



République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumediene

**Faculté d'Electronique et d'Informatique
Département Informatique**

Projet de fin d'étude pour l'obtention du diplôme Master

Option

Sécurité des Systèmes Informatiques

Développement d'un POC d'antivirus et analyse de malwares

Sujet proposé par:

M. KHALDI Adel

Présenté par:

M. GRINI Mohammed
M. ZERROUKI Fahem

Soutenu le 24/06/2014

Devant le jury composé de:

M. GUERROUMI
Mme. BELGUERCHE
Mme. GUEBLI

Président
Membre
Membre

Binôme N° 088/2014

Remerciement

Avant tout, nous remercions notre Dieu Tout Puissant de nous avoir gardé en bonne santé et qui nous a donné la volonté et le courage ainsi que la patience pour aboutir à l'accomplissement de ce mémoire.

Nous souhaitons exprimer nos respects et notre profonde gratitude à notre encadreur M. KHALDI Adel pour la confiance qu'il nous a accordée et pour nous avoir activement guidés et encouragés. Nous ne saurons avec ces quelques mots le remercier assez.

Nos respects et nos gratitude vont également aux membres du jury qui nous ont fait l'honneur de juger ce travail ainsi qu'à tous les enseignants du département d'informatique à l'USTHB.

Nous ne terminerons pas sans témoigner notre reconnaissance à nos familles, nos proches et amis ainsi qu'à toute personne ayant contribué de près ou de loin à l'aboutissement de ce modeste travail.

Table des matières

1 Introduction aux Malwares et Antivirus	3
1.1 Introduction	3
1.2 Terminologie	3
1.2.1 La sécurité du système d'information	3
1.2.2 Vulnérabilités	3
1.2.3 Menaces	4
1.2.4 Attaques	4
1.2.4.1 Les attaques d'accès	4
1.2.4.2 Les attaques de modification	4
1.2.4.3 Les attaques par saturation	4
1.2.4.4 Les attaques de répudiation	5
1.2.5 Faux positif et faux négatif	5
1.3 Les Malwares	5
1.3.1 Définition	5
1.3.2 Catégories des malwares	5
1.3.2.1 Les Virus	5
1.3.2.2 Les vers (worms)	6
1.3.2.3 Cheval de Troie (trojan)	6
1.3.2.4 Logiciel espion (spyware)	6
1.3.2.5 Les portes dérobées (backdoor)	6
1.3.2.6 Rootkit	7
1.3.2.7 Outils de téléchargement et injecteurs (Downloaders et Droppers)	7
1.3.2.8 Enregistreurs de frappes (keyloggers)	7
1.3.2.9 Scareware (Faux logiciels de sécurité)	8
1.3.2.10 Adwares	8
1.3.2.11 Ransomware	8
1.3.2.12 Kits d'exploitation	8
1.3.3 Les cycles de vie d'un malware	9
1.3.3.1 La phase d'infection	10
1.3.3.2 La phase d'incubation	10
1.3.3.3 La phase de maladie	10
1.3.4 Les principales sources d'infection	10
1.3.5 L'évolution des malwares	11
1.3.5.1 Tableau historique des malwares	11
1.3.5.2 Evolution des malwares sur mobiles	13
1.3.5.3 2013 en chiffres	14
1.3.5.4 Les malwares sur la plateforme GNU Linux	15
1.3.5.5 Les malwares sur la plateforme Mac OS X	16
1.3.5.6 Les malwares multi-plateformes	18
1.4 Les Antivirus	18

1.4.1	Définition	18
1.4.2	Fonctionnement des Antivirus	19
1.4.2.1	Mode statique :	19
1.4.2.2	Mode dynamique :	19
1.4.3	Les techniques antivirales	19
1.4.3.1	Recherche de signatures	19
1.4.3.2	Analyse spectrale	20
1.4.3.3	Contrôle d'intégrité	20
1.4.3.4	La surveillance comportementale	20
1.4.3.5	L'émulation de code	21
1.4.4	Eradication des malwares	21
1.4.4.1	Méthode d'éradication	21
1.4.4.2	Mise à jour des antivirus	21
1.5	Le besoin de l'analyse des malwares	22
1.6	Conclusion	22
2	Etude du format PE	24
2.1	Introduction au format Portable Executable	24
2.2	Bref historique	24
2.3	Schéma du format PE	24
2.3.1	En-tête MZ-DOS	25
2.3.2	Segment DOS	26
2.3.3	En-tête PE	27
2.3.4	L'en-tête du fichier	27
2.3.5	L'en-tête facultatif	28
2.3.6	Table des Sections	30
2.3.7	La table d'importation	33
2.3.8	La table d'exportation	34
2.4	Le PE Loader	34
2.5	Les Packers	35
2.5.1	Détection des Packers avec PEiD	36
2.6	Conclusion	37
3	Conception de l'antivirus	38
3.1	Introduction	38
3.2	Conception de l'antivirus	38
3.2.1	Parseur de PE	39
3.2.1.1	Objectif du parseur	39
3.2.1.2	Vérifier la validité d'un fichier PE	39
3.2.1.3	Visualiser le contenu d'un fichier PE	40
3.2.1.4	La table d'importation	40
3.2.1.5	Détection des packers	40
3.2.2	Scanneur de malwares	40
3.2.2.1	Objectifs	40
3.2.2.2	Mécanismes de détection	41
3.2.3	Composants du moteur antivirus	41
3.2.3.1	Base de signatures	41
3.2.3.2	Type de la base de données	41
3.2.3.3	Type de signature	42
3.2.3.4	Choix du type de signature	43

3.2.3.5	Format de signatures	44
3.2.3.6	La mise à jour	44
3.2.3.7	Les types de scan	45
3.2.3.8	La présentation des résultats d'analyse	45
3.2.4	Autres fonctionnalités	46
3.3	La façon de présenter l'antivirus	46
3.4	Fonctionnement de l'antivirus :	46
3.4.1	Parseur PE	46
3.4.2	Scanneur de malwares	47
3.4.2.1	phase de scan	47
3.4.2.2	Traitement des résultats de scan	48
3.5	Gestion de la base de signatures virales	48
3.6	Les faux positifs	50
3.6.1	Description de la base de fichiers légitimes	50
3.6.2	Gestion de la base de fichiers légitimes	50
3.7	Conclusion	51
4	Implémentation	52
4.1	Introduction	52
4.2	Outils utilisés	52
4.2.1	L'environnement de développement (Visual Studio)	52
4.2.2	Visual Studio	52
4.2.3	Langages du Visual Studio	52
4.2.4	Langage de programmation C#	52
4.2.5	Jimdo	53
4.2.6	GitHub	53
4.3	Outil de gestion de la base de des signatures	53
4.3.1	Authentification	53
4.3.2	Gestion de la base de fichiers légitimes	54
4.3.2.1	Base de fichiers légitimes	54
4.3.2.2	Ajouter une signature	54
4.3.3	Gestion de la base de signatures virales	55
4.3.3.1	Base de malwares	55
4.3.3.2	Base de signatures virales	55
4.3.3.3	Ajouter une signature	56
4.3.3.4	Supprimer une signature	56
4.3.4	Génération du fichier "info.txt"	56
4.4	Antivirus	57
4.4.1	Page d'accueil	57
4.4.2	Page Menu	58
4.4.2.1	Scanneur	58
4.4.2.2	Parseur PE	63
4.4.2.3	Maintenance	63
4.4.2.4	USTHB_AV	64
4.4.2.5	Fichier suspect	65
4.4.2.6	Protection	65
4.5	L'accès à l'interface de l'antivirus	67
4.6	Conclusion	68

5 Analyse de malwares	69
5.1 Introduction	69
5.2 Objectifs de l'analyse de malwares	69
5.3 Où trouver des malwares ?	70
5.4 Construire un laboratoire sécurisé d'analyse de malwares	70
5.4.1 La virtualisation	71
5.4.1.1 Avantages de la virtualisation	71
5.4.1.2 Inconvénients de la virtualisation	71
5.4.2 Considérations du système d'exploitation	71
5.4.3 L'isolement du réseau	72
5.4.4 Les distributions dédiées à l'analyse de malwares	72
5.5 Les méthodologies d'analyse de malwares	73
5.5.1 L'analyse statique	73
5.5.1.1 Définition	73
5.5.1.2 Le désassemblage	74
5.5.1.3 Les outils et les services d'analyse statique	74
5.5.2 L'analyse dynamique	77
5.5.2.1 Définition	77
5.5.2.2 Les outils d'analyse dynamique	77
5.5.3 L'analyse des documents malveillants	80
5.5.3.1 L'analyse des documents PDF malveillants	80
5.6 Partie pratique	81
5.6.1 Présentation du malware	81
5.6.2 L'analyse statique	81
5.6.3 L'analyse dynamique	83
5.6.4 Protection contre ce type d'attaques	86
5.7 Conclusion	86

Table des figures

1.1	L'évolution des kits d'exploitation.	9
1.2	Les vulnérabilités ciblées par les kits d'exploitation.	9
1.3	nombre de malwares les dix dernières années	13
1.4	les plateformes ciblés par des malwares mobiles	14
1.5	Catégorie des malwares mobiles	15
1.6	Les malwares sous la plate-forme MAC OS X.	17
2.1	Organisation générale d'un fichier PE	25
2.2	En-tête MZ-DOS d'un fichier PE avec CFF Explorer.	26
2.3	Segment DOS d'un fichier PE avec Hex Editor.	27
2.4	L'en-tête du fichier PE avec CFF Explorer.	28
2.5	L'en-tête facultatif d'un fichier PE avec CFF Explorer.	29
2.6	La table des sections d'un fichier PE avec CFF Explorer.	31
2.7	La table des sections d'un fichier PE avec Hex Editor.	32
2.8	Table d'importation d'un fichier PE	33
2.9	Le Packer UPX	35
2.10	Fichier original et fichier packé	36
2.11	Le programme PEiD	36
3.1	Diagramme des cas d'utilisation de l'antivirus.	39
3.2	Test de validité d'un fichier PE.	40
3.3	Les étapes de la mise à jour de l'antivirus.	45
3.4	Diagramme des étapes de scan.	48
3.5	Les étapes de génération d'une signature.	49
3.6	Détail de la gestion de la base de signatures.	51
4.1	L'authentification	53
4.2	La page d'accueil de l'outil de gestion de la base de signatures.	54
4.3	La page de la base de fichiers légitimes.	54
4.4	La page d'ajouter un fichier légitime.	55
4.5	La base de malwares.	55
4.6	La base de signatures virales.	55
4.7	Ajouter une signature virale.	56
4.8	Supprimer une signature virale.	56
4.9	Génération du fichier "info.txt".	57
4.10	Page d'accueil de l'antivirus	58
4.11	Page Menu de l'antivirus	58
4.12	Les scans	59
4.13	Un virus détecté	60
4.14	L'affichage des résultats d'analyse	60
4.15	Scan un dossier sélectionné	61

4.16 Résultats d'analyse d'un dossier sélectionné	62
4.17 Le parseur PE de l'antivirus	63
4.18 La mise à jour de l'antivirus	64
4.19 Le site web de l'antivirus	64
4.20 L'envoie d'un fichier suspect	65
4.21 Les malwares détectés par l'antivirus	66
4.22 Les informations relatives à l'antivirus	66
4.23 Quarantaine	67
4.24 Menu démarrer.	67
4.25 Raccourci au Bureau.	67
4.26 Zone de notification "Systray"	68
4.27 Lancement de l'antivirus au démarrage de windows.	68
 5.1 Exemple d'un laboratoire d'analyse de malwares.	72
5.2 Remnux, distribution pour l'analyse de malwares.	73
5.3 Distribution ZeroWine pour l'analyse de malwares.	73
5.4 Utilisation de la commande file.	75
5.5 L'outil md5sum.	75
5.6 Aperçu sur l'interface du site www.virustotal.com.	75
5.7 L'outil strings de Sysinternals suite.	76
5.8 L'interface de PEiD.	76
5.9 Interface d'IDA Pro Free.	77
5.10 L'outil Process Monitor de Sysinternals suite.	78
5.11 L'outil Process Explorer de Sysinternals suite.	78
5.12 L'interface de Wireshark.	79
5.13 L'interface de Regshot.	79
5.14 L'interface de Cuckoo Sandbox.	80
5.15 Vérification de type de fichier dans l'éditeur HxD.	81
5.16 Le hash MD5 de fichier avec "WinMD5".	81
5.17 Le scan avec VirusTotal.	82
5.18 L'interface de PDFiD.	82
5.19 L'interface de paf-parser.	82
5.20 Exemple d'un fichier intégré.	83
5.21 Exemple d'un fichier crée par le document PDF.	84
5.22 L'interface Processus Monitor.	84
5.23 Processus suspect.	84
5.24 Exemple d'un fichier crée par le document PDF.	85
5.25 L'interface d'Autoruns.	85
5.26 Le trafic réseau par "fakenet"	85
5.27 Le trafic réseau par "Wireshark"	86

Liste des tableaux

1.1	Tableau historique des malwares	13
2.1	La structure de l'en-tête du fichier PE	28
2.2	Les champs intéressants de l'en-tête facultatif	30
2.3	La structure de Table des Sections	31
2.4	Sections des fichiers PE	32
3.1	Comparaison entre les techniques anti-virales	41
3.2	Comparaison entre les types de signatures	43
3.3	Format de signatures	44

Introduction générale

Devant le grand nombre de malwares identifiés chaque jour, des entreprises se sont spécialisées dans leur identification et éradication. Ce sont les éditeurs d'antivirus.

Les cybercriminels emploient désormais des techniques d'obfuscation de plus en plus complexes afin de rendre plus difficile l'identification de la nature malveillante des souches de malwares ainsi que la génération d'une signature utile à leur détection.

Des techniques d'infection novatrices sont également employées, ce qui nécessite de plus grands efforts de la part des éditeurs d'antivirus pour les désinfections. Afin d'identifier et éviter toute infection, les antivirus sont obligés d'opérer de lourdes modifications au niveau des systèmes d'exploitation.

Par conséquent, le déploiement d'un antivirus, provenant principalement d'Europe de l'Est ou des États Unis, dans un milieu professionnel (gouvernement, entreprises, associations, ...) ou personnel nécessite une confiance totale en ces états et pose, par conséquent, un sérieux problème relatif à la souveraineté nationale.

Dans le but de développer des compétences nationales dans le domaine de la lutte contre les malwares, notre projet englobera les points suivants :

- Introduction générale sur les malwares et les antivirus
- L'établissement d'un état de l'art sur les mécanismes utilisés par les antivirus pour la détection et éradication des malwares
- Étude sur le format PE (Portable Executable)
- Le développement d'une preuve de concept d'une solution antivirale
- L'implémentation de la preuve de concept de la solution antivirale
- L'étude manuelle de plusieurs souches de malwares en vue de l'établissement d'une méthodologie d'analyse.

Résumé

Notre sujet est : "conception et développement d'un POC d'antivirus et analyse de malwares"

L'objectif est d'apprendre à analyser des programmes malveillants dans le but de mieux les comprendre et concevoir des outils pour les éliminer de manière ponctuelle ou automatique.

Après la maîtrise de l'analyse manuelle de programmes malveillants, une preuve de concept d'antivirus sera conçue dans le but d'avoir un exemple fonctionnel d'un antivirus fonctionnant d'une manière classique telle qu'employée par les différents moteurs d'antiviraux sur le marché.

Abstract

Our topic is "Design and development of a POC antivirus and malware analysis"

The goal is to learn how to analyze malware in order to better understand and develop tools to remove in a timely manner or automatic.

After mastering the manual malware analysis, proof of concept virus will be designed in order to have a working example of an antivirus running in a conventional manner as used by the various antivirals engines on the market.

Chapitre 1

Introduction aux Malwares et Antivirus

1.1 Introduction

De nos jours les entreprises et les particuliers sont de plus en plus interconnectés via Internet pour des raisons diverses, par exemple : l'accès à des services bancaires, vente en ligne, échanges de données ...

Cette évolution offre évidemment beaucoup d'avantages mais elle s'accompagne également du risque inquiétant d'attaques contre les systèmes informatiques. Ces attaques peuvent se présenter sous plusieurs formes comme : le vol de données confidentielles, l'accès illégal au compte utilisateur, la diffusion de codes malicieux,...

Compte tenu des conséquences de ces types de menace, l'installation des systèmes de sécurité est devenue indispensable face aux nouvelles techniques d'attaques qui reposent sur des méthodes de plus en plus complexes et sophistiquées.

1.2 Terminologie

1.2.1 La sécurité du système d'information

Ensemble de mesures de sécurité physique, logique, administrative et de mesures d'urgence, mises en place dans une organisation, en vue d'assurer :

- la confidentialité et l'intégrité des données de son système d'information
- la protection de ses biens informatiques
- la continuité de service [1].

1.2.2 Vulnérabilités

Ce sont des vulnérabilités de sécurité dans un ou plusieurs systèmes permettant à un intrus de placer un système informatique dans un état qui accroît le risque de comportement indésirable du système et qui est contraire aux souhaits du responsable du système [2]. Ces vulnérabilités peuvent être ou non exploitable.

Exemples de vulnérabilités :

- utilisation des mots de passe non robustes
- présence de comptes non protégés par mot de passe
- vulnérabilités logicielles (débordements de tampon, injections SQL, ...)
- faiblesses cryptographiques.

1.2.3 Menaces

Une menace est un événement ou une circonstance ayant le potentiel d'endommager un système informatique en le détruisant, le divulguant, modifiant ses données, et/ou en faisant un déni de services [3].

les menaces peuvent être accidentnelles comme :

- panne disque
- chute de tension
- échange des disquettes infectée

ou bien intentionnelles comme :

- le vol
- l'écoute
- la fouille.

1.2.4 Attaques

Les attaques sont les moyens d'exploiter une vulnérabilité afin d'obtenir un accès non autorisé aux services, ressources, ou informations d'un système d'informations. C'est aussi la tentative de compromettre l'intégrité, la disponibilité, ou la confidentialité d'un système informatique [3].

1.2.4.1 Les attaques d'accès

parmis ces attaques nous trouvons :

- **Ingénierie sociale** : l'attaquant établit des relations avec le personnel pour obtenir des informations sur les mots de passe, La topologie du réseau,...
- **Portes dérobées (backdoors)** : injecter un code dans la cible pour l'exploiter plus tard
- **Sniffing** : l'attaquant se met à l'écoute sur le réseau pour obtenir des informations.

1.2.4.2 Les attaques de modification

Ces attaques visent l'intégrité des informations (modification, rejeu, ...), tel que les virus, les vers et les trojans.

1.2.4.3 Les attaques par saturation

- **Le flooding** : envoyer à une machine de nombreux paquets d'une forme spécialement conçue. La machine cible ne pourra pas traiter tous les paquets et finira par se déconnecter du réseau
- **Le smurf** : cela consiste à envoyer une trame ICMP à une adresse IP de Broadcast réseau. Le but est de coupler cette trame à une adresse IP source correspondante à celle de la cible. Et grâce à cela, le flux de réponse en destination de la cible sera multiplié
- **Le débordement de tampon** : une attaque par débordement de tampon consiste à exploiter une vulnérabilité applicative sur la machine cible afin d'y exécuter un code arbitraire qui la compromettra. Pour cela, on fait planter le programme en provoquant un dépassement de capacité du tampon par injection de données, le surplus de données est mis dans les tampons adjacents, ce qui écrasera l'adresse de retour de la fonction en cours d'exécution. On peut donc choisir les prochaines instructions qui seront exécutées par le processeur.

1.2.4.4 Les attaques de répudiation

- **IP spoofing** : L'IP spoofing est une technique qui consiste à masquer l'identité de l'attaquant en changeant l'adresse IP source. On peut facilement coupler cette technique à d'autres techniques d'attaques ne nécessitant pas de réponse, comme le ping flood, UDP flood, SYN flood, l'attaque ICMP redirect, et bien d'autres.

1.2.5 Faux positif et faux négatif

Un faux positif est une erreur de jugement d'un programme de détection, qui va réagir et renvoyer une alerte alors qu'il n'y a pas lieu de le faire. Pour un antivirus, cela se produit lorsque le programme scanne un fichier sain et le déclare infecté (positif pour son test) alors qu'il ne l'est pas.

Un faux négatif est l'absence de détection d'une vulnérabilité ou le non déclenchement d'une alerte d'intrusion.

1.3 Les Malwares

1.3.1 Définition

Les travaux de Cohen [4] en 1986 et Adleman [5] en 1988 constituent les fondements de la virologie. Un virus, au sens de Cohen, peut être formalisé par un mot sur le ruban (zone mémoire) d'une machine de Turing, qui se duplique ou se modifie sur ce même ruban lorsqu'il est activé. La notion de réplication automatique est une caractéristique primordiale du virus. Adleman propose une notion plus générale d'infection informatique. La réplication n'entre pas dans sa définition qui est alors élargie à tout programme nuisible pour la machine ou l'utilisateur.

Dans son livre, Filoli [6] propose la définition suivante pour une infection informatique ou malware :

Définition : *Programme simple ou auto-reproducteur, à caractère offensif, s'installant dans un système d'informations, à l'insu du ou des utilisateurs, en vue de porter atteinte à la confidentialité, l'intégrité, ou la disponibilité de ce système, ou susceptible d'incriminer à tort son possesseur ou l'utilisateur dans la réalisation d'un crime ou d'un délit.*

1.3.2 Catégories des malwares

Voici les principaux types de programmes malveillants :

1.3.2.1 Les Virus

Un virus est un logiciel malveillant, généralement de petite taille, qui se transmet par les réseaux ou les supports d'information amovibles, s'implante au sein des programmes en les parasitant, se duplique à l'insu des utilisateurs et produit ses effets dommageables quand le programme infecté est exécuté ou quand survient un évènement donné [7].

On distingue :

- **le virus de boot** : il est chargé en mémoire au démarrage et prend le contrôle de l'ordinateur

- **le virus d'application** : il infecte un programme exécutable et se déclenche à l'exécution de celui-ci
- **le macro virus** : il infecte les documents bureautiques en utilisant leur langage de programmation.

