenti\B3 - Sécurité & méthodologie\Projets\nieto-wylan-v2\build\quarantine". Le fichier original est supprimé, et une entrée est ajoutée dans le fichier de log "virus\_log.txt" pour enregistrer l'opération.



## 2.8 CONCLUSION

Au cours de ce TP, nous avons appris comment créer un programme en Python qui utilise l'API de VirusTotal pour scanner des fichiers à la recherche de virus. Nous avons vu comment configurer le programme à l'aide d'un fichier de configuration, comment parcourir les répertoires à scanner, comment scanner chaque fichier et comment interpréter la réponse de l'API. Nous avons également appris comment traiter les alarmes en cas de détection d'une infection et comment tracer les anomalies détectées dans un fichier de référence et placer le fichier infecté en quarantaine.

Ces compétences sont très importantes pour toute personne travaillant dans le domaine de la sécurité informatique. La capacité à détecter les menaces et à y répondre rapidement est essentielle pour protéger les systèmes informatiques contre les attaques. Python est un langage de programmation très utile pour la sécurité informatique car il est facile à apprendre et dispose de nombreuses bibliothèques utiles pour la sécurité informatique.

En conclusion, ce TP a permis d'apprendre les bases de la création d'un programme de sécurité informatique en utilisant Python et l'API de VirusTotal. Ces compétences sont utiles pour tous ceux qui souhaitent travailler dans le domaine de la sécurité informatique et peuvent être appliquées dans de nombreuses situations.

## 4 BIBLIOGRAPHIE

---

Définitions de toutes les notions de Python :  
<https://code.visualstudio.com/docs/python>

API Key :  
<https://www.virustotal.com/gui/my-apikey>