1.3.2.2 Les vers (worms)

Un ver informatique est un logiciel malveillant qui se reproduit sur des ordinateurs à l'aide d'un réseau informatique comme l'Internet ou tout autre support permettant sa propagation. Un ver, contrairement à un virus informatique, n'a pas besoin d'un programme hôte pour se reproduire. Il exploite les différentes ressources afin d'assurer sa reproduction. La définition d'un ver s'arrête à la manière dont il se propage de machine en machine, mais le véritable but de tels programmes peut aller au delà du simple fait de se reproduire : espionner, offrir un point d'accès caché (porte dérobée), détruire des données, faire des dégâts, envoi de multiples requêtes vers un site Internet dans le but de le saturer, etc. Les effets secondaires peuvent être aussi un ralentissement de la machine infectée, ralentissement du réseau, plantage des services ou du système, etc [8].

Des vers écrits sous forme de script peuvent être intégrés dans un courriel ou sur une page HTML sur Internet. Ils sont activés par les actions de l'utilisateur qui croit accéder à des informations lui étant destinées.

Un ver peut tout aussi bien être programmé en C, C++, Delphi, assembleur, etc. Il utilise la plupart du temps des bugs de logiciels pour se propager.

1.3.2.3 Cheval de Troie (trojan)

Un cheval de Troie est un programme installé par des pirates informatiques de manière invisible et frauduleuse. Il peut-être intégré dans la pièce-jointe d'un courriel via un lien Internet piégé, par échange de clés USB ou par le téléchargement de logiciel. L'objectif est de pouvoir exécuter des actions à l'insu de l'utilisateur (récupération, détournement, diffusion ou destruction des données), et /ou pour prendre à distance, le contrôle de l'ordinateur. Contrairement aux vers et virus, les chevaux de Troie ne se reproduisent pas.

Les chevaux de Troie servent très fréquemment à introduire une porte dérobée sur un ordinateur. L'action nuisible à l'utilisateur est alors le fait qu'un pirate informatique peut à tout moment prendre à distance (par Internet) le contrôle de l'ordinateur [9].

1.3.2.4 Logiciel espion (spyware)

Les spywares, ou logiciels espions, permettent de voler des données utilisateur : mots de passe, documents, clés d'enregistrement de logiciels, adresses électroniques, etc. Les données sont recherchées sur les supports de données ou sont filtrées à partir du trafic réseau. Les données saisies au niveau des formulaires Web (des banques en ligne, notamment) sont également collectées. Dans le pire des cas, les pirates ont alors accès à tous les comptes de messagerie, forums et boutiques en ligne utilisés par la victime. Les spywares sont généralement très utilisés par les cybercriminels.

1.3.2.5 Les portes dérobées (backdoor)

Une porte dérobée peut être introduite soit par le développeur du logiciel, soit par un tiers, typiquement un pirate informatique. La personne connaissant la porte dérobée peut l'utiliser

pour surveiller les activités du logiciel, voire en prendre le contrôle (par contournement de l’authentification) [10].

Parmi les motivations amenant les développeurs de logiciel à créer des portes dérobées :

- l’intérêt pratique d’un accès facile et toujours ouvert au logiciel pour pouvoir mener efficacement les actions de maintenance
- la possibilité de désactiver subrepticement le logiciel en cas de désaccord avec son client (non paiement de licence).

Parmi les motivations amenant les pirates informatiques à installer une porte dérobée :

- la possibilité de surveiller ce que fait l’utilisateur légitime et de copier ou détruire des données ayant une valeur (mots de passe, clé privée, coordonnées bancaires, secrets commerciaux, etc.).
- La possibilité de prendre le contrôle d’un ordinateur et de pouvoir l’utiliser pour mener des actions malfaisantes (envoi de pourriels notamment pour l’hameçonnage, de virus informatiques, déni de service).
- Le contrôle d’un vaste réseau d’ordinateurs (voir botnet), qui peut être utilisé pour du chantage au déni de service distribué (DDoS), ou revendu à des criminels.

1.3.2.6 Rootkit

Un rootkit, parfois simplement ”kit”, est un ensemble de techniques mises en œuvre par un ou plusieurs logiciels, dont le but est d’obtenir et de pérenniser un accès (généralement non autorisé) à un ordinateur de la manière la plus furtive possible, à la différence d’autres logiciels malveillants. Le terme peut désigner la technique de dissimulation ou plus généralement un ensemble particulier d’objets informatiques mettant en œuvre cette technique.

Leur furtivité est assurée par plusieurs mécanismes de dissimulation : effacement de traces, masquage de l’activité et des communications, etc. Un rootkit peut s’installer dans un autre logiciel, une bibliothèque ou dans le noyau d’un système d’exploitation. Certains peuvent modifier l’hyperviseur fonctionnant au-dessus des systèmes ou le micrologiciel intégré dans un matériel. La plupart des rootkits servent à installer des logiciels malveillants sur les machines où l’accès est obtenu. Certains fournisseurs de matériels informatiques, tel Sony, les utilisent pour s’assurer du respect des conditions d’utilisation de leurs produits par leurs clients. Certains kits ne jouent pas sur la discréetion mais sur le fait qu’enlever le kit serait une opération ardue.

Pour l’attaquant, l’utilité d’un rootkit est soit de mettre à disposition des ressources système (temps processeur, connexions réseaux, etc.) sur une, parfois en utilisant la cible comme intermédiaire pour une autre attaque; soit d’espionner, d’accéder aux données stockées ou en transit sur la machine cible [11].

1.3.2.7 Outils de téléchargement et injecteurs (Downloaders et Droppers)

Les outils de chargement et les injecteurs ont pour mission de charger ou de copier un fichier sur l’ordinateur infecté. Pour ce faire, ils essaient souvent de modifier ou de compromettre les paramètres de sécurité du système.

1.3.2.8 Enregistreurs de frappes (keyloggers)

Les Keyloggers sont un exemple de spyware. En informatique, un enregistreur de frappe est un équipement ou un logiciel qui espionne électroniquement l’utilisateur d’un ordinateur. Les enregistreurs de frappes peuvent être légitimes ou malveillants, la séparation entre les deux

étant assez floue. Dans tous les cas, il peut enregistrer les touches saisies au clavier, réaliser des captures d'écran, lister les actions de l'utilisateur et les applications actives, puis transmettre régulièrement les informations obtenues à l'individu mal intentionné [10].

1.3.2.9 Scareware (Faux logiciels de sécurité)

Les Scareware sont des faux logiciels de sécurité qui manipule et effraie les gens pour acheter une version complète du faux logiciel. Ce dernier affiche de faux rapports et alertes d'analyses, qui sont en fait simulés de tromper l'utilisateur. Le programme prend en charge l'ensemble du système de l'ordinateur pour empêcher l'enlèvement et dans la plupart des cas bloquer les autres applications, y compris les programmes antivirus légitimes de s'exécuter.

1.3.2.10 Adwares

Les adwares, ou logiciels publicitaires, enregistrent les activités et les processus d'un ordinateur (habitudes de navigation, par exemple). Lorsque l'occasion s'y prête, des messages publicitaires sont alors affichés à l'utilisateur. Des résultats affichés dans les moteurs de recherche peuvent aussi être manipulés afin de conduire à des sites commerciaux spécifiques.

1.3.2.11 Ransomware

Un ransomware, ou rançongiciel, est un logiciel malveillant qui prend en otage des données personnelles. Pour se faire, un rançongiciel chiffre des données personnelles puis demande à leur propriétaire d'envoyer de l'argent en échange de la clé qui permettra de les déchiffrer [12].

Un ransomware peut aussi bloquer l'accès de tout utilisateur à une machine jusqu'à ce qu'une clé ou un outil de débridage soit envoyé à la victime en échange d'une somme d'argent. Les modèles modernes de rançongiciels sont apparus en Russie initialement, mais on constate que le nombre d'attaques de ce type a grandement augmenté dans d'autres pays, entre autres l'Australie, l'Allemagne, les États-Unis.

Exemple d'un ransomware :

“CryptoLocker” est un logiciel malveillant, découvert en 2013 et qui tourne sous Microsoft Windows. Le programme se diffuse principalement via des mails infectés, déguisés en factures UPS, FedEx ou de banques américaines. Une fois activé, il chiffre les données personnelles de l'utilisateur via une clé RSA secrète, stockée sur des serveurs pirates, et demande une rançon (payable en bitcoins ou par des services externes comme GreenDot ou MoneyPack2) pour les déverrouiller. Le message d'alerte s'accompagne d'un compte à rebours de 72 ou 100 heures, qui menace de supprimer les données si la rançon n'est pas payée. Une fois arrivé à zéro, il augmente en réalité fortement le montant de cette dernière.

Début novembre 2013, Microsoft estime que plus de 34 000 appareils sont infectés par le programme, principalement dans des pays anglophones.

1.3.2.12 Kits d'exploitation

Les kits d'exploitation sont des paquets contenant des programmes malveillants qui servent principalement à exécuter des attaques automatisées « à la dérobée » afin de propager un programme malveillant. Ces kits sont vendus au marché noir pour des sommes allant de quelques centaines à plusieurs milliers de dollars. De nos jours, la mise en location de kits d'exploitation hébergés est chose courante et nous nous trouvons face à un marché très compétitif avec de nombreux et différents acteurs et auteurs.

Apparu il y a plusieurs années, MPack fut un des premiers exemples de cet « outil » d'un genre nouveau et beaucoup d'autres tels que ICE-Pack et Fire-Pack ont suivi. Aujourd'hui parmi les kits d'exploitation les plus célèbres, citons Eleonore, le kit d'exploitation YES et Crimepack.

Kits d'exploitation en chiffres :

La figure 1.6 montre l'évolution des différents kits d'exploitation en circulation depuis janvier 2009. (Données fournies par MalwareDomainLists).

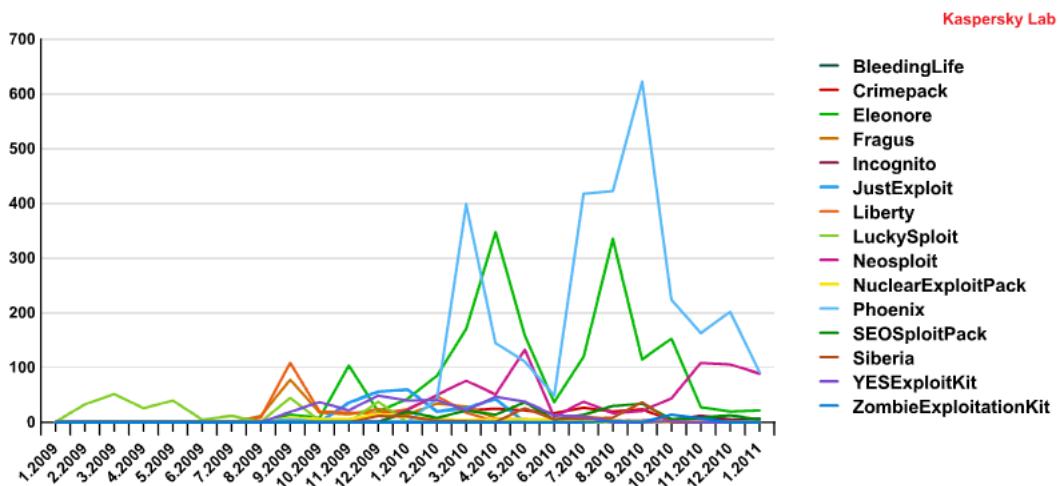


FIGURE 1.1 – L'évolution des kits d'exploitation.

La figure 1.2 montre les vulnérabilités ciblées par ces kits d'exploitation :

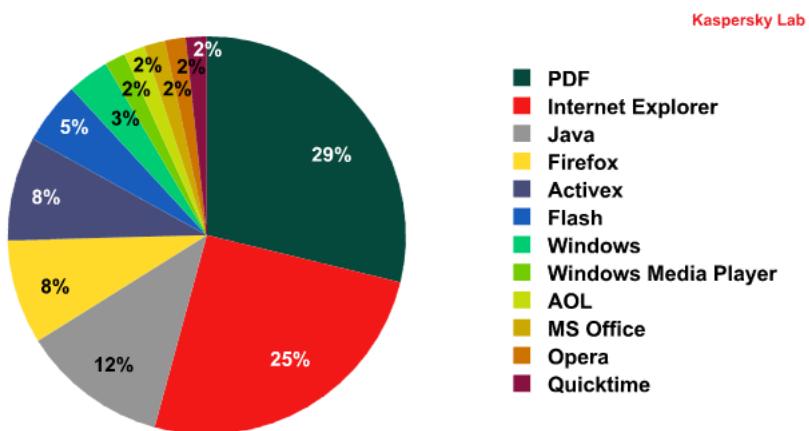


FIGURE 1.2 – Les vulnérabilités ciblées par les kits d'exploitation.

Les vulnérabilités d'Internet Explorer, PDF et Java représentent 66% des vecteurs d'attaque utilisés par les kits d'exploitation les plus répandus.

1.3.3 Les cycles de vie d'un malware

Si on ignore la phase de conception et de test, par le créateur du malware, trois phases dans la vie d'un malware peuvent être identifiées. Leurs durées de vie respectives peuvent être plus ou moins longues, selon le type de malware et l'effet recherché.

1.3.3.1 La phase d'infection

Durant cette phase, le malware va se propager dans l'environnement informatique cible. Cela peut se produire de deux manières :

- Passivement : le malware est copié sur un support (disquette, CDROM, clé USB, site de téléchargement, forum de discussion...) et transmis. Les victimes peuvent alors le copier dans leur propre environnement, avant de l'exécuter.
- Activement : l'utilisateur exécute le malware (cas de la première infection dans le système, appelée primo-infection) ou un fichier déjà contaminé lors d'une infection antérieure (primo-infection ou non).

1.3.3.2 La phase d'incubation

Cette phase constitue la plus grande partie de la vie d'un malware. La mission principale de cette phase est d'assurer la survie du malware, à travers toutes ses copies dans l'environnement cible. Il s'agit de limiter, voire d'empêcher, sa détection :

- soit par l'utilisateur : en particulier, la phase de conception veillera tout particulièrement à éviter les erreurs d'exécution qui pourraient alerter l'utilisateur
- soit par l'antivirus : dans cette optique, le malware va développer plusieurs techniques qui vont lui permettre de se dérober à la surveillance antivirale.

1.3.3.3 La phase de maladie

Lors de cette phase, la charge finale est activée. Son mode de déclenchement peut dépendre de nombreux facteurs et sera fonction de l'endroit, dans le code, où la routine offensive sera placée :

- En tête de code, la charge finale sera systématiquement exécutée, avant toute infection. Ce cas est rare, il a pour conséquence de limiter généralement la phase de survie du malware.
- En fin de code, elle n'aura lieu qu'après les processus d'infection ;
- Au milieu du code, en particulier si elle est conditionnée par la réussite ou non de l'infection.

1.3.4 Les principales sources d'infection

La plupart des sources d'infection sont les suivantes :

- **Internet** : Le réseau d'information global est la principale source de propagation de tout type de malware. En règle générale les virus et autres programmes malveillants sont situés sur des sites Web, déguisés sous forme de logiciels libres et utiles
- **Le courrier électronique** : Les mails reçus par l'utilisateur stockés dans les bases de messagerie, peuvent aussi bien contenir des virus. Le malware peut être soit en pièce jointe ou être présent dans le corps d'un message. Les données seront infectées, soit lors de l'ouverture du message ou lors de la sauvegarde sur un disque. Le courrier peut aussi être la source de deux autres menaces : le spam et le phishing. Si le spam cause principalement une perte de temps, le but du phishing est de récupérer la confidentialité des informations (par exemple numéro de carte de crédit)
- **Vulnérabilités de logiciels** : Les vulnérabilités accordent aux hackers l'accès à distance, et donc aux données personnelles, aux ressources réseau via le LAN/Ethernet, et d'autres

sources d'information

- **Support amovible** : les disques amovibles sont largement utilisés pour transférer les informations, tel que les flash disk et les disques durs. Lors de l'ouverture d'un fichier enregistré sur un disque amovible, il peut endommager les données sur le système par un virus, et le diffuser sur d'autres lecteurs.

1.3.5 L'évolution des malwares

1.3.5.1 Tableau historique des malwares

Le tableau 1.1 donne une liste des malwares les plus connus.

Date	Nom	Type	Effets
1986	Brian	Virus	Infection du système de démarrage de la disquette et corruption des données de la disquette
1987	Jerusalem	Virus	Infection et destruction de fichiers .exe et .com, résidant en mémoire
1988	Morris Worm	Ver	Premier ver écrit pour Internet, infectant les ressources machines de l'utilisateur
1991	Michelangelo	Bombe logique	Ecrasement des 256 premiers secteurs du disque dur le 6 mars 1991
1999	Melissa	Macro-virus	Via un fichier Word contaminé, la machine envoie par courrier électronique aux 50 premières adresses électroniques trouvées le fichier infecté
2000	I love you	ver	Envoi massif du ver à tous les contacts Outlook, infection et corruption de fichiers et de bases de registres
2001	Naked	Virus	Animation en Flash, présentant une femme nue, se diffusant à l'ensemble du carnet d'adresse et supprimant les répertoires Windows et systèmes
2002	BugBear	Ver avec keylogger intégré	Installation d'un logiciel espion et envoie des enregistrements de frappes de clavier sur un serveur distant

Date	Nom	Type	Effets
2003	Blaster	ver	Il fut aperçu pour la première fois dans la nature le 11 août. Sa vitesse de propagation augmenta exponentiellement jusqu'à atteindre un pic le 13 août. Le but de ce ver était de lancer une attaque en déni de service de type SYN flood. Pour se propager, le ver utilisait une vulnérabilité de type dépassement de tampon dans le service DCOM RPC, affectant le système d'exploitation dans son intégralité
2004	Sasser	ver	il profite d'une vulnérabilité LSASS de Microsoft Windows pour télécharger sur la machine infectée un fichier nommé avserve.exe dans le répertoire Windows via FTP et le port TCP 5554 et lance son exécution à distance sans aucune intervention de l'utilisateur
2005	GPCode	Ransomware	Chiffrait les fichiers et proposait à l'utilisateur de déchiffrer les fichiers en échange de 300\$
2006	Nyxem	Ver avec bombe logique	Mass mailer qui supprimait les documents Office, modifiant les paramètres systèmes, désactivant les antivirus, prévu pour s'exécuter le 3 février
2007	Storm	Trojan	Dissimulé dans une pièce jointe, infection de la machine et transformation en zombie
2008	Conficker	Ver	Ce ver exploite une vulnérabilité du Windows Server Service utilisé par Windows 2000, Windows XP, Windows Vista, Windows 7, Windows Server 2003 et Windows Server 2008
2009	Psybot	Virus	Infection des routeurs DSL pouvant être manipulés sans mot de passe en raison d'un firmware trop vieux et transformation en zombie
2010	Stuxnet	Ver	Stuxnet est un ver informatique supposé développé conjointement par les États-Unis et Israël pour s'attaquer à des systèmes iraniens. Spécifique au système Windows. La complexité du ver est très inhabituelle pour un malware. Il a été décrit par différents experts comme cyber arme, conçue pour attaquer une cible industrielle déterminée. Il s'agirait d'une première dans l'histoire.
2010	Zeus/Zitmo	Trojan	interceptait les informations bancaires, découvert en 2007 mais une nouvelle variante voit le jour en 2010 en s'attaquant aux terminaux mobiles ainsi qu'aux consoles de jeux vidéo
2011	DuQu	virus	Une vulnérabilité de Windows a été utilisée par des pirates pour mettre au point le virus DuQu, repéré le mois dernier. De façon classique, ce virus se transmet par courrier électronique dans un document Word infecté. Une fois l'ordinateur contaminé, les pirates peuvent prendre le contrôle de la machine à distance. Il aurait été créé par le même groupe à l'origine de Stuxnet

Date	Nom	Type	Effets
2012	Flame	Virus	Considéré comme le descendant plus perfectionné que Stuxnet, il vise également les systèmes industriels sensibles, intercepte les données et exploite les vulnérabilités des machines Windows

TABLE 1.1 – Tableau historique des malwares

La figure 1.3 montre que le nombre de malwares, entre 2005 et 2014, est dans une augmentation continue.

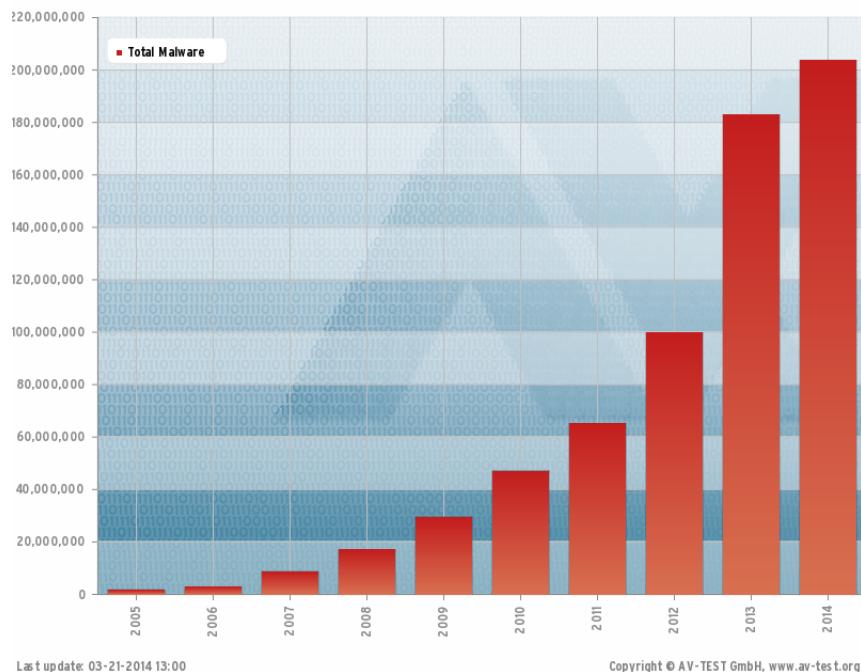


FIGURE 1.3 – nombre de malwares les dix dernières années

1.3.5.2 Evolution des malwares sur mobiles

Le nombre de malwares sur mobiles conçues pour des attaques de phishing, le vol de coordonnées de carte bancaire et d'argent sur les comptes en banque, a quasiment été multiplié par 20.

les mails sont aussi une source pour infecter les terminaux mobiles. les postes infectés forcent aussi l'utilisateur à télécharger et installer une application mobile malveillante sur son terminal. pour effectuer une fraude bancaire, la quasi totalité des banques envoient un code de sécurité par SMS à l'utilisateur. Le malware sur mobile a pour objectif d'intercepter et transférer ce code au cybercriminel, autrement il ne pourrait pas effectuer des virements bancaires à l'aide des identifiants de la victime. cette technique s'appelle "MITMO" : Malware In The MOBILE.

Les chevaux de Troie bancaires sont de loin les plus dangereux. Entre janvier et décembre 2013, leur nombre s'est multiplié. Kaspersky Lab dénombrait 67 chevaux de Troie bancaires connus début 2013, et à la fin de cette année-là, la base de données de Kaspersky Lab contenait déjà 1 321 variantes.

Les cybercriminels recourent de plus en plus à la technique d'obfuscation , qui consiste à rendre délibérément le code malveillant très complexe et donc plus difficile à analyser. Une solution antivirus mettra alors plus de temps à neutraliser le code laissant assez de temps aux cybercriminels pour arriver à leurs fins.

Parmi les méthodes pour infecter un mobile, on compte désormais la contamination via des sites légitimes compromis. La propagation du malware se fait par l'intermédiaire d'**app stores** alternatifs et de bots (ces derniers s'autopropagent généralement par l'envoi des sms, contenant un lien malveillant, aux destinataires qui figurent dans le répertoire de la victime).

1.3.5.3 2013 en chiffres

Si l'on regarde 2013 de plus près, il est possible de constater l'énorme augmentation qu'ont connue les malwares mobiles. Néanmoins, bien que les malwares aient réussi à atteindre des sommets.

- En 2013, 3 905 502 paquets d'installations ont été utilisés par les cybercriminels pour distribuer les malwares mobiles
- les terminaux Android sont la cible privilégiée voire unique des malwares pour mobiles. Selon le rapport annuel 2014 de Cisco sur la sécurité [13], ces logiciels malveillants ciblent à 98% les appareils Android. Sur cette plateforme, le malware mobile le plus répandu, à 43,8 pour cent, est ***Andr/Qdplugin*** [13]. voir figure 1.4

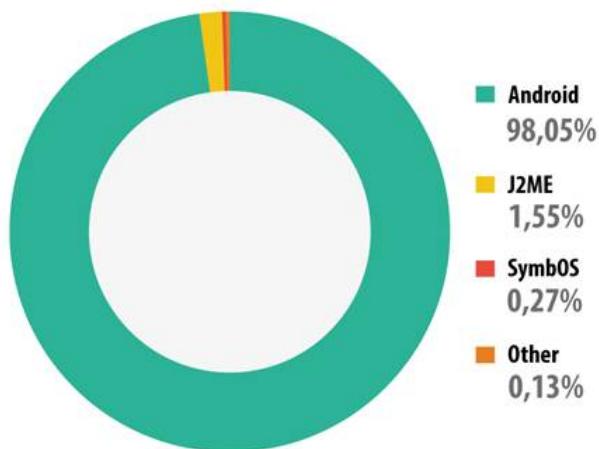


FIGURE 1.4 – les plateformes ciblés par des malwares mobiles

- La majorité des malwares mobiles(figure 1.5) est toujours spécialisée en vols d'argent mineurs grâce à des appels et à des messages vers des numéros surtaxés. Cependant, au cours de l'année, le nombre de modifications de malwares mobiles conçues pour le phishing, le vol d'informations de cartes bancaires ou d'argent a été multiplié par 19,7.

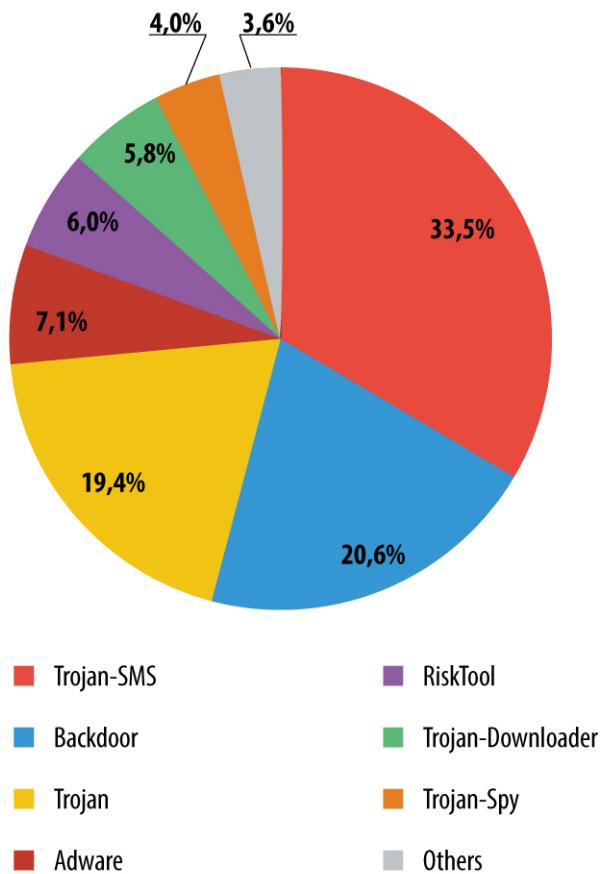


FIGURE 1.5 – Catégorie des malwares mobiles

1.3.5.4 Les malwares sur la plateforme GNU Linux

Le système d’exploitation Linux, au même titre que les systèmes d’exploitation Unix et apparentés, est généralement assez bien protégé contre les malwares. Cependant, certains programmes malveillants peuvent potentiellement endommager des systèmes GNU Linux non sécurisés.

Comme les autres systèmes Unix, Linux implémente un environnement multi-utilisateur, dans lequel les utilisateurs possèdent des droits spécifiques correspondant à leur besoin. Il existe ainsi un système de contrôle d'accès visant à interdire à un utilisateur de lire ou de modifier un fichier. Ainsi, les malwares ont typiquement moins de capacités à altérer et à infecter un système fonctionnant sous Linux que sous DOS ou encore les systèmes Microsoft Windows ayant toujours des systèmes de fichiers en FAT32 (le système de fichier NTFS a le même type de protection que les fichiers UNIX, les systèmes Microsoft Windows à base NT isolent également les comptes entre eux). C'est pourquoi, aucun des malwares écrits pour Linux, n'a pu se propager avec succès. En outre, les vulnérabilités qui sont exploitées par les malwares sont corrigées en quelques jours par les mises à jour du noyau Linux et des logiciels composant le système.

Des scanners du malwares sont disponibles pour des systèmes Linux afin de surveiller l'activité des malwares actifs sur Windows. Ils sont principalement utilisés sur des serveurs mandataires ou des serveurs de courrier électronique, qui ont pour client des systèmes Microsoft Windows.

Opération windigo :

Windigo est un malware qui prend la forme d'un cheval de Troie et qui se sert des serveurs Unix/Linux comme plateforme de diffusion de Spams. Différent pays sont touchés comme les USA, l'Allemagne, la France, L'Italie, la Grande Bretagne, les Pays-Bas, la Russie, l'Ukraine, le Mexique ou encore le Canada [14].

Plus de 25000 serveurs ont été infectés par ce malware ces 2 dernières années, et plus de 10000 le sont toujours aujourd'hui. Ces serveurs sont principalement compromis par la vulnérabilité "Linux/Ebury". Le groupe à l'origine de la vulnérabilité "Linux/Ebury" est également l'auteur des vulnérabilités "Linux/Cdorked", "Perl/Calfbot" et "Win32/Glupteba.M".

Windigo s'appuie sur trois composants :

- **Linux/Ebury** : Backdoor OpenSSH pour garder un contrôle du serveur et pour voler les identifiants
- **Linux/Cdorked** : Backdoor HTTP pour la redirection du trafic, ainsi qu'un serveur DNS modifié connu sous le nom de Linux/Onimiki
- **Perl/Calfbot** : Script PERL pour générer du spam.

Les serveurs infectés redirigent plus d'un demi-million de visiteurs vers du contenu malicieux chaque jour. Ces serveurs donnent accès à une grande capacité de bande passante, de stockage, de puissance de calcul et de mémoire. Les attaquants sont capables d'envoyer plus de 35000000 pourriels par jours. Il est intéressant de noter que l'installation est faite de manière manuelle par les auteurs du malware.

Windigo affecte aussi bien les serveurs que les PC. Les systèmes d'exploitation affectés sont Linux, FreeBSD, OpenBSD, OS X, et même Microsoft Windows (avec Perl tournant sous Cygwin). Les serveurs cPanel et kernel.org font partie de la liste des victimes de ce malware. Il agit de différentes manières en fonction des systèmes :

- Sous Microsoft Windows il installe un kit d'exploitation
- Sous MAC OS, il affiche des publicités pour des sites de rencontre
- Sous IOS les utilisateurs sont redirigés vers des sites à contenu pornographique.

En cas d'infection, un formatage et une réinstallation complète du système sont nécessaires. la commande suivante dans un terminal permet de savoir si le système est infecté ou non :

```
$ ssh -G 2>&1 | grep -e illegal -e unknown > /dev/null && echo "System_clean"
|| echo "System_infected"
```

1.3.5.5 Les malwares sur la plateforme Mac OS X

Nombreux sont ceux à croire que la plateforme Mac OS X de Apple est plus sécurisée que Microsoft Windows. Certains pensent que son mode UNIX pour la gestion des privilèges et des permissions lui confère une sécurité plus solide que Microsoft Windows, et sa gamme de matériel plus limitée implique moins de code et donc moins d'exposition aux vulnérabilités liées au code.

Pourtant, comme les Macs sont moins répandus en milieu professionnel, ils constituent une cible réduite pour les cybercriminels. Ceci explique que le problème des malwares sur Macs ne représente qu'une infime partie de ce que l'on peut voir sur les plateformes Microsoft Windows. Cependant, de nouveaux malwares continuent sans cesse d'émerger. Et même si la zone d'attaques ne présente pas autant de possibilités d'infection et de propagation, les utilisateurs de Mac sont toujours vulnérables aux escroqueries et aux arnaques utilisées pour les piéger et les

inciter à installer des logiciels suspects visant à accéder à leurs systèmes et à leurs données à distance.

Par exemple, 2010 a vu une nouvelle version du cheval de Troie OSX/Pinhead, qui se présente comme une copie de l'application iPhoto comprise avec tous les nouveaux Mac. Si l'utilisateur dupé installe le logiciel, il ouvre une porte dérobée qui donne aux pirates un plein accès au système compromis. Plus tard dans l'année, c'était au tour du cheval de Troie Boonana de viser les utilisateurs de Mac, Microsoft Windows et même GNU Linux. Il s'est propagé via des liens spammés sur Facebook et a utilisé des arnaques classiques d'ingénierie sociale pour piéger ses victimes et leur faire installer une application Java, qui télécharge et exécute une autre série d'applications malveillantes.

En 2011, l'émergence des malwares Mac a attiré beaucoup d'attention. Il est évident que le problème des malwares Windows est bien supérieur à la menace sur Mac, mais les évènements de 2011 ont rappelé aux utilisateurs Mac qu'ils devaient être prudents. Les cybercriminels utilisent désormais les mêmes techniques sur Mac que sur PC, à savoir l'infection par les faux antivirus tels que MacDefender, Mac Security, MacProtector et MacGuard, qui sont tous apparus dans le courant de l'année.

Le système d'exploitation Mac OS X 10.7 d'Apple, surnommé Lion, prétend protéger l'utilisateur contre les malwares. Cependant, il ne détecte qu'un nombre limité de téléchargements malveillants issus de 10 différents types de malwares Mac.

Bien que cette année la plate-forme Mac OS X n'ait pas subi d'attaque égalant l'étendue mondiale de Flashback en 2012, les attaques ont continué à évoluer en 2013, sous des formes diverses : chevaux de Troie, attaques contre les vulnérabilités de Java et les formats des documents Word, plug-ins agressifs, JavaScript malveillant et malwares conçus pour passer outre la protection Apple Gatekeeper grâce à une fausse identité Apple Developer.

Mac malware

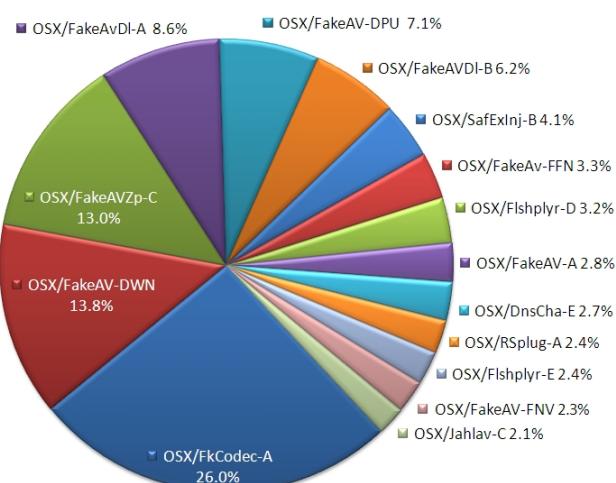


FIGURE 1.6 – Les malwares sous la plate-forme MAC OS X.

1.3.5.6 Les malwares multi-plateformes

Récemment, les attaques contre les Mac, Facebook, Apple et Microsoft ont rappelé que les cybercriminels n'avaient pas que Microsoft Windows dans leurs lignes de mire. Et le fait de cibler deux voire plusieurs plateformes à la fois, comme le font déjà les adwares et les spams, devient de plus en plus fréquent pour les malware. En utilisant des langages de programmation multi-plateformes comme Java, les cybercriminels mettent en place des logiciels malveillants qui, en compromettant un site web, peuvent percevoir le système d'exploitation qu'utilise le visiteur. Ils y déposent un malware, affichent des publicités, redirigent l'internaute vers un site web frauduleux, tentant de générer de l'argent en s'adaptant à l'appareil servant à la connexion, quel qu'il soit.

Les malwares multi-plateformes, générés par la popularité des applications multi-plateformes ciblent généralement les macro des suites bureautiques, en particulier celles qui autorisent l'exécution de scripts comme ActiveX, mais cela peut se faire par l'ajout de plugins ou d'extensions provenant de sources non fiables. Cela est vrai aussi pour les navigateurs web.

Exemples :

- Un botnet java multi-plateforme est utilisé dans des attaques DDOS. L'application Java malveillante peut être exécutée sur des machines Microsoft Windows, OS X et GNU Linux. L'éditeur de solutions de sécurité Kaspersky avertit sur son blog d'un nouveau malware utilisant une faille de la machine virtuelle Java. Même si la brèche a déjà été colmatée, nombreux sont les ordinateurs à posséder une ancienne mouture, ce qui rend la menace potentiellement sérieuse. Ce n'est pas la première fois qu'un botnet infecte les trois systèmes d'exploitation de bureau les plus populaires. Un côté multiplateforme que lui confère à chaque fois Java.
- L'exemple le plus flagrant de malware multiplateforme est Zeus et sa version mobile Zeus-in-the-Mobile (ZitMo) qui avaient pour but de voler des données bancaires. Zeus utilise un procédé assez simple. Le cheval de Troie commence par compromettre un ordinateur Microsoft Windows et attend que l'utilisateur ouvre son navigateur et accède à ses comptes en ligne. Ensuite, il ajoute des champs additionnels à la page Web de la banque, où il invite les utilisateurs à renseigner leurs numéro de téléphone ainsi que leurs modèle de téléphone mobile pour des soi-disant renouvellements de certificat. Par conséquent, les propriétaires du Zeus obtiennent toutes les informations qui ont été rentrées sur la page modifiée, y compris toutes les informations bancaires requises en ligne. Après cela, un SMS est envoyé au numéro de téléphone qu'avez renseigné : celui-ci contient un lien vers un certificat qui n'existe pas et qui est en fait le cheval de Troie, ZitMo. Une fois le smartphone infecté, le malware mobile commence à intercepter les codes d'autorisation et les notifications de mouvement sur le compte qui sont envoyés par SMS par la banque. Ainsi, sans que vous le sachiez, le cybercriminel pourra travailler à distance sur le compte bancaire de la victime en utilisant des données volées.

1.4 Les Antivirus

1.4.1 Définition

Il s'agit d'un logiciel capable de détecter et de détruire les virus contenus sur un disque. Le logiciel a pour charge de surveiller la présence de virus et éventuellement de nettoyer, supprimer ou mettre en quarantaine le ou les fichiers infectés. Ils surveillent tous les espaces dans lesquels

un virus peut se loger, c'est à dire la mémoire et les unités de stockage qui peuvent être locales ou sur le réseau [6].

1.4.2 Fonctionnement des Antivirus

les antivirus fonctionnent à quelques rares exceptions près - selon deux modes : [6]

1.4.2.1 Mode statique :

L'antivirus n'est alors actif que par une action volontaire de l'utilisateur (déclenchement manuel ou pré-programmé), il est donc le plus souvent inactif et aucune détection n'est possible. C'est le mode le plus adapté aux machines de faible puissance. La technique de surveillance de comportement n'est pas disponible dans ce mode.

Notre antivirus qu'on va déployer, il sera en mode statique.

1.4.2.2 Mode dynamique :

l'antivirus est, en fait, résident et surveille en permanence l'activité du système d'exploitation, du réseau et surtout de l'utilisateur. Il prend la main avant toute action et tente de déterminer si un risque viral existe, lié à cette action. Ce mode est gourmand en ressources et nécessite des machines relativement puissantes, pour ne pas être handicapant et pousser l'utilisateur (cas trop souvent rencontré) à désactiver ce mode au profit du précédent.

1.4.3 Les techniques antivirales

Les antivirus modernes, pour les plus efficaces, conjuguent plusieurs techniques afin de réduire le risque de fausses alertes et de non-détection, au minimum. Elles peuvent être classées en deux groupes : les techniques statiques et les techniques dynamiques [6].

1.4.3.1 Recherche de signatures

Cette technique consiste à rechercher une suite d'octets, caractéristique d'un malware donné. Cette suite est analogue à l'empreinte digitale d'une personne. Utilisée comme signature, elle doit posséder deux propriétés importantes [6] :

- Elle doit être *discriminante*. Cela signifie que la signature doit identifier spécifiquement le malware
- Elle doit être *non incriminante*. Autrement dit, elle ne doit théoriquement pas incriminer un autre malware, ou un programme sain. Elle doit donc posséder une taille et des caractéristiques suffisamment pertinentes pour ne pas provoquer de faux positifs.

En général, plus la séquence utilisée pour définir la signature est longue, plus cette signature réalisera ces deux propriétés.

Cette signature peut être :

- soit une séquence d'instructions ;
- soit un message affiché par le malware ;
- soit tout simplement la signature que le malware lui-même utilise pour éviter la surinfection d'un exécutable.

La base de signatures comporte, pour chaque malware qui s'y trouve recensé :

- La signature proprement dite ;

- l'endroit où la chercher (en-tête de Inexécutable, début ou fin du code...). plut que de rechercher la séquence d'octets qui la définit dans tout l'exécutable. l'antivirus se limite à une zone spécifique de cet exécutable, cela permet d'accélérer la recherche ;
- le mode de recherche : recherche simple de la signature, décompression du code, déchiffrement
- ...

- le nom du malware : quand l'antivirus identifie une souche virale, il informe l'utilisateur en utilisant le nom du malware comme : duqu, stuxnet, conficker, blaster, sasser, ...

Si la détection par signatures peut se révéler très efficace, elle se limite aux malwares connus et analysés. Le problème avec cette technique est qu'elle est facilement contournable. Un simple changement de compilateur suffit à leurrer la plupart des antivirus. Elle ne permet de gérer ni les virus polymorphes, ni certains virus chiffrés, et encore moins les malwares inconnus. Le taux de faux positifs est faible bien que l'identification correcte laisse quelquefois à désirer (problème de fausse incrimination).

Le principal inconvénient de ce mode de détection est la nécessité de maintenir la base de signatures virales avec les contraintes que cela comporte :taille de la base, stockage sécurisé (des sites d'antivirus contenant les bases de signatures de leurs produits sont quelquefois attaqués), la distribution sécurisée, la mise à jour plus ou moins régulière et effective par l'utilisateur, souvent négligent. A noter que la mise à jour de ces bases permet la détection de nouveaux malwares, mais aussi, dans certains cas, d'améliorer la détection des malwares précédemment repérés (par d'autres techniques) en diminuant, par exemple, les ressources machine nécessaires.

1.4.3.2 Analyse spectrale

L'analyse spectrale repose sur le postulat que tout code généré automatiquement contiendra des signes révélateurs du compilateur utilisé. De même, on part du principe qu'il est impossible de retrouver dans un vrai programme exécutable compilé certaines séquences de code. L'analyse spectrale vise donc elle aussi à repérer les virus polymorphes ou inconnus. Lorsqu'un virus polymorphe chiffre son code, la séquence en résultant contient certaines associations d'instructions que l'on ne trouverait pas dans un vrai programme. C'est ce que l'analyse spectrale tente de détecter. Par exemple, si dans un programme exécutable, l'antivirus trouve une instruction de lecture d'un octet au delà de la taille limite de la mémoire, on sera probablement en présence de code chiffré, packé ou polymorphe [6].

1.4.3.3 Contrôle d'intégrité

Puisque les virus modifient les programmes qu'ils infectent, certains antivirus utilisent un contrôleur d'intégrité pour vérifier si les fichiers de la machine ont été modifiés. Ainsi, une base de données est construite, qui contient des détails sur les fichiers exécutables du système, comme leur taille ou leur date de modification, et éventuellement un condensé cryptographique (checksum).. Dès lors, si une de ces caractéristiques change pour un exécutable, l'antivirus s'en aperçoit [15].

1.4.3.4 La surveillance comportementale

L'antivirus est résident en mémoire et tente d'identifier tout comportement suspect (la définition d'un tel comportement se faisant par rapport à une base de comportements viraux) et le bloquer si nécessaire. Les actions suivantes peuvent être identifiées comme malveillantes :

tentatives d'ouvertures en lecture/écriture de fichiers exécutables, écriture sur des secteurs systèmes (partition ou démarrage), tentative de mise en résident, etc.

Techniquement, l'antivirus agit par détournement d'interruptions (le plus souvent, les interruptions 13H et 21H) ou d'API.

Cette technique permet de détecter quelquefois des malwares inconnus (utilisant cependant des techniques connues) et de lutter avant l'infection. Toutefois, certaines techniques virales y échappent. De plus, l'antivirus doit être en mode dynamique, ce qui ralentit, quelquefois sensiblement, le système. Les faux positifs sont relativement nombreuses. Notons que l'analyse de l'antivirus permet de connaître la base de comportements et tout le jeu du programmeur du malware consistera à utiliser cette connaissance pour mieux contourner la protection [6].

1.4.3.5 L'émulation de code

Cette technique permet de disposer de la surveillance de comportement en mode statique, ce qui est assez utile car beaucoup d'utilisateurs impatients préfèrent ce mode pourtant dangereux. Lors du scan, le code étudié est chargé dans une zone mémoire confinée, puis est émulé afin de détecter un comportement potentiellement viral. L'émulation de code est particulièrement adaptée à la lutte contre les virus polymorphes. Cette technique souffre toutefois des mêmes limitations que son homologue dynamique [6].

1.4.4 Eradication des malwares

1.4.4.1 Méthode d'éradication

Une fois un malware détecté, il faut le supprimer. Mais il n'est pas toujours simple de supprimer un malware sans endommager le programme original. En effet, certains programmes malicieux détruisent une partie du programme sain lors de leur duplication. Il ne reste plus alors qu'à détruire purement et simplement le fichier infecté. Dans les autres cas, la suppression d'un malware n'est pas forcément évidente non plus. Il s'agit d'abord de découvrir très précisément où est localisé le malware dans le fichier, sachant qu'il peut être composé de plusieurs parties. Il faut ensuite supprimer ces octets infectés, et récupérer la partie du programme dont le malware avait pris la place, afin de la restaurer. Toutes ces manipulations nécessitent bien sûr une parfaite connaissance du malware et de son mode opératoire. C'est à cela que servent les fichiers de signatures des antivirus, régulièrement remis à jour. Il faut non seulement pouvoir détecter le virus, mais aussi savoir où il cache la portion de code dont il a pris la place [15].

1.4.4.2 Mise à jour des antivirus

Cela pose donc la problématique de la mise à jour rapide des antivirus, et donc de la mise à disposition rapide des signatures de détection et méthodes de désinfection.

L'efficacité des solutions antivirus dépend des bases de données de définitions du malware. Ces bases sont dynamiques par nature étant donné la constante activité des auteurs des malwares. Par exemple, les analystes viraux de Kaspersky Lab détectent et ajoutent 100 nouvelles menaces quotidiennement à la base antivirus.

Les antivirus sont devenus de plus en plus sophistiqués au fil des années afin de contrer la complexité croissante des programmes malicieux. Des mécanismes de protection proactifs, heuristiques, conçus pour détecter les nouvelles menaces avant qu'elles n'apparaissent dans la nature offrent une première ligne de défense importante.

Néanmoins, une mise à jour régulière de la protection antivirus est plus importante que jamais étant donné la rapidité à laquelle les menaces actuelles sont capables de se propager. Les éditeurs d'antivirus ont réduit l'intervalle entre les mises à jour de signatures de trimestriellement à mensuellement et finalement à quotidiennement. Malgré toutes ces bonnes intentions, rien ne garantit que l'utilisateur final effectuera des mises à jour assez régulières. Ni qu'un virus à propagation rapide n'aura pas déjà infecté les machines avant la mise à disposition de mise à jour [16].

1.5 Le besoin de l'analyse des malwares

Tout comme l'écriture de code malveillant, il y a une myriade de raisons d'analyser les virus, les vers et les trojans. La raison principale derrière cela est qu'il n'y a aucun code source disponible pour de tels programmes. Le seul moyen de comprendre ces programmes est de les analyser et déterminer leur fonctionnement interne. Une autre raison pourrait être que beaucoup de chercheurs aiment explorer les fonctionnements cachés d'un programme en l'examinant avec un désassembleur et un débogueur.

Il y a deux techniques majeures pour analyser ce code :

- ***l'analyse statique*** : L'analyse statique de malwares consiste à explorer le contenu des fichiers suspects à l'aide de divers outils, dans le but d'extraire le maximum d'informations sans exécuter le code malveillant qu'ils pourraient contenir. A ce stade il s'agit uniquement d'observer le contenu "visible"
- ***l'analyse dynamique*** : Contrairement à l'analyse statiques, l'analyse dynamique consiste à exécuter réellement le code malveillant dans un environnement adéquat afin d'observer toutes ses actions et d'en déduire son comportement global. Ce type de méthode est parfois appelée analyse comportementale.

Ces deux techniques seront discutées dans le chapitre 5.

1.6 Conclusion

Tout au long de ce chapitre, une problématique majeure de la sécurité informatique a été abordée : les malwares. Il a été constaté qu'il existait plusieurs types de malware, tel que les virus, les vers, les trojans ...

Les éditeurs d'antivirus ont donc de beaux jours devant eux, leur fond de commerce n'étant pas prêt de disparaître. Il leur faut néanmoins travailler d'arrache-pied pour trouver (ou améliorer) de nouvelles solutions pour combattre ce fléau. Car, à chaque innovation des antivirus, les virus franchissent eux aussi un cap. Avec un avantage majeur à l'heure actuelle : il faut qu'ils soient découverts avant de pouvoir être combattus...

Si, aujourd'hui, Microsoft Windows est le système d'exploitation majoritairement touché, il est à craindre qu'avec l'explosion de GNU Linux et Apple Mac OS, et son arrivée imminente dans un plus grand nombre de foyers, fasse augmenter les attaques tournées vers ces systèmes. Surtout si des utilisateurs moins expérimentés viennent à l'utiliser. Il a cependant une bonne marge d'avance, puisque le nombre de virus en activité sous Linux est dérisoire par rapport à

ceux pour l'OS de Microsoft.

Dans le chapitre suivant on va parler sur les fichiers de format PE (Portable exécutable) de Microsoft Windows avec détails parce que les antivirus doivent être capable de parser le format PE afin de pouvoir appliquer les différentes méthodes de détection (signature, émulation de code, ...). Aussi a connaissance du format PE est importante pour l'analyste afin de mieux comprendre et analyser manuellement le malware.

Chapitre 2

Etude du format PE

2.1 Introduction au format Portable Executable

Le format PE (Portable Executable), est un format du fichier natif spécifique à Microsoft Windows, qui structure les fichiers exécutable du système d'exploitation. Développé par Microsoft, et dérivé d'Unix COFF (Common Object File Format), ce format structure les binaires à extension *.exe (Logiciel), *.ocx (Object Linking and Embedding), *.dll (Dynamic Link Library) et *.cpl (Panneau de Configuration) sur toutes les plateformes Win32 et supérieur [17].

D'un point de vue sécuritaire, la connaissance de ce format est en soit une base qu'il est nécessaire de posséder lorsque l'on souhaite se lancer dans la conception et développement d'un parseur de PE.

2.2 Bref historique

Microsoft migra vers le format PE avec l'introduction de Microsoft Windows NT 3.1. Toutes les versions suivantes de Microsoft Windows, incluant Microsoft Windows 95/98/ME, supportent ce format. Auparavant les fichiers avec « exécutable ». étaient au format NE « Executable File Format ».

La création du format PE a été induite par le fait que Microsoft souhaitait créer une structure du fichier portable, de sorte qu'elle puisse s'adapter aux différents systèmes de Microsoft Windows NT, car il faut savoir que Microsoft Windows NT était au départ capable de supporter d'autres architectures que le x86 d'Intel, Power PC et Motorola 68000 en faisait partie [18]. L'idée fût donc de créer une structure commune à ces architectures.

2.3 Schéma du format PE

PE étant une structure du fichier, cela signifie qu'il est formaté d'une certaine façon. La figure 2.1 montre son formatage :



FIGURE 2.1 – Organisation générale d'un fichier PE

Tous les fichiers PE respectent ce formatage. Si on essaye, par exemple, d'ouvrir un fichier *.exe avec un éditeur hexadécimal on s'aperçoit que les deux premiers octets sont MZ, qui correspondent bien aux 2 premiers octets de l'en-tête MZ-DOS décrit ci-dessous.

2.3.1 En-tête MZ-DOS

L'en-tête MZ-DOS permet au système d'exploitation de reconnaître le fichier comme étant un exécutable valide dans le cas où celui-ci serait lancé depuis MS-DOS, afin de pouvoir exécuter son segment DOS. Cet en-tête est également soumis à une structuration, nommée IMAGE_DOS_HEADER, cette structure permet de formater l'en-tête MZ-DOS. Ci-dessous, son prototype en langage C.

```

typedef struct IMAGE_DOS_HEADER { // DOS .EXE header
WORD e_magic; // Magic number
WORD e_cblp; // Bytes on last page of file
WORD e_cp; // Pages in file
WORD e_crlc; // Relocations
WORD e_cparhdr; // Size of header in paragraphs
WORD e_minalloc; // Minimum extra paragraphs needed
WORD e_maxalloc; // Maximum extra paragraphs needed
WORD e_ss; // Initial (relative) SS value
WORD e_sp; // Initial SP value
WORD e_csum; // Checksum
WORD e_ip; // Initial IP value
WORD e_cs; // Initial (relative) CS value
WORD e_lfarlc; // File address of relocation table
WORD e_ovno; // Overlay number
WORD e_res[4]; // Reserved words
WORD e_oemid; // OEM identifier (for e_oeminfo)
WORD e_oeminfo; // OEM information; e_oemid specific
WORD e_res2[10]; // Reserved words
LONG e_lfanew; // File address of new exe header
} IMAGE_DOS_HEADER, *PIMAGE_DOS_HEADER;
  
```

La figure 2.2 montre un exemple réel avec le parseur CFF Explorer :

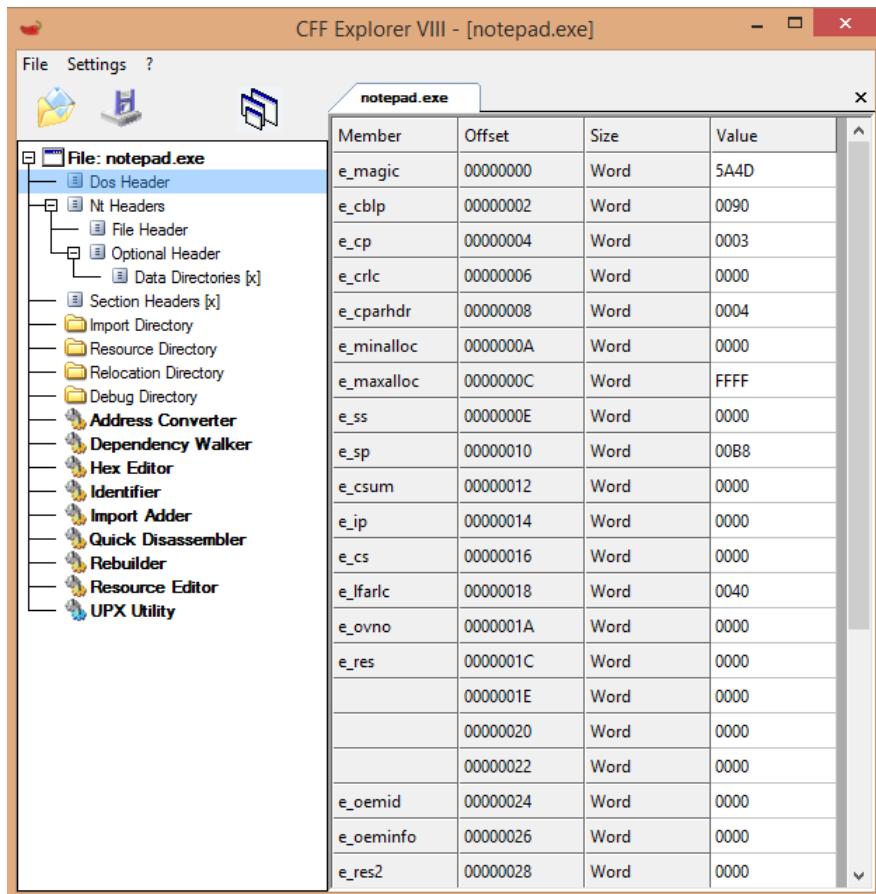


FIGURE 2.2 – En-tête MZ-DOS d'un fichier PE avec CFF Explorer.

les champs les plus importants sont :

- **e_magic** : qui doit valoir "MZ"
- **e_lfanew** : contient l'adresse du début de l'en-tête PE.

2.3.2 Segment DOS

Le segment DOS est exécuté lorsque l'application est lancée sous MS-DOS au lieu d'un environnement fenêtré Microsoft Windows, il affiche en général un message comme "This program must be run under Win32", autrement dit "Ce programme doit être exécuté sous Win32". Il s'agit d'un message implémenté par le compilateur, lors de la compilation du code logiciel, et dans le plupart des cas une exécution de int 21h, une interruption du BIOS (Basic Input Output System) qui permet d'afficher un texte à l'écran.

La figure 2.3 montre l'ouverture d'un fichier PE avec éditeur hexadécimal.

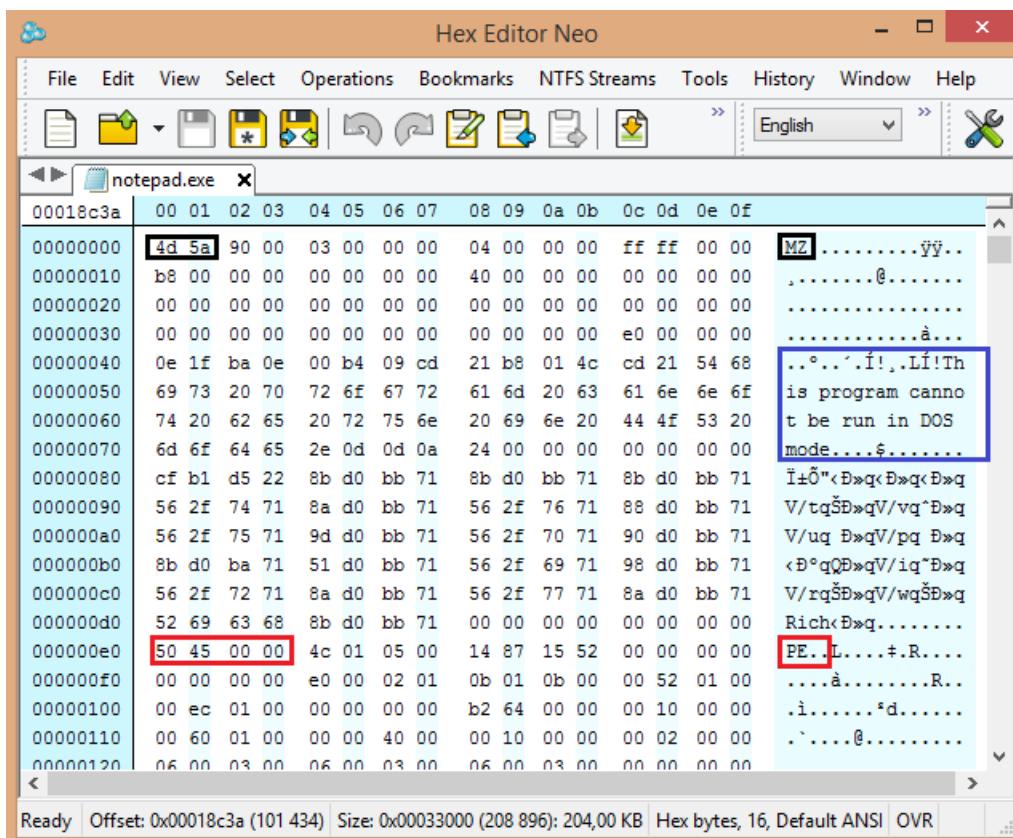


FIGURE 2.3 – Segment DOS d'un fichier PE avec Hex Editor.

2.3.3 En-tête PE

L'en-tête PE est un ensemble de structures, regroupées dans une même et unique nommé IMAGE_NT_HEADER, voici son prototype en langage C.

```
typedef struct _IMAGE_NT_HEADERS {
    DWORD           Signature;
    IMAGE_FILE_HEADER FileHeader;
    IMAGE_OPTIONAL_HEADER OptionalHeader;
} IMAGE_NT_HEADERS, *PIMAGE_NT_HEADERS;
```

Le premier champ, **Signature** : signature permettant d'identifier le fichier, qui doit être égale à 0x00004550, soit "PE00". (voire la figure 2.3 qui est en rouge).

2.3.4 L'en-tête du fichier

L'en-tête du fichier est une structure nommée IMAGE_FILE_HEADER, qui contient les informations concernant la structuration du fichier. Elle est prototypée comme suit :

```
typedef struct _IMAGE_FILE_HEADER {
    WORD   Machine;
    WORD   NumberOfSections;
    DWORD  TimeDateStamp;
    DWORD  PointerToSymbolTable;
    DWORD  NumberOfSymbols;
    WORD   SizeOfOptionalHeader;
    WORD   Characteristics;
} IMAGE_FILE_HEADER, *PIMAGE_FILE_HEADER;
```

Le Tableau 2.1 explique les champs de la structure IMAGE_FILE_HEADER : Exemple réel

champ	contenu
Machine	Le processeur pour lequel le fichier est prévu.
NumberOfSections	Le nombre de sections dans le fichier.
TimeDateStamp	La date et l'heure auxquelles le fichier a été créé.
PointerToSymbolTable	Utilisé pour le débogage. Semble être toujours 0.
NumberOfSymbols	Utilisé pour le débogage. Semble être toujours 0.
SizeOfOptionalHeader	Taille de la structure OptionalHeader.
Characteristics	Contient des flags pour le fichier.

TABLE 2.1 – La structure de l'en-tête du fichier PE

avec CFF Explorer (figure 2.4) :

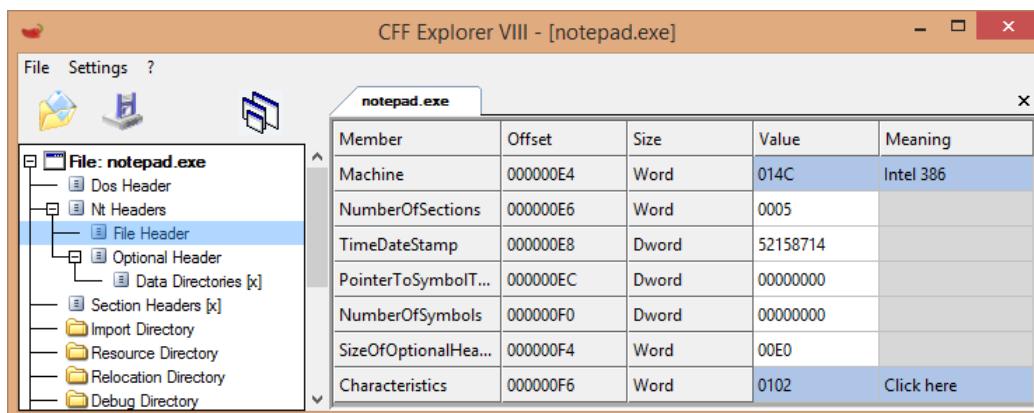


FIGURE 2.4 – L'en-tête du fichier PE avec CFF Explorer.

2.3.5 L'en-tête facultatif

L'en-tête facultatif est le dernier membre de la structure IMAGE_NT_HEADER. Il contient des informations sur l'organisation logique du fichier PE [19]. Il est prototypé comme suit :

```
typedef struct _IMAGE_OPTIONAL_HEADER {
    WORD           Magic;
    BYTE           MajorLinkerVersion;
    BYTE           MinorLinkerVersion;
    DWORD          SizeOfCode;
    DWORD          SizeOfInitializedData;
    DWORD          SizeOfUninitializedData;
    DWORD          AddressOfEntryPoint;
    DWORD          BaseOfCode;
    DWORD          BaseOfData;
    DWORD          ImageBase;
    DWORD          SectionAlignment;
    DWORD          FileAlignment;
    WORD           MajorOperatingSystemVersion;
    WORD           MinorOperatingSystemVersion;
    WORD           MajorImageVersion;
    WORD           MinorImageVersion;
    WORD           MajorSubsystemVersion;
    WORD           MinorSubsystemVersion;
```

```

DWORD          Win32VersionValue ;
DWORD          SizeOfImage ;
DWORD          SizeOfHeaders ;
DWORD          CheckSum ;
WORD           Subsystem ;
WORD           DllCharacteristics ;
DWORD          SizeOfStackReserve ;
DWORD          SizeOfStackCommit ;
DWORD          SizeOfHeapReserve ;
DWORD          SizeOfHeapCommit ;
DWORD          LoaderFlags ;
DWORD          NumberOfRvaAndSizes ;
IMAGE_DATA_DIRECTORY DataDirectory [IMAGE_NUMBEROF_DIRECTORY_ENTRIES] ;
} IMAGE_OPTIONAL_HEADER, *PIMAGE_OPTIONAL_HEADER;

```

Cette structure comporte 31 champs. Certains d'entre eux sont cruciaux et d'autres ne sont pas utiles. Nous verrons uniquement les champs qui sont réellement utiles.
Exemple réel avec CFF Explorer (voire figure 2.5) :

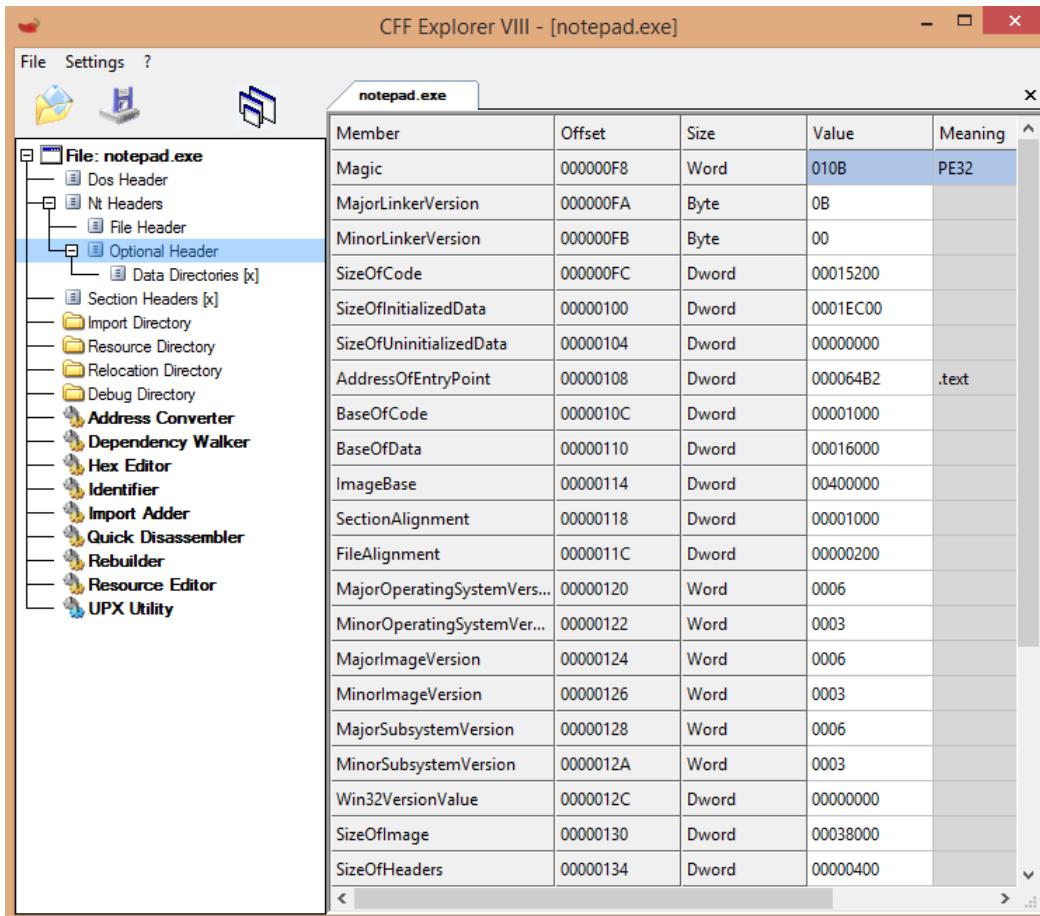


FIGURE 2.5 – L'en-tête facultatif d'un fichier PE avec CFF Explorer.

Un mot revient fréquemment en relation avec le format PE : **RVA**.

RVA signifie Relative Virtual Address. RVA est un terme un peu rebutant pour un concept aussi simple. Pour parler simplement, une RVA est une distance à partir d'un point de référence dans l'espace d'adressage virtuel. Une RVA est exactement la même chose qu'un offset fichier. Toutefois, elle est relative à une adresse virtuelle plutôt qu'au début d'un fichier.

Voici un exemple :

Si un fichier PE est chargé à l'adresse \$400000 dans l'espace d'adressage virtuel et que le pro-

gramme commence son exécution à l'adresse \$401000, on peut dire que le programme débute à la RVA \$1000. Une RVA est relative à l'adresse virtuelle du premier octet du module. Pourquoi le format PE utilise-t-il des RVA ? Le but est de réduire le temps de chargement du PE Loader. Sachant qu'un module peut être déplacé n'importe où dans l'espace d'adressage virtuel, ce serait infernal pour le PE Loader de fixer les adresses de chaque élément à position variable du module. Par opposition, si tous les éléments ayant une position variable dans le fichier utilisent des RVA, le PE Loader n'a alors pas besoin de fixer quoi que ce soit : il déplace simplement tout le module à une nouvelle adresse virtuelle. Tout cela rejoint le concept de chemin relatif et chemin absolu : une RVA est assimilable à un chemin relatif, une adresse virtuelle à un chemin absolu.

Voici le tableau 2.2 qui donne les champs intéressants de l'en-tête facultatif :

champ	contenu
AddressOfEntryPoint	Il s'agit de la RVA de la première instruction qui sera exécutée lorsque le PE Loader sera prêt à lancer le fichier.
ImageBase	C'est l'adresse de chargement souhaitable pour le fichier PE.
SectionAlignment	La granularité de l'alignement des sections en mémoire.
FileAlignment	La granularité de l'alignement des sections dans le fichier.
MajorSubSystemVersion	La version du système Win32.
MinorSubSystemVersion	La version du système Win32.
SizeOfImage	La taille totale du fichier PE en mémoire. C'est la somme de tous les en-têtes et des sections alignées avec la valeur SectionAlignment.
SizeOfHeaders	La taille de tous les en-têtes et de la table des sections. Cette valeur est égale à la taille du fichier moins la taille de toutes les sections.
Subsystem	Donne pour quel sous-système NT le fichier PE est prévu.
DataDirectory	Un tableau de structures IMAGE_DATA_DIRECTORY.

TABLE 2.2 – Les champs intéressants de l'en-tête facultatif

2.3.6 Table des Sections

La table des sections est en fait un tableau de structures suivant immédiatement l'en-tête PE. Le nombre de membres dans ce tableau est donné par le champ NumberOfSections dans l'en-tête fichier (IMAGE_FILE_HEADER). La structure en question est appelée IMAGE_SECTION_HEADER. La Table des Sections est prototypée comme suit :

```
typedef struct _IMAGE_SECTION_HEADER {
    BYTE Name[IMAGE_SIZEOF_SHORT_NAME];
    union {
        DWORD PhysicalAddress;
        DWORD VirtualSize;
    } Misc;
    DWORD VirtualAddress;
    DWORD SizeOfRawData;
    DWORD PointerToRawData;
    DWORD PointerToRelocations;
    DWORD PointerToLinenumbers;
    WORD NumberOfRelocations;
    WORD NumberOfLinenumbers;
}
```

```
DWORD Characteristics;
```

```
} IMAGE_SECTION_HEADER, *PIMAGE_SECTION_HEADER;
```

La figure 2.6 montre la table des sections d'un fichier PE avec CFF Explorer :

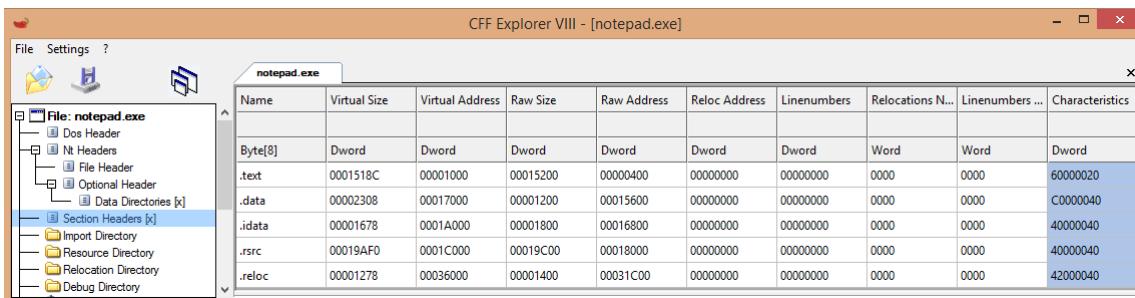


FIGURE 2.6 – La table des sections d'un fichier PE avec CFF Explorer.

le tableau 2.3 explique les champs qui sont vraiment intéressants :

champ	contenu
Name	Ce champ contient le nom de la section, il est là seulement à titre informatif pour le programmeur.
VirtualAddress	La RVA de la section. Le PE Loader vérifie et utilise la valeur de ce champ lorsqu'il place une section en mémoire.
SizeOfRawData	La taille des données de la section, arrondie au prochain multiple de FileAlignment. Le PE Loader vérifie ce champ pour savoir combien d'octets de la section il va devoir placer en mémoire.
PointerToRawData	L'offset fichier du début de la section. Le PE Loader utilise la valeur de ce champ pour trouver les données de la section dans le fichier.
Characteristics	Contient des drapeaux indiquant par exemple si la section contient du code exécutable, des données non initialisées, si elle dispose d'un accès en écriture ou en lecture.

TABLE 2.3 – La structure de Table des Sections

Chaque fichier PE contient un ensemble de sections, définies dans la Table des Sections. Une section est en fait un "segment" du fichier, possédant certaines particularités. Ci-dessous, le tableau 2.4 montre les différentes sections existantes.

Nom	Description
.text	Généralement le code (instructions) du programme.
.bss	Contient des données non-initialisées.
.reloc	Relocation.
.data	Contient des données initialisées.
.rsrc	Généralement les ressources du fichier (Curseurs, Sons, Menus ...)
.rdata	Contient l'IAT d'un fichier.
.idata	Contient l'IAT d'un fichier.
.upx	Signe d'une compression UPX, propre au logiciel UPX.
.aspack	Signe d'un package ASPACK, propre au logiciel ASPACK.
.adata	Signe d'un package ASPACK, propre au logiciel ASPACK.

TABLE 2.4 – Sections des fichiers PE

La figure 2.7 montre la table des sections d'un fichier PE avec éditeur hexadécimal :

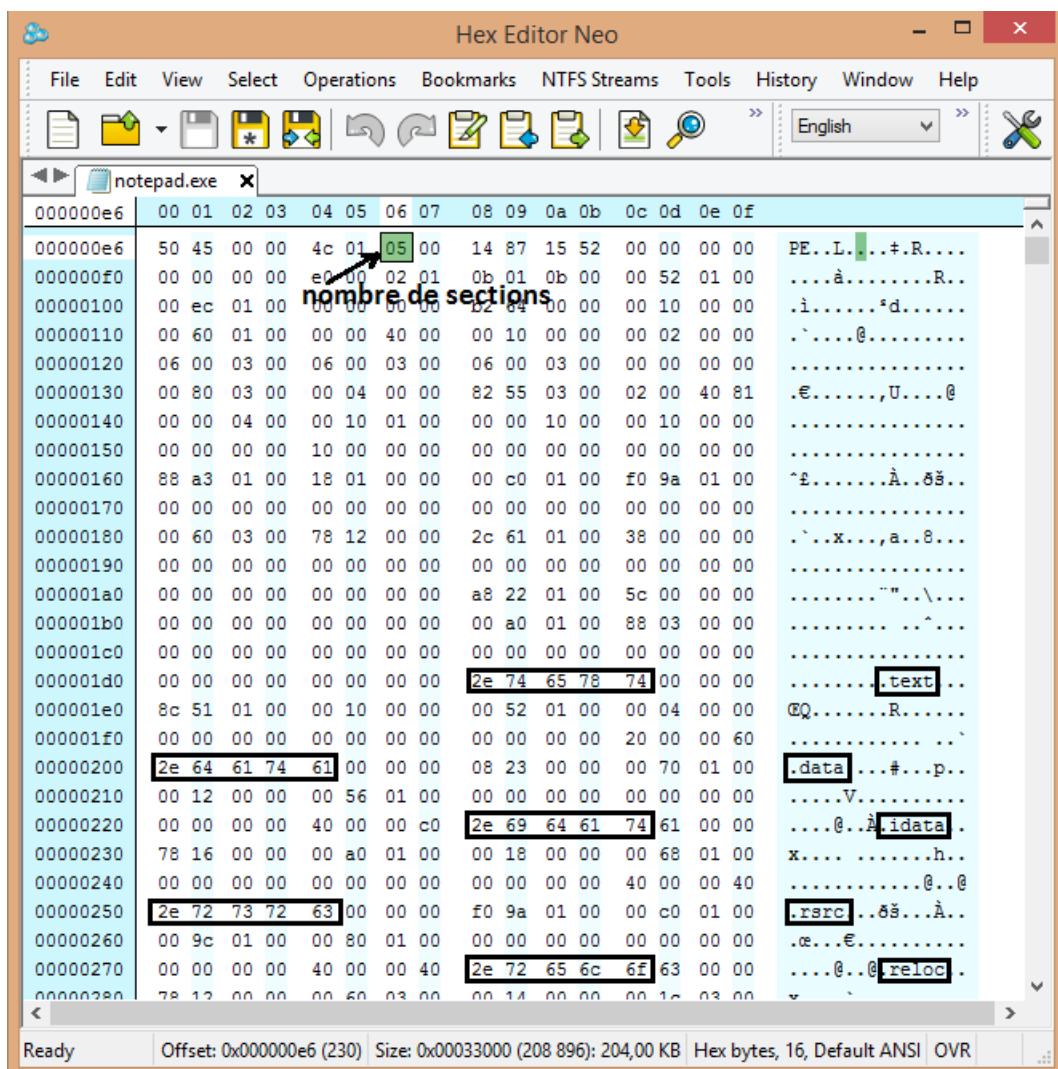


FIGURE 2.7 – La table des sections d'un fichier PE avec Hex Editor.

2.3.7 La table d'importation

L'IAT, qui signifie **Import Address Table**, est une section (.idata ou .rdata) contenant les adresses des API importées par un logiciel, ainsi que les noms des DLL exportant ces fonctions. Les API, exportées par ces DLL, permettent aux logiciels de fonctionner correctement. Son existence est dû au fait que les API sont adressées différemment en fonction des OS. La section **.idata** (ou **.rdata** parfois), qui contient cette IAT, est formaté d'une certaine façon. Il faut savoir en premier lieu qu'une structure nommée "ImageImportDescriptor" est utilisé pour chaque DLL appelée ; plus une dernière de 5 DWORD mise à zéro qui définit une terminaison. Pour chaque DLL importée, une structure nommée IMAGE_THUNK_DATA sera utilisée pour chaque API de cette DLL ; il y aura donc autant de IMAGE_THUNK_DATA que de fonctions exportées par une DLL, plus un dernier DWORD spécifiant la terminaison de cette DLL. Une troisième structure, IMAGE_IMPORT_BY_NAME, définit le nom des API ainsi que leur numéro ORDINAL (Nombre de 16 bits identifiant une fonction au sein d'une DLL). Il en existe autant qu'il y a d'API importées par DLL. Par exemple, sur l'image 2.8 d'exemple ci-dessous, il y a trois IMAGE_IMPORT_BY_NAME définis pour user32.dll, car seulement trois de ses API sont utilisées par le programme.

Module Name	Imports	OFTs	TimeStamp	ForwarderChain	Name RVA	FTs (IAT)
00016D34	N/A	00016B88	00016B8C	00016B90	00016B94	00016B98
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.DLL	62	00000000	00000000	00000000	0001A534	0001A020
ADVAPI32.dll	7	00000000	00000000	00000000	0001A544	0001A000
COMCTL32.dll	2	00000000	00000000	00000000	0001A4BC	0001A368
COMDLG32.dll	9	00000000	00000000	00000000	0001A500	0001A2F0
GDI32.dll	22	00000000	00000000	00000000	0001A528	0001A11C
user32.dll	22	mnmmnn	mnmmnn	mnmmnn	0001A510	0001A204

FIGURE 2.8 – Table d'importation d'un fichier PE

La Table d'importation est prototypée comme suit :

```
typedef struct _IMAGE_IMPORT_DESCRIPTOR {
    DWORD Characteristics;
    DWORD OriginalFirstThunk;
    DWORD TimeStamp;
    DWORD ForwarderChain;
    DWORD Name;
    DWORD FirstThunk;
} IMAGE_IMPORT_DESCRIPTOR;

typedef struct _IMAGE_IMPORT_BY_NAME {
```

```

WORD Hint; //Ordinal Number
BYTE Name[1]; //Name of function

} IMAGE_IMPORT_BY_NAME, *PIMAGE_IMPORT_BY_NAME;

typedef struct IMAGE_THUNK_DATA {
    WORD Function;
    PIMAGE_IMPORT_BY_NAME AddressOfData;
} IMAGE_THUNK_DATA, *PIMAGE_THUNK_DATA;

```

2.3.8 La table d'exportation

Quand le PE Loader exécute un programme, il charge les DLL associées dans l'espace d'adressage du processus. Il extrait ensuite les informations des fonctions importées à partir du programme principal. Il utilise ces informations pour rechercher les adresses de ces fonctions dans les DLL, afin de les renseigner dans le code du processus. L'endroit où le PE Loader cherchera ces adresses se trouve être la table d'exportation de ces mêmes DLL.

Il existe deux manières pour une DLL ou un EXE d'exporter une fonction afin de la rendre utilisable par un programme externe : elle peut être exportée par nom, ou uniquement par ordinal.

Comme pour la table d'importation, la localisation de la table d'exportation peut être retrouvée dans le data directory. Ici, la table d'exportation en est le premier membre. La structure d'exportation est appelée IMAGE_EXPORT_DIRECTORY. Elle est prototypée comme suit :

```

typedef struct IMAGE_EXPORT_DIRECTORY {
    WORD Characteristics; /* 0x00 */
    WORD TimeStamp; /* 0x04 */
    WORD MajorVersion; /* 0x08 */
    WORD MinorVersion; /* 0x0a */
    WORD Name; /* 0x0c */
    WORD Base; /* 0x10 */
    WORD NumberOfFunctions; /* 0x14 */
    WORD NumberOfNames; /* 0x18 */
    WORD AddressOfFunctions; // 0x1c RVA from base of image
    WORD AddressOfNames; // 0x20 RVA from base of image
    WORD AddressOfNameOrdinals; // 0x24 RVA from base of image
} IMAGE_EXPORT_DIRECTORY, *PIMAGE_EXPORT_DIRECTORY;

```

2.4 Le PE Loader

Le PE Loader est un élément de Microsoft Windows permettant de reconnaître et de charger en mémoire des fichiers PE. C'est grâce à lui que Microsoft Windows peut exécuter les instructions d'un tel fichier.

Voici son fonctionnement global :

- Le PE Loader examine l'en-tête MZ-DOS afin de trouver l'offset de l'en-tête PE. S'il le trouve il saute dessus
- Il vérifie la validité de l'en-tête PE. Si tel est le cas il saute à la fin de cet en-tête
- Il lit les informations concernant les sections puis mappe ces sections en mémoire en employant un procédé de "File Mapping" (copie d'un fichier en mémoire).

2.5 Les Packers

Un Packer est un utilitaire dont le but est de compresser un programme afin de réduire sa taille initiale, tout en conservant son aspect exécutable (le code original est retrouvé en mémoire lors de l'extraction).

Néanmoins, avec le temps, des nouveaux packers sont apparus et ces derniers possèdent également la capacité de chiffrer le code du programme cible, ce qui aura pour effet de réduire son poids certes, mais qui de plus, va modifier son code d'exécution. Un algorithme des plus connu, et un des plus simple à contourner (clé généralement en dur dans le programme), est sans nul doute XOR (OU-Exclusif).

Bien entendu, la partie "déchiffrement" effectué lors du lancement du programme packé/chiffré sera établie de manière transparente pour l'utilisateur. Mais un désassembler verra cela autrement, en visualisant un code (très) différent de la normale.

Les cybercriminels ont recours aux packers pour éviter que l'antivirus puisse identifier le malware rapidement. Le même malware (même identifié par l'antivirus) lorsqu'il est packé, la signature ne matchera pas et il ne sera pas détecté. Mais comme les antivirus ont du mal à analyser les malwares en mémoire (lorsqu'ils sont dépackés), alors les packers sont une bonne protection pour les cybercriminels. Aussi, ils servent à ralentir les analystes dans l'analyse de malware car ils intégrant des méthodes d'antidebug, antivm, et autres. Un packer malveillant peut identifier qu'il est exécuté dans une virtuel machine, qu'il est analysé avec un debugger, ...

Il existe de nombreux packers tel que : Armadillo, ASPack & ASProtect, NeoLite, PKLite, Petite v1.4/2.x, PolyCrypt PE, Shrinker, VBox, WWPack(32), ...

Un des Packers sans nul doute le plus connu est UPX, qui s'occupe de réduire la taille d'un exécutable de manière très efficace (près de 50%). La figure 2.9 montre le packer UPX.

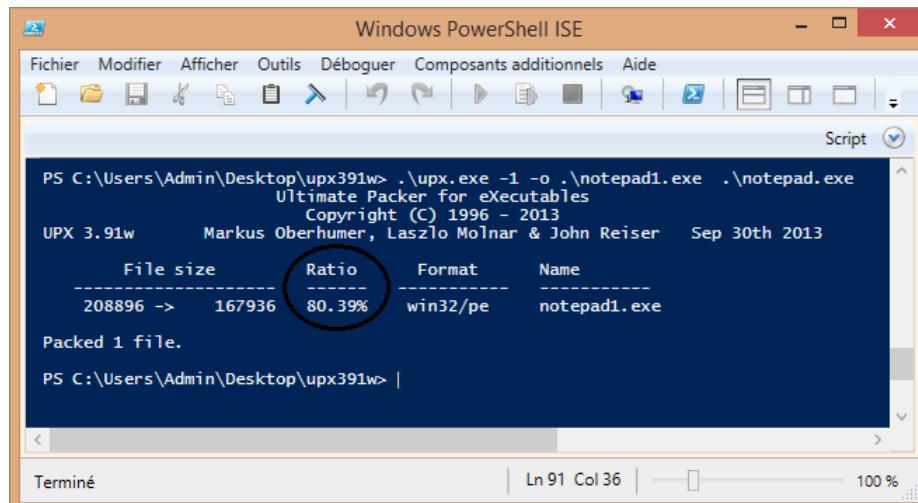


FIGURE 2.9 – Le Packer UPX

Lorsque le programme packé est exécuté, un chargeur/décompresseur (wrapper program) est exécuté aussi pour décompresser le fichier packé puis exécuter le fichier décompressé, comme le montre la figure 2.10. Quand un programme packé est analysé statiquement, seul le chargeur/décompresseur peut être disséqué [20].

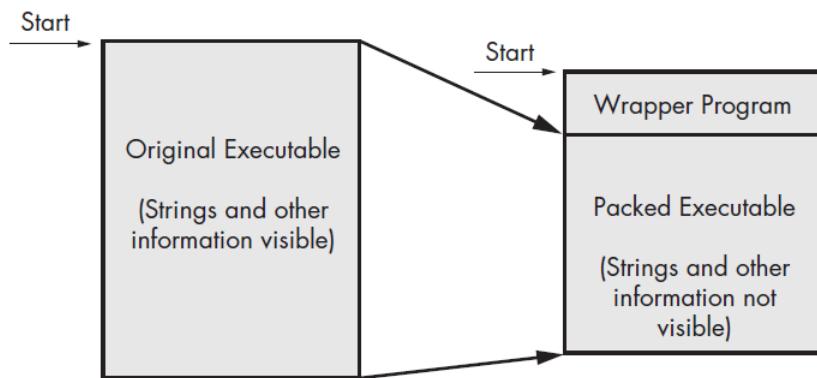


FIGURE 2.10 – Fichier original et fichier packé

2.5.1 Détection des Packers avec PEiD

Une façon de détecter des fichiers compressés (packés) est avec le programme PEiD. Il est possible d'utiliser PEiD pour détecter le type de packer ou le compilateur utilisé pour construire une application, ce qui rend l'identification d'un fichier packé, avec un packer connu, beaucoup plus facile.

PEiD contient une base de signature permettant d'identifier les packers/compilateurs utilisés, cette base s'appelle UpdateDB.txt. La figure 2.11 montre l'analyse d'un fichier .exe par PEiD.

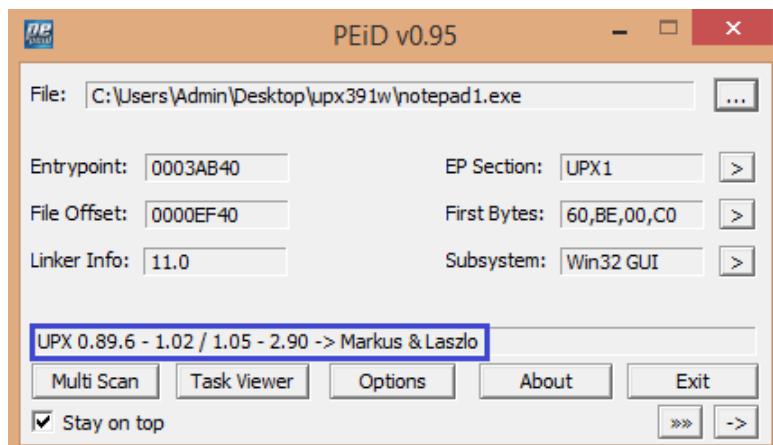


FIGURE 2.11 – Le programme PEiD

On remarque que PEiD a identifié le fichier comme étant packé avec UPX Version 0.89.6-1.02 ou de 1.05 à 2.90.

2.6 Conclusion

Le format PE (Portable Executable, exécutable portable) est le format des fichiers exécutables et bibliothèques pour les systèmes d'exploitation Microsoft Windows : .exe (programmes), .ocx (OLE et ActiveX), .dll et .cpl (élément du panneau de configuration Windows). C'est un format dérivé du COFF.

Un fichier exécutable PE est structuré d'en- MZ-DOS, segment DOS, en-tête PE et la table des sections. Le PE Loader est un élément de Microsoft Windows permettant de reconnaître et de charger en mémoire des fichiers PE.

Un Packer est un utilitaire dont le but est de compresser un programme afin de réduire sa taille initiale, tout en conservant son aspect exécutable.

Dans le chapitre suivant, on va présenter une conception de notre antivirus ainsi la conception de parseur de PE.

Chapitre 3

Conception de l'antivirus

3.1 Introduction

Un antivirus est un logiciel de protection dont le but est de détecter les malwares (comme par exemple les virus, vers, chevaux de Troie, ...). Pour cela, il inspecte la mémoire, les disques durs de l'ordinateur et les volumes amovibles (CD, DVD, clé USB, disque dur externe...) pour vérifier que les fichiers qui y sont présents ne contiennent pas de codes malveillants connus. Il permet aussi d'effectuer régulièrement des analyses planifiées.

Un antivirus protège contre les codes malveillants qu'il connaît ou qu'il reconnaît. Le temps que l'éditeur écrive la signature d'un nouveau malware et que l'antivirus la télécharge, l'internaute n'est pas protégé.

Dans ce chapitre, nous présentons les différentes phases suivies pour la conception de l'antivirus.

3.2 Conception de l'antivirus

Après avoir cité les phases de cycle de vie d'un malware (section 1.3.3, chapitre 1), nous constatons que les éditeurs des antivirus ne peuvent pas intervenir durant les deux phases "infection" et "maladie", contrairement à la phase d'incubation.

Donc l'antivirus sera composé de deux outils essentiels. Un outil qui analyse les fichiers PE (parseur de PE) et un scanneur de malwares.

Le diagramme des cas d'utilisation suivant donne une vision globale du comportement fonctionnel de l'antivirus.

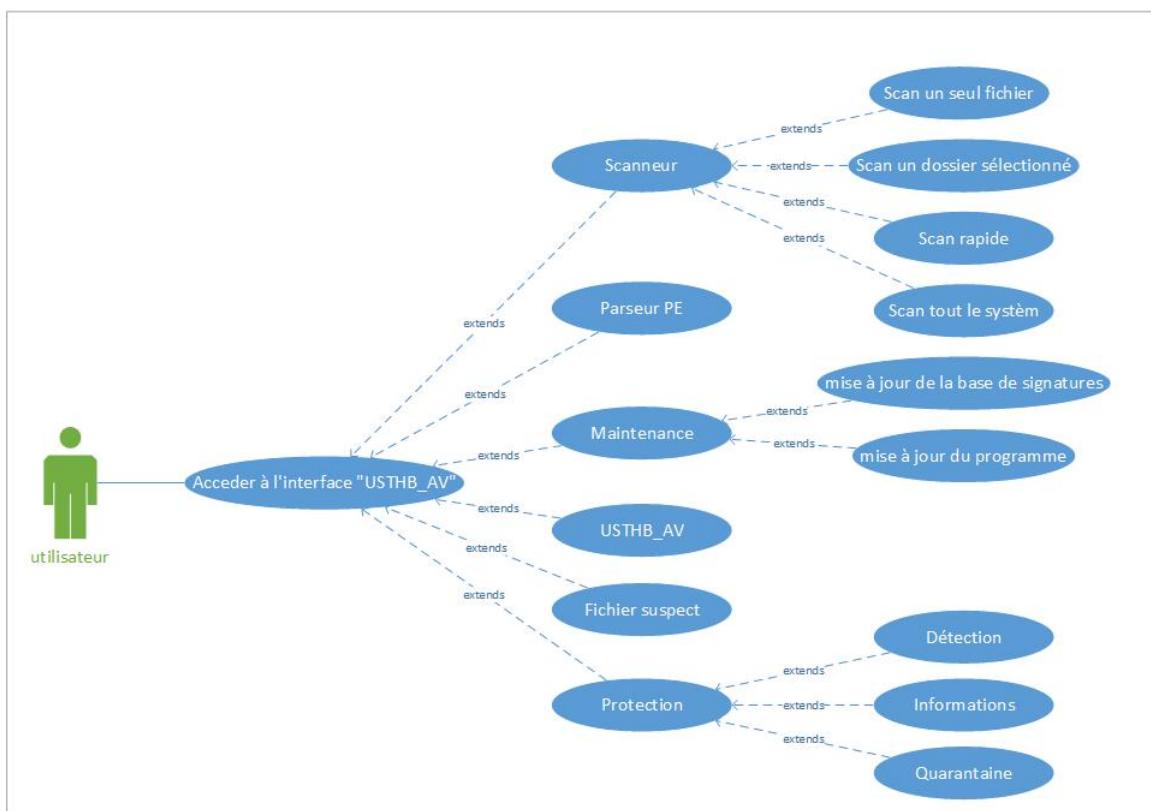


FIGURE 3.1 – Diagramme des cas d'utilisation de l'antivirus.

3.2.1 Parseur de PE

3.2.1.1 Objectif du parseur

L'objectif de ce travail est d'observer le contenu d'un fichier exécutable. Avec ce parseur nous pouvons visualiser et examiner les fichiers de format PE ainsi que leurs structures internes (voir chapitre 2).

Le parseur PE sert à analyser les fichiers PE et d'autres usages différents. Il comporte un analyseur d'entête PE, une visionneuse des fonctions API importées, un testeur de validité du format PE, vérificateur et détecteur des packers.

3.2.1.2 Vérifier la validité d'un fichier PE

Le test de validité d'un fichier de format PE, se fait selon les étapes suivantes :

- Vérifier que le fichier a un entête DOS-MZ valide en comparant le premier WORD du fichier avec la valeur IMAGE_DOS_SIGNATURE qui doit être égale à "MZ" ;
- Si le fichier a un entête DOS valide, on utilise la valeur contenue dans le champs _lfanew pour trouver l'entête PE ;
- Comparer le premier DWORD de l'entête PE avec la valeur IMAGE_NT_SIGNATURE qui doit être égale à "PE00". Si les deux valeurs concordent, alors on peut considérer le fichier comme un fichier potentiellement valide.

L'organigramme suivant montre le test de validité d'un fichier PE :

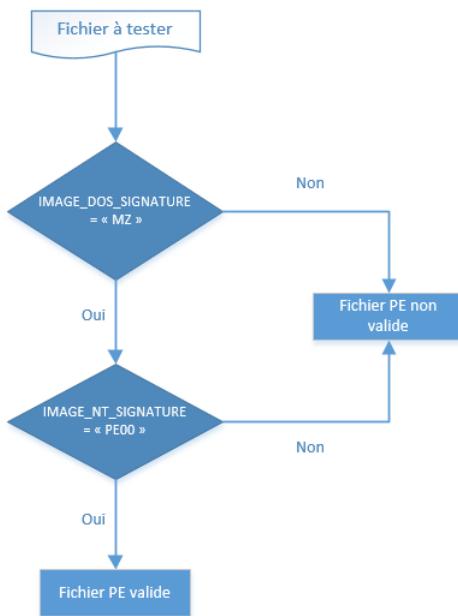


FIGURE 3.2 – Test de validité d'un fichier PE.

3.2.1.3 Visualiser le contenu d'un fichier PE

Dans cette étape nous affichons les informations internes du fichier PE, le nombre de sections, ses détails et quelques caractéristiques de ce dernier.

- Entry point : récupérer l'adresse mémoire où sera chargé le fichier PE ;
- Détail PE : afficher les caractéristiques du fichier PE et toutes les informations nécessaires pour charger ce fichier en mémoire (citées dans la section 2.3.4 et 2.3.5 , chapitre 2) ;
- Les sections PE : afficher les sections du fichier PE et leurs détails (cités dans la section 2.3.6 de chapitre 2).

3.2.1.4 La table d'importation

Afficher les DLL importées par le fichier PE après avoir chargé en mémoire, lister, pour chaque DLL, les fonctions utilisées (voir la section 2.3.7 et 2.3.8, chapitre 2).

3.2.1.5 Détection des packers

Enfin, nous examinons si le fichier PE est pacqué (voir la section 2.4 de chapitre 2) ou non, en utilisant une liste noire :

- La détection se fera au moyen d'un test de l'existence des noms de sections de fichier PE dans une liste noire des fonctions, les plus utilisées dans la compression. Si la fonction existe dans la liste, le fichier est conséquemment pacqué avec cette dernière,

3.2.2 Scanneur de malwares

3.2.2.1 Objectifs

L'objectif de l'antivirus est tacitement de détecter les malwares avant leur exécution en mémoire, rechercher sur disque dur toute indication des malware ; l'antivirus fonctionne en mode statique (voir la section 1.4.2.1, chapitre 1).

3.2.2.2 Mécanismes de détection

Pour atteindre cet objectif, nous devons étudier les techniques anti-virales utilisées dans le mode de fonctionnement statique, recherche par signature, analyse spectrale et le contrôle d'intégrité (voir la section 1.4.3, chapitre 1). Le tableau 3.1 montre une comparaison entre ces trois techniques :

Technique	Avantages	Inconvénients
Recherche par signature	détecter les malwares avant leurs exécutions en mémoire	n'est pas capable de détecter les nouveaux malwares
Analyse spectral	permet de détecter des nouveaux malwares	génère trop de faux positif
Contrôle d'intégrité	permet de détecter les modifications de fichiers sur le disque	génère trop de faux positif

TABLE 3.1 – Comparaison entre les techniques anti-virales

D'après cette comparaison nous préférons la recherche par signature, car elle répond avantageusement et favorablement à notre objectif.

3.2.3 Composants du moteur antivirus

3.2.3.1 Base de signatures

La base de signatures référence des dizaines de milliers de malwares, elle doit être mise à jour fréquemment pour reconnaître les nouveaux programmes malveillants [21].

La base de données d'un antivirus est un conteneur servant à stocker l'ensemble des signatures de malwares reconnus par l'antivirus pour être une référence de scan, cette base de signatures a plusieurs caractéristiques : type de la base de données, type et format des signatures et la façon de sa mise à jour.

3.2.3.2 Type de la base de données

Nous pouvons extraire des données de diverses bases de données, comme Microsoft Office Access, Microsoft SQL Server, les services OLAP de Microsoft SQL Server, de classeurs Excel et de fichiers de texte.

D'après le type et le format des signatures de l'antivirus, qui est un format texte, notre base de signatures sera une base de données orientée texte.

Une base de données orientée texte (ou "Base de données dans un fichier plat", de l'anglais "Flat file database") est un modèle de base de données (qui se présente généralement, sous forme d'une table) sous la forme d'un simple fichier (formats .txt, .ini,...).

Un fichier plat est un fichier texte contenant généralement un seul enregistrement par ligne, cet enregistrement est la signature de malware, chaque ligne de ce fichier texte correspond à une signature.

Le choix des fichiers plats se fait à seule fin de stocker les signatures des malwares, on n'a pas fait recours à une base de données "sql" car il n'y a pas d'avantages réels en faveur de l'utilisation de celle-ci et comprend mêmement de façon certaine quelques désavantages. Pour les actions standards (visualiser, éditer, ...), conserver l'information dans des fichiers plats est clairement plus rapide que d'y accéder au moyen d'une base de données. Sachant que l'avantage est essentiellement basé sur la simplicité, facilité de déploiement, légèreté et la non nécessité de dépendre d'un SGBD.

3.2.3.3 Type de signature

La technique anti-virale utilisée par notre antivirus pour détecter les malwares est la détection par signature "scanning", qui peut s'exprimer sous plusieurs types tels que : hash de malware et suite hexadécimale, cette dernière est récupérée depuis le code binaire du malware.

- **Hash de fichier** : une fonction de hachage (parfois appelée fonction de condensation) est une fonction permettant d'obtenir un condensé (appelé aussi condensat ou haché ou en anglais message digest) d'un texte, c'est-à-dire une suite de caractères assez courte représentant le texte qu'il condense. La fonction de hachage doit être telle qu'elle associe un seul haché à un texte en clair (cela signifie que la moindre modification du document entraîne la modification de son haché). D'autre part, il doit s'agir d'une fonction à sens unique (one-way function) afin qu'il soit impossible de retrouver le message original à partir du condensé. S'il existe un moyen de retrouver le message en clair à partir du haché [22].

Les algorithmes de hachage les plus utilisés actuellement sont :

- **Hash MD5**(Message Digest) : Développé par Rivest en 1991, MD5 crée une empreinte digitale de 128 bits à partir d'un texte de taille arbitraire en le traitant par blocs de 512 bits. Il est courant de voir des documents en téléchargement sur Internet accompagnés d'un fichier MD5, il s'agit du condensé du document permettant de vérifier l'intégrité de ce dernier [23].

Exemple pour md5 : **e1ade9d04ad99dd70ec25b83841d8b72**

- **SHA**(Secure Hash Algorithm) : pouvant être traduit par Algorithme de hachage sécurisé, SHA crée des empreintes d'une longueur de 160 bits. SHA-256 est une version améliorée de SHA, SHA-256 devient en 2002 un standard fédéral de traitement de l'information (FIPS du NIST). Elle produit un hachage de 256 bits [23].

Exemple pour sha1 : **2fd4e1c67a2d28fced849ee1bb76e7391b93eb12**

En 2004, une équipe chinoise a découvert des collisions complète "même résultat d'hachage pour deux fichier différent", md5 n'est donc plus considérée comme sûr au sens cryptographie, et il est recommander d'utiliser la fonction sha256 qui est plus fiable pour notre signature [24].

- **Suite hexadécimale :** la signature est une suite hexadécimale récupérée depuis le code binaire de malware, d'une taille variable et un offset différent, il faut que le malware soit bien identifié par cette suite.

3.2.3.4 Choix du type de signature

Quel est le type le plus fiable et rapide pour le bon fonctionnement de l'antivirus ?

Pour répondre à cette question, nous faisons une expérience par rapport aux points suivants : nombre de malwares détectés, temps de détection, faux positif et nombre de malwares polymorphes détectés, par rapport à une base du 7404 signatures, en plaçant 10 malwares dans un dossier qui contient 50 fichiers sains, et pour tester l'efficacité de détection des malwares polymorphes par les deux types de signature, nous ajoutons quatre malwares modifiés avec l'éditeur Hexadécimal HxD, en gardant toujours leur fonctionnement, le volume de données analysées est de 4.2 G, à la fin de cette expérience nous avons obtenu les résultats indiqués dans le tableau 3.3 :

Type de signature	malwares détectés	Temps de détection	Faux positifs	Malwares polymorphes
Hashage de Fichier	10	04 :23 :12 :34	0	0
Suite hexadécimal	15	00 :01 :34 :01	1	4

TABLE 3.2 – Comparaison entre les types de signatures

Analyse des résultats

- Hashage de fichier : dix malwares détectés, pas de faux positif et aucun malware polymorphe n'a été détecté dans un temps dépassant de quatre heures
- Suite hexadécimal : onze malwares détectés, un faux positif et quatre malwares polymorphes détectés dans un temps très court.

D'après les résultats obtenus, nous préférons utiliser la suite hexadécimale pour notre signature car elle nous permet de détecter tous les malwares "10 malware", ainsi que les malwares polymorphes, mais avec un seul Faux positif, dans un temps inférieur à ce de type hachage fichier qui ne nous permet pas à détecter les malwares polymorphes.

3.2.3.5 Format de signatures

Le format de signature est le suivant :

Offset	Suite hexadécimale	Nom de malware	Type de malware
--------	--------------------	----------------	-----------------

TABLE 3.3 – Format de signatures

- **Offset** : l'offset d'où la suite hexadécimale a été récupéré
- **Suite hexadécimale** : une suite d'octets récupérés depuis le code du malware
- **Nom de malware** : nom de malware attribué par les éditeurs de l'antivirus
- **Type de malware** : virus, worm, cheval de troie, ...

Ex : 58ab—38896E0E41B0CB35F3D7B94D6D973F143A75CD97F974A514869E1 : Conficker/-Worm

3.2.3.6 La mise à jour

L'efficacité des solutions antivirus dépend des bases de signatures de définitions du malware. Ces bases sont dynamiques intrinsèquement, et ce, étant donné l'activité des auteurs de malware. Par exemple, les analystes viraux de Kaspersky Lab détectent et ajoutent cent nouvelles menaces, quotidiennement, à la base de l'antivirus [25].

Néanmoins, une mise à jour régulière de la protection antivirus est plus importante que jamais, étant donné la rapidité avec laquelle les menaces actuelles sont capables de se propager. Les éditeurs d'antivirus ont réduit l'intervalle entre les mises à jour de signatures de façon trimestrielle à une façon mensuelle et finalement à une façon quotidienne.

Désormais, Kaspersky Lab fournit les mises à jour de définitions virus toutes les heures. Et pour cela, nous proposons pour notre antivirus deux méthodes pour le mettre à jour, l'une est **automatique** et l'autre **manuelle** déclenchée de la part de l'utilisateur.

Dans les deux cas, la mise à jour sera effectuée selon les étapes suivantes :

- Télécharger un fichier nommé "info.txt" depuis "Github"
- Récupérer un hash de la base de signatures de site web depuis ce fichier téléchargé
- Comparaison entre le hash récupéré de fichier "info.txt" et le hash de la base de signatures actuel, si les deux hash sont identiques alors l'antivirus informe l'utilisateur que sa base virale est à jour, sinon nous effectuons les tâches suivantes :
 - Télécharger un fichier texte depuis le site de l'antivirus nommé "new_bdd_mal.txt"
 - Remplacer l'ancienne base de signatures par la nouvelle
 - Informer l'utilisateur que la mise à jour est bien terminée.

Le diagramme suivant montre les étapes pour faire la mise à jour de notre antivirus :

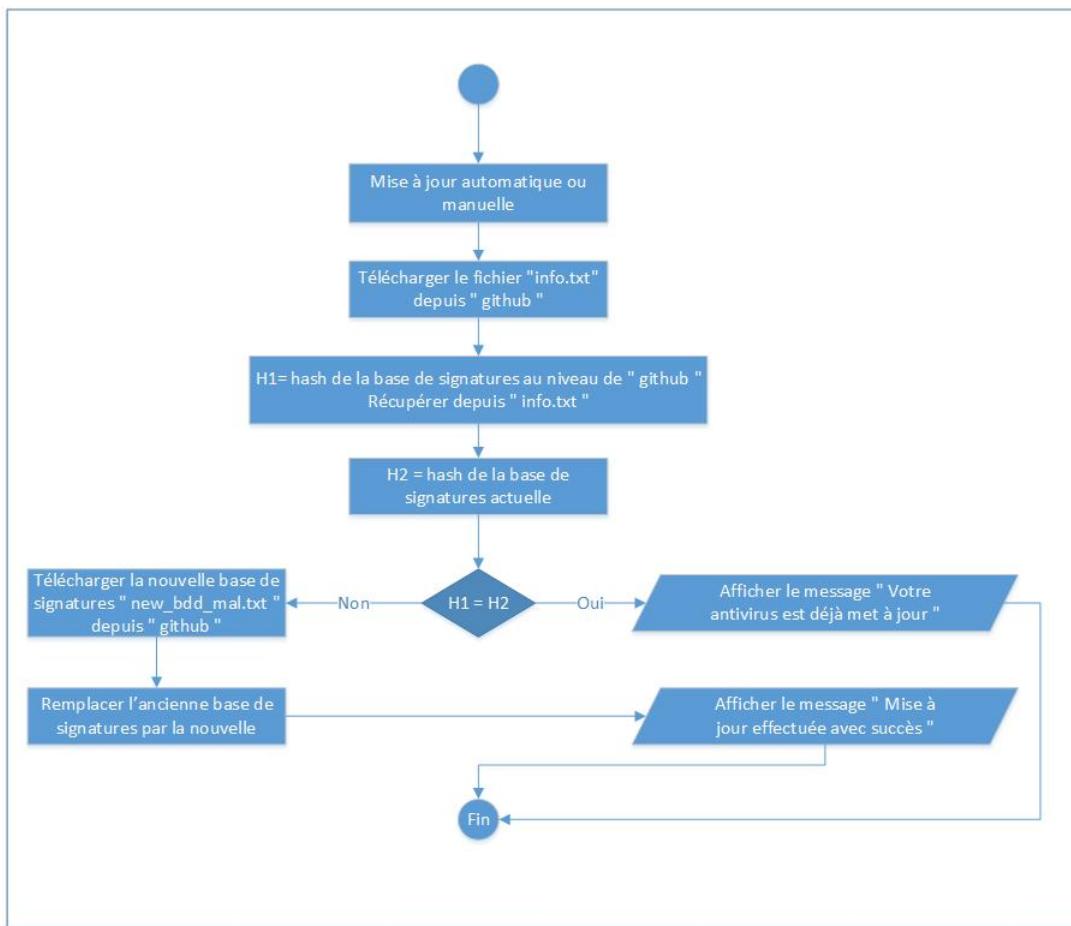


FIGURE 3.3 – Les étapes de la mise à jour de l'antivirus.

3.2.3.7 Les types de scan

L'antivirus examine (scan) le système à la demande : un fichier, un dossier ou tous les fichiers du disque. Un scan complet consomme beaucoup de ressources matérielles et du temps, mais il est conseillé de le faire de temps en temps.

Dans cette partie, nous allons proposer quatre manières pour effectuer un scan :

- Scan un seul fichier : il effectue une analyse d'un seul fichier sélectionné
- Scan un dossier sélectionné : il effectue une analyse d'un répertoire sélectionné
- Scan rapide : il effectue uniquement une analyse rapide du disque system "C : de l'ordinateur", seulement les fichiers aux extensions perceptibles à une infection (.exe, .ocx, .dll, .cpl) seront testés
- Scan tout le système : il effectue un scan de tous les disques de l'ordinateur.

3.2.3.8 La présentation des résultats d'analyse

Après avoir effectué un scan selon l'un des types cités précédemment, les résultats de scan seront présentés à l'utilisateur dans une grille, pour chaque type de scan, le rapport comportera les champs suivant :

- Nom du malware : un nom attribué par les éditeurs des malwares
- Type : c'est le type de malware (section 1.3.2, chapitre 1)
- Chemin : le chemin courant du malware détecté.

3.2.4 Autres fonctionnalités

- La langue d'utilisation : dans cette option, nous offrons à l'utilisateur la possibilité de choisir la langue, nous proposons deux langues : Anglais et Français
- Information sur la protection : cette fonctionnalité permet à l'utilisateur de visualiser des informations concernant l'état actuel de l'antivirus comme :
 - La liste des malwares détectés par notre antivirus classés par leur nom et type
 - Informations sur la base de signatures telles que sa taille "nombre de signatures", le nombre des virus et la version actuelle de l'antivirus
 - Quarantaine : cette fonctionnalité permet d'accéder au dossier quarantaine, visualiser son contenu.
- L'accès au site officiel de l'antivirus : permet à l'utilisateur de découvrir les fonctionnalités de l'antivirus, et les éventuels problèmes pouvant être rencontrés durant l'utilisation, télécharger le guide d'utilisation et actualités concernant la sécurité informatique et bien évidemment, soumettre des questions.
- Guide d'utilisation : un guide pratique expliquant la manière d'utilisation de l'antivirus et toutes ses tâches
- Mise à jour de programme : effectuera selon les mêmes étapes de la mise à jour de la base de signatures virales, au lieu de télécharger la base de signatures, des fichiers ".dll" seront téléchargés
- L'envoi d'un fichier suspect : permet à l'utilisateur d'envoyer un fichier suspect compressé, protégé par le mot de passe "infected" à l'équipe d'analyse de l'antivirus.

3.3 La façon de présenter l'antivirus

L'antivirus sera présenté au moyen d'une interface graphique, qui permettra de visualiser l'état actuel de la protection, modifier la langue, effectuer manuellement un scan et d'utiliser l'outil parseur PE pour avoir des détails techniques concernant un fichier PE, l'accès à cette interface peut se faire selon trois manières différentes :

- A partir d'un raccourci, qui permet d'appeler l'interface de l'antivirus
- A partir de l'icône de l'antivirus présentée dans la zone de notification "systray", en bas à droite de l'écran près de l'horloge par un simple clic
- A partir du menu Démarrer, par un clic sur Programmes suivi par la sélection du nom d'antivirus.

L'antivirus se lance automatiquement au démarrage de Microsoft Windows.

3.4 Fonctionnement de l'antivirus :

D'après cette conception, notre antivirus a deux tâches essentielles et primordiales, un parseur PE est un outil de scan

3.4.1 Parseur PE

L'utilisateur sélectionne un fichier depuis le système ou les disques amovibles pour afficher ses détails techniques, par un simple clic sur un bouton appelé "Parseur PE" qui permet d'accéder à une nouvelle page et un nouveau bouton "...", ce dernier permet d'accéder aux différents disques de votre système pour choisir un fichier par un clic sur le nom de fichier sélectionné,

suivi d'une validation du choix avec le bouton "OK", les résultats de l'analyse seront affichés dans une fenêtre graphique, ainsi que le chemin de fichier sélectionné.

3.4.2 Scanneur de malwares

3.4.2.1 phase de scan

L'utilisateur sélectionne un type de scan, à savoir ; scan rapide, scan d'un seul fichier, scan d'un dossier sélectionné ou un scan de tout le système, ensuite l'antivirus analysera les fichiers relatifs au scan sélectionné, et ce, comme suit :

- Parcourir la base de signatures, pour chaque signature, nous testons leur apparence dans le code binaire de fichier testé selon les étapes suivantes :
 - Récupérer un offset depuis la signature testée
 - Récupérer la suite hexadécimale relative a l'offset récupéré, depuis le code binaire du fichier testé
 - Comparer la suite de la signature testé avec la suite récupérée.
- Si les deux suites comparées sont identiques, nous sommes devant deux versions :
 - Scan d'un seul fichier : le fichier est signalé comme malware, et c'est la fin de scan
 - Autre type de scan : le fichier est signalé comme malware, ce qui nécessite la ré-exécution des étapes de scan dès le début pour le fichier suivant.
- Si les deux suites comparées sont différents, nous testons la signature qui suit et ainsi de suite, à la fin de cette opération, si aucune signature n'a été trouvée ; on est devant deux cas :
 - Scan d'un seul fichier : fin de scan
 - Autre type de scan : ce qui nécessite de refaire les étapes de scan dès le début pour le fichier suivant.

Le diagramme suivant montre les étapes de scan :

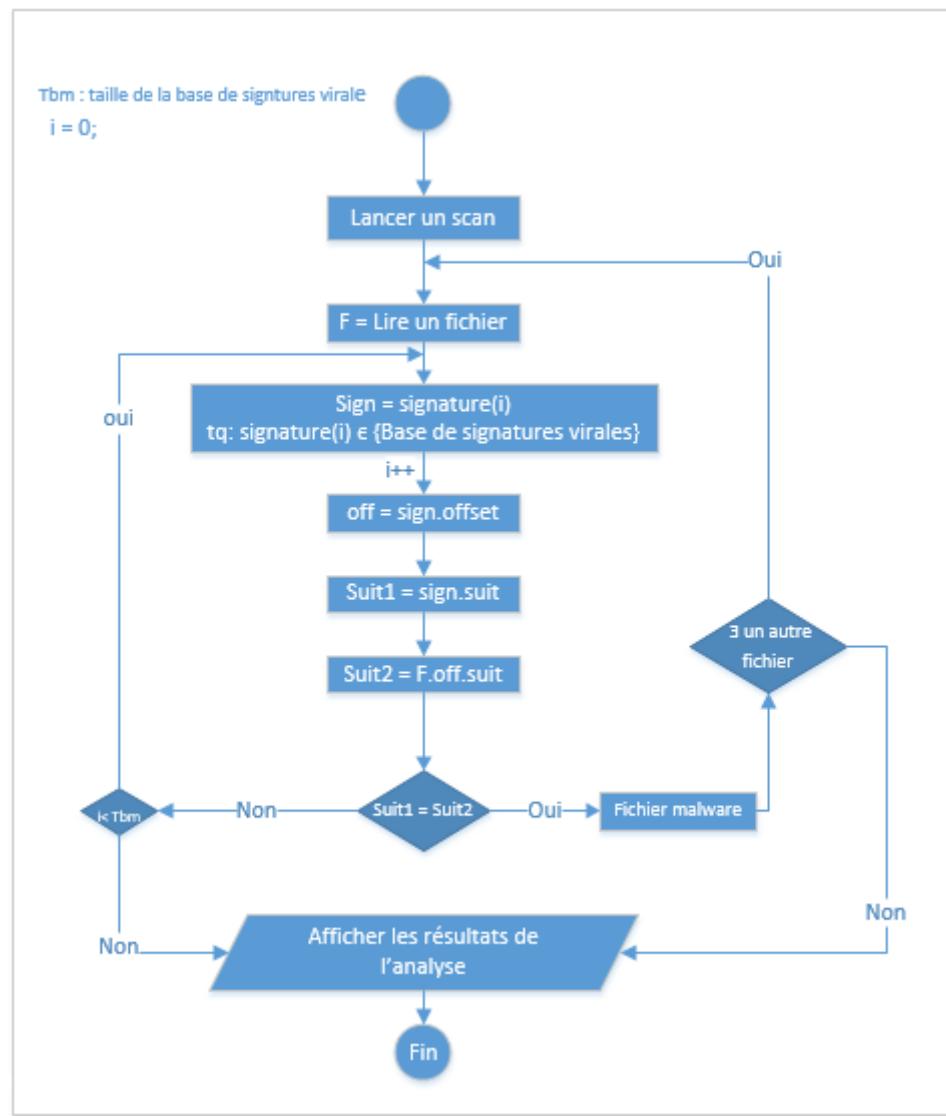


FIGURE 3.4 – Diagramme des étapes de scan.

3.4.2.2 Traitement des résultats de scan

Après la détection d'un malware, nous proposons les actions suivantes :

- **Supprimer** : permet de supprimer le fichier signalé comme malware depuis son emplacement d'origine ;
- **Mettre en Quarantaine** : si le fichier signalé ne peut ou ne doit pas être supprimé (fichier sensible, document personnel) alors il sera isolé jusqu'à une éventuelle opération, l'isolation se fait dans un dossier nommé "quarantaine"
- **Ne rien faire** : Si aucune action n'est souhaitable, le fichier est ignoré.

3.5 Gestion de la base de signatures virales

La détection d'un nouveau malware se fera par les éditeurs d'antivirus ou par un signalement (l'utilisateur envoie le malware compressé avec le mot de passe de décompression) de la part des utilisateurs, une fois détecté le malware sera analysé pour identifier son nom, son type,

son fonctionnement et son objectif.

Après la phase d'analyse, nous allons attribuer une signature de détection à ce malware, qui lui est propre, la génération de la signature virale commence par la récupération de la suite hexadécimale qui identifie de manière unique le malware, suivie par l'association de l'offset, nom et du type de malware à cette suite.

Pour chaque nouveau malware détecté, nous devons mettre à jour la base de signatures virales, cette mise à jour comporte plusieurs tâches qui sont gérées uniquement par les administrateurs d'antivirus, ces tâches sont :

- **Ajouter une signature** : permet à l'administrateur d'ajouter une nouvelle signature à la base de signatures, l'ajout d'une signature se fait selon les étapes suivantes :

- Sélectionner le malware à signer
- Choisir l'offset de la suite hexadécimale à l'aide de parseur PE, l'offset peut être la valeur de l'EntryPoint ou l'offset de l'un des sections ".data", ".code", ...
- Récupérer la suite relative à l'offset choisi
- Tester l'apparence de cette suite dans la base de fichiers légitimes afin d'éviter les faux positifs pour la signature générée
- Associer le nom et le type de malware qui ont été définis par l'analyste à la suite et l'offset généré. La génération d'une suite est terminée par l'insertion de cette dernière dans la base de signatures virale.

La figure 3.5 montre les étapes à suivre pour générer des signatures :

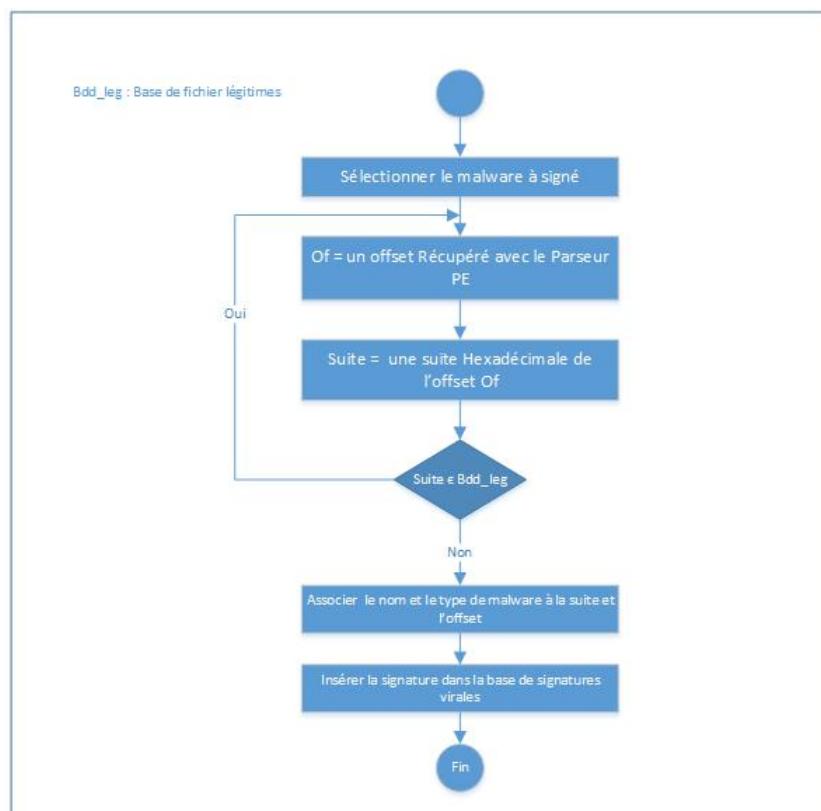


FIGURE 3.5 – Les étapes de génération d'une signature.

- **Afficher la base de signatures** : permet d'afficher le contenu de la base de signatures

- **Afficher la base de malwares** : permet de lister les noms de malwares
- **Supprimer une signature** : supprime une signature identifiée par le nom de malware
- **générer le fichier "info.txt"** : après chaque mise à jour, l'administrateur doit générer le fichier "info.txt" pour le charger sur GitHub.

3.6 Les faux positifs

Pour éviter ou minimiser les faux positifs, nous allons construire une base de données des fichiers légitimes, qui présentent les fichiers fondamentaux d'une installation Microsoft Windows fraîche.

Et pour chaque signature virale générée, nous allons tester son apparence dans cette base des fichiers légitimes, si cette dernière existe, nous devons régénérer une autre signature, et ce, jusqu'à l'arrivée d'une signature qui n'existe plus.

3.6.1 Description de la base de fichiers légitimes

La base des fichiers légitimes, comporte les fichiers déclarés par les éditeurs d'antivirus comme fichiers légitimes. Pour bien vérifier l'intégrité d'un fichier, nous utilisons la fonction d'hachage sha256, donc le fichier légitime sera stocké par son hash dans un fichier plat.

3.6.2 Gestion de la base de fichiers légitimes

Pour chaque nouveau fichier qui fait partie d'une nouvelle version Microsoft Windows ou mise à jour, l'administrateur met à jour la base de fichiers légitimes par l'insertion de ce fichier dans cette dernière, attribuer un nom au fichier qui est le hash de ce dernier .

Le diagramme de cas d'utilisation suivant donne une vision globale sur les tâches effectuées par l'administrateur d'antivirus.

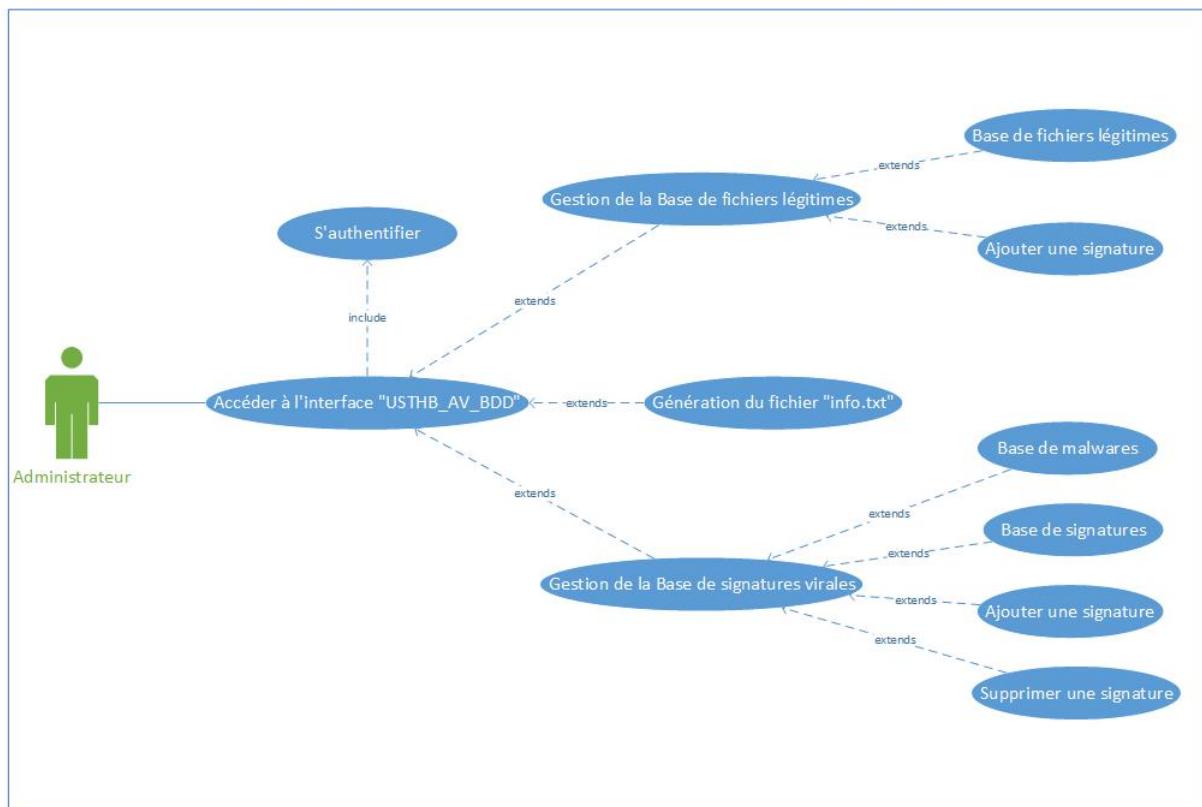


FIGURE 3.6 – Détail de la gestion de la base de signatures.

3.7 Conclusion

Dans ce chapitre, nous avons détaillé la conception de l'antivirus qui est composée de deux outils, **un parseur PE** et **un Scanneur de malwares**.

Le parseur PE permet de visualiser les informations techniques d'un fichier PE et pour le scanneur, nous avons adopté la technique anti-virale de recherche par signature de type suite Hexadécimale, et pour éviter les faux positifs nous avons proposé une base de signatures des fichiers légitimes.

Dans le chapitre suivant, l'implémentation de la solution anti-virale sera abordée.

Chapitre 4

Implémentation

4.1 Introduction

L'implémentation d'un projet consiste à réaliser au mieux les fonctionnalités décrites au niveau de la conception.

Dans ce qui suit, nous présentons l'environnement de développement, les fonctionnalités générales du l'antivirus ainsi que des captures d'écran illustrant cela.

4.2 Outils utilisés

4.2.1 L'environnement de développement (Visual Studio)

Pour implémenter notre antivirus nous avons opté pour l'utilisation de Microsoft Visual Studio comme un environnement de développement.

4.2.2 Visual Studio

Microsoft Visual Studio est un ensemble complet d'outils et de services destinés pour aider à créer des applications très variées, et pas seulement pour la plateforme Microsoft. Visual Studio connecte également l'ensemble de projets, équipes et parties prenantes. Désormais, les développeurs peuvent travailler avec une plus grande agilité de presque n'importe où, quel que soit l'outil de développement utilisé (par exemple : Eclipse ou Xcode).

4.2.3 Langages du Visual Studio

Visual studio est un environnement polyvalent très riche en termes de langage de programmation permettant aux développeurs de créer toute une gamme d'applications, il comporte 05 langages de programmations : Visual C++, Visual C#, Visual Basic, Visual F#, JavaScript.

4.2.4 Langage de programmation C#

Dans l'implémentation de notre antivirus, nous avons utilisé le langage C#. Visual C# est un langage de programmation multi paradigme moderne et généraliste qui permet de créer des applications à l'aide de Visual Studio. Par sa conception, C# se veut un langage simple, puissant, de type sécurisé et orienté objet. Les nombreuses innovations dont bénéficie C# autorisent un développement rapide d'applications tout en conservant l'expressivité et l'élégance des langues de type C.

4.2.5 Jimdo

Jimdo est un système de gestion de contenu web permettant de créer un site web. Utilisable directement en ligne, il se caractérise par sa facilité et sa rapidité d'utilisation.

Ce service gratuit s'inscrit dans la lignée du Web 2.0 puisqu'il remplace totalement un logiciel et que l'interface utilisateur, qui permet de créer les pages de son site, utilise la technologie WYSIWYG. Les sites sont automatiquement mis en ligne et hébergés sur les serveurs de l'entreprise dès leur création et les adresses des sites sont au départ de type nom-d-utilisateur.jimdo.com (cela fonctionne comme un sous-domaine du site Jimdo.com), mais il est également possible d'installer un nom de domaine personnel sur un site Jimdo [26].

4.2.6 GitHub

GitHub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. GitHub propose des comptes professionnels payants, ainsi que des comptes gratuits pour les projets de logiciels libres.

Le nom Github est composé du mot "git" faisant référence à un système de contrôle de version open-source et le mot "hub" faisant référence au réseau social batît autour du système Git [27].

4.3 Outil de gestion de la base de des signatures

4.3.1 Authentification

Pour accéder à la gestion de la base des signatures, l'administrateur doit s'authentifier à l'aide de son login et mot de passe, pour interdire l'accès illicite à la base de signatures, comme il est indiqué dans la fenêtre suivante (Figure 4.1) :

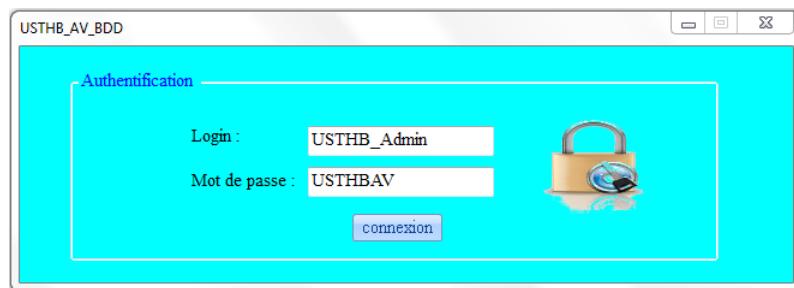


FIGURE 4.1 – L'authentification

La gestion de la base de signatures permet d'afficher, modifier et supprimer une signature, ainsi que de créer le fichier "info.txt", la figure 4.2 représente la page d'accueil avec trois fonctionnalités à savoir, la gestion de la base de fichiers légitimes et la gestion de signatures virales et la génération du fichier "info.txt".



FIGURE 4.2 – La page d'accueil de l'outil de gestion de la base de signatures.

4.3.2 Gestion de la base de fichiers légitimes

La gestion de la base de fichiers légitimes est composée de deux fonctionnalités :

4.3.2.1 Base de fichiers légitimes

Cette fonctionnalité permet d'afficher la base de fichiers légitimes ainsi que sa taille, la figure 4.3 suivante correspond à une base de 4140 fichiers légitimes.

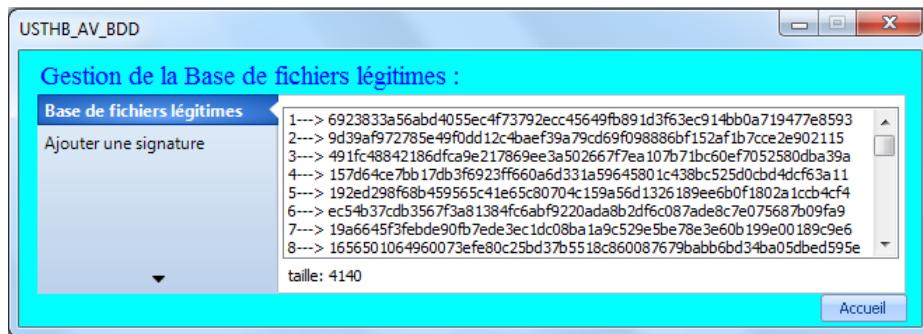


FIGURE 4.3 – La page de la base de fichiers légitimes.

4.3.2.2 Ajouter une signature

Pour la mise à jour de la base de signatures, l'administrateur peut ajouter une nouvelle signature en suivant les étapes suivantes :

- Sélectionner le fichier légitime à signer en appuyant sur le bouton "..." , qui sera suivi par l'affichage du chemin de fichier sélectionné dans le champ de saisie.
- Un clic sur le bouton "ajouter" permet d'insérer le fichier sélectionné dans la base de fichiers légitimes (Figure 4.4)

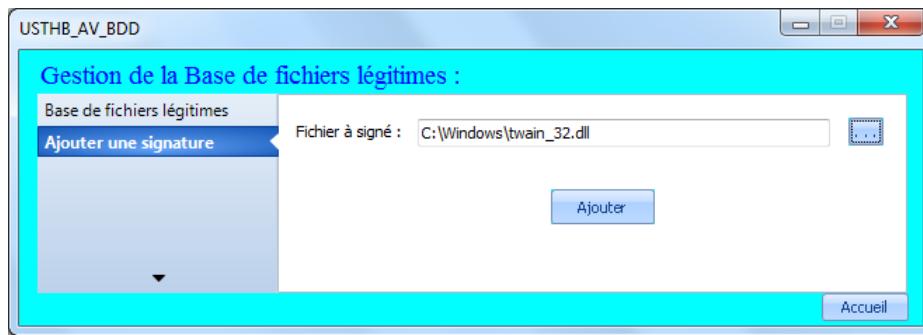


FIGURE 4.4 – La page d'ajouter un fichier légitime.

4.3.3 Gestion de la base de signatures virales

La gestion de la base de signatures virales est composée de quatre fonctionnalités :

4.3.3.1 Base de malwares

Cette fonctionnalité permet d'afficher la base de malwares ainsi que sa taille, la figure 4.5 correspond à une base de 7404 malwares.

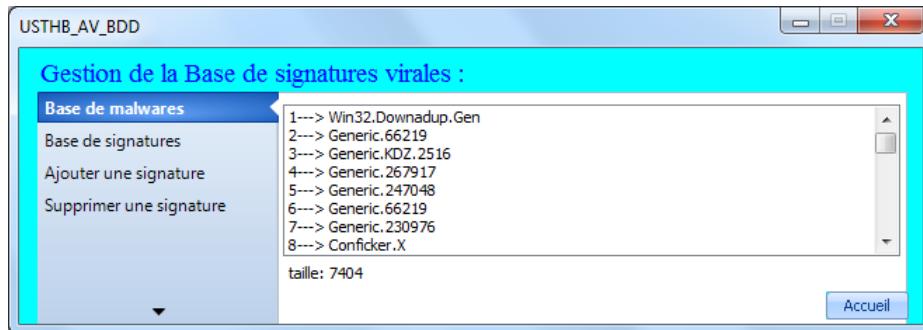


FIGURE 4.5 – La base de malwares.

4.3.3.2 Base de signatures virales

Cette fonctionnalité permet d'afficher la base de signatures virales, ainsi que sa taille, la figure 4.6 représente une base de 7325 signatures virales :

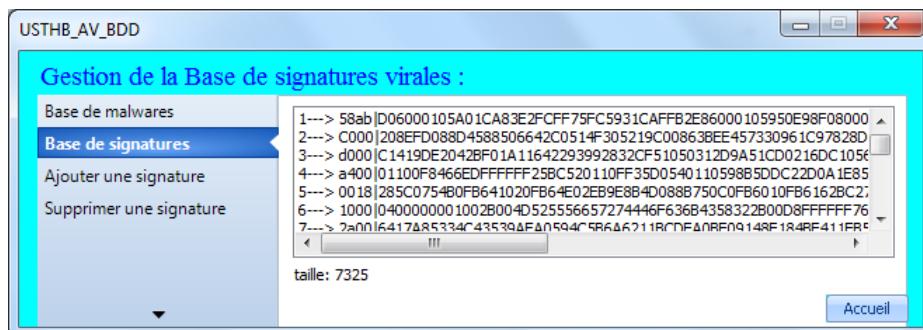


FIGURE 4.6 – La base de signatures virales.

4.3.3.3 Ajouter une signature

Comme nous avons cité dans le chapitre précédent, une signature virale est composée de quatre champs : un offset, une suite hexadécimale, nom et type de malware, pour ajouter une signature, nous devons suivre les étapes suivantes :

- Sélectionner le malware à signer avec le bouton “...”
- Définir l’offset de la suite hexadécimale
- Récupérer la suite hexadécimale avec le bouton ”Récupérer la suite hexadécimale”
- Définir un nom qui n’existe pas déjà dans la base de signature virale pour ce malware.
- Définir le type de ce malware
- Cliquer sur le bouton ”ajouter” pour ajouter la nouvelle signature (Figure 4.7).

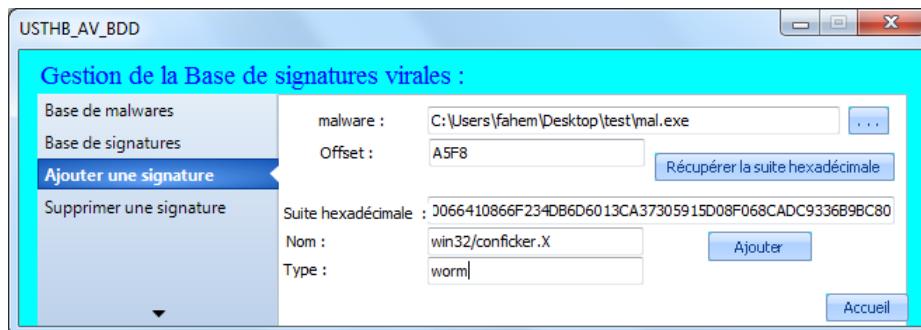


FIGURE 4.7 – Ajouter une signature virale.

4.3.3.4 Supprimer une signature

La suppression d’une signature virale nécessite d’indiquer le nom du malware, dont la signature à supprimer, cette opération s’effectue en deux phases :

- Saisir le nom du malware
- Valider la suppression en cliquant sur bouton ”supprimer” (Figure 4.8).



FIGURE 4.8 – Supprimer une signature virale.

4.3.4 Génération du fichier ”info.txt”

Pour créer le fichier ”info.txt” l’administrateur doit cliquer sur le bouton ”Génération du fichier ”info.txt” de la page d’accueil, une fenêtre, contenant des renseignements devant être mis à jour concernant la version de la base de signatures et la version de moteur,s’affiche puis valider par un clic sur le bouton ”Générer le fichier” (Figure 4.9).

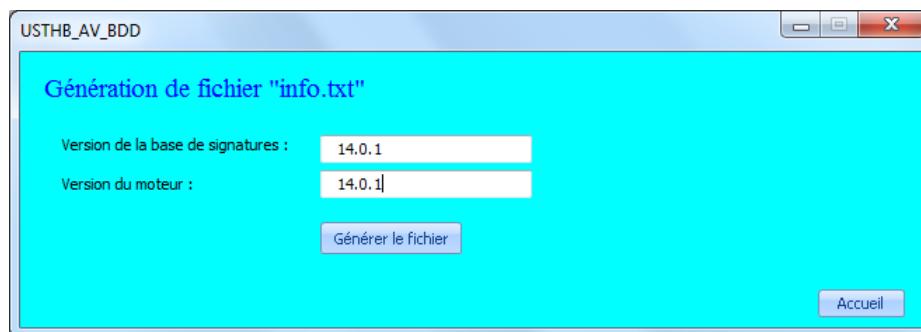


FIGURE 4.9 – Génération du fichier ”info.txt”.

4.4 Antivirus

Dans cette partie, nous allons présenter le fonctionnement de notre antivirus.

4.4.1 Page d'accueil

Dès que l'utilisateur exécute l'antivirus, une page d'accueil sera affichée (Figure 4.10), portant les informations et les fonctionnalités suivantes :

– **Les informations :**

- Le nom de l'antivirus (USTHB_ AntiVirus)
- L'état actuel de l'antivirus relatif à la dernière mise à jour,dernière analyse, et la version courante de l'antivirus.

– **Les fonctionnalités :**

- La langue : L'utilisateur coche la langue à utiliser
- Deux boutons, l'un pour réduire la fenêtre et l'autre pour la fermer
- Une barre de tâches composée de cinq raccourcis :
 - **Gestionnaire des tâches** : sert à afficher les programmes, les processus et les services en cours d'exécution sur l'ordinateur et surveiller les performances de l'ordinateur ou pour fermer un programme
 - **Aide** : permet l'accès au guide d'utilisation
 - **Exit** : permet de quitter le programme
 - **Accès internet** : permet un accès rapide sur l'Internet
 - **Commande cmd** : une fonctionnalité qui offre un point d'entrée pour la saisie de commande Windows.
- Sur la gauche de la page d'accueil, se trouve une barre verticale dont le clic fait apparaître le menu de l'antivirus.

La figure 4.10 montre la page d'accueil de l'antivirus :



FIGURE 4.10 – Page d'accueil de l'antivirus .

4.4.2 Page Menu

Nous trouvons dans cette page les tâches principales de l'antivirus, listées comme suit : Antivirus, Parseur PE, Maintenance, USTHB_AV, Fichier suspect et protection.

La figure 4.13 montre la page Menu de l'antivirus :



FIGURE 4.11 – Page Menu de l'antivirus .

4.4.2.1 Scanneur

L'utilisateur clique sur le champ "Scanneur" pour lancer un scan, une nouvelle page sera affichée (figure 4.12), qui contient les quatre types de scan, à savoir ; scan un dossier sélectionné, scan un seul fichier, scan rapide, et scan tout le système.



FIGURE 4.12 – Les scans .

Selon l'action souhaitable l'utilisateur choisie une des fonctionnalités suivantes :

Scan un seul fichier :

• Fonctionnement :

Ce champ est choisi une fois le scan aura été pour un seul fichier, le clic sur "scan un seul fichier" permet d'accéder au chemin de ce fichier, après validation (par un clic sur le bouton OK) le chemin est déclaré dans le champ "fichier analysé".

Une fois l'analyse est achevée, trois situations possibles se présentent :

- Le fichier testé est un fichier légitime : un message "le fichier est légitime" est affiché
- Le fichier testé n'est pas un malware un message "le fichier n'est pas un malware" est affiché
- Le fichier testé est un malware, dans ce cas, des informations suivantes sont affichées :
 - * Le type de malware détecté
 - * Une image relative à ce type de malware
 - * Une sonnerie d'alerte
 - * Un lien vers la page des résultats d'analyse.

La figure 4.6 représente un test d'un fichier "adobe.exe" :



FIGURE 4.13 – Un virus détecté .

L'utilisateur peut abandonner ou lancer un nouveau scan par un clic sur le bouton "Nouveau scan", ou visualiser les résultats d'analyse par un clic sur le lien "Résultats d'analyse".

- **Résultat d'analyse :** Après un clic sur le lien "Résultats d'analyse", ces résultats s'affichent dans une nouvelle page, ainsi que les actions qui peuvent être appliquées à ce malware, à savoir ; supprimer, mettre en quarantaine, et ne rien faire, la figure 4.14 correspond aux résultats d'analyse du fichier "adobe.exe".

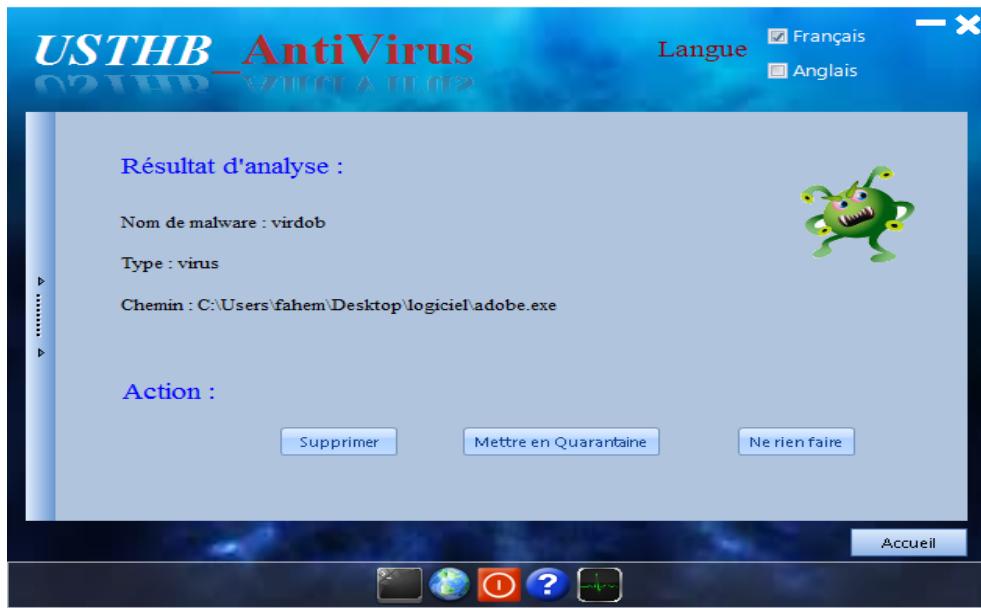


FIGURE 4.14 – L'affichage des résultats d'analyse .

Scan un dossier sélectionné :

- **Fonctionnement :**

Ce champ est choisi une fois le dossier à scanner est bien précis, le clic sur "scan un dossier

sélectionné” fait apparaître une nouvelle page (figure)

- Le dossier à scanner est sélectionné en cliquant sur le bouton ” . . . ” cette action permet un accès au chemin de ce dossier
- après validation (par un clic sur le bouton OK) le chemin est déclaré dans le champ ”répertoire analysé”
- Au cours de l’analyse des informations ponctuelles vont s’afficher à savoir :
 - * Une barre de progression : déclare le taux d’avancement de l’opération
 - * Fichier examiné : dans ce champ s’affiche le chemin du fichier en cours d’analyse
 - * Temps d’analyse : dans ce champ s’affiche le temps écoulé depuis le début d’analyse
 - * Nombre totale : dans ce champ s’affiche le nombre total des fichiers à analyser
 - * Malware détecté : dans ce champ s’affiche le nombre final des malwares détectés.
- A tout moment l’utilisateur peut effectuer un nouveau type de scan par un simple clic sur le bouton ”nouveau scan” qui permet le retour à la page scanneur ou retourner carrément à la page d’accueil par le bouton ”accueil”
- Une fois l’analyse est achevée, deux situations possibles se présentent :
 - * Aucun malware n’est détecté : un message signalant l’absence des malwares s’affiche
 - * Présence des malwares : une sonnerie d’alerte est déclenchée, signalant la présence de ces malwares, le nombre de ces derniers sera communiqué dans le champ ”malware détecté”, un nouveau bouton ”Résultat d’analyse” sera affiché, pour accéder à ces résultats, il suffit de cliquer sur ce bouton.

La figure correspond à une analyse d’un dossier nommé logiciel, dont le chemin est C:\User\fahem\Desktop\logiciel, le résultat de l’analyse est :

- Dernier fichier examiné nommé adobe.exe
- Temps d’analyse : 0 :01 :47
- Nombre total : 12
- Malware détecté : 3



FIGURE 4.15 – Scan un dossier sélectionné .

- **Résultat d'analyse :**

Après un clic sur le bouton ”résultat d’analyse”, les résultats s’affichent dans une grille, avec deux choix de traitement :

- La suppression ou la mise en quarantaine des malwares un par un, et ce, comme suit : sélectionner le malware à supprimer ou à mettre en quarantaine, après on valide l’action par le bouton ”Supprimer” ou ”mettre en quarantaine”, si l’action choisie est faisable, le malware ne s’affiche plus dans la grille, et vice versa
- Une action collective de traitement pour tous les malwares détectés, à savoir ; supprimer, mettre en quarantaine, ou ne rien faire, qui sera choisie dans le ComboBox, suivie d’une validation par le bouton ”Appliquer”.

Le bouton ”Quitter” permet d’abandonner les résultats d’analyse, et un retour automatique à la page antivirus(Figure 4.16).

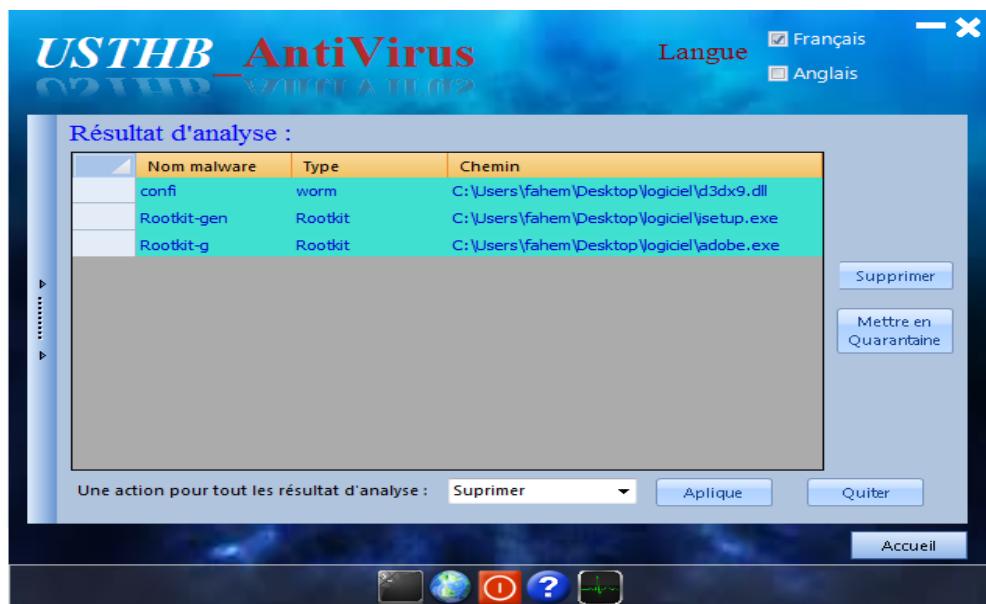


FIGURE 4.16 – Résultats d’analyse d’un dossier sélectionné .

Scan rapide :

Le fonctionnement de ce scan est le même que le scan d’un dossier sélectionné, sauf que la cible d’analyse dans le scan rapide est prédéfinie ”C :\”; contrairement au scan d’un dossier sélectionné où l’utilisateur doit sélectionner la cible d’analyse.

Le traitement des résultats d’analyse dans le scan rapide est basé sur les mêmes principes de scan d’un dossier sélectionné.

Scan tout le système

Le fonctionnement et le traitement des résultats de ce scan sont les mêmes que le scan rapide, sauf que dans le scan de tout le système, l’ensemble des disques de l’ordinateur seront analysés.

4.4.2.2 Parseur PE

Pour observer le contenu d'un fichier exécutable, l'utilisateur clique sur le champ "Parseur PE" de la page Menu, une nouvelle page apparaît, ensuite l'utilisateur doit cliquer sur le bouton "..." afin d'accéder au chemin du fichier à analyser, après validation (par un clic sur le bouton OK) le chemin est déclaré dans le champ "fichier à analyser", deux cas possibles ont été constatés :

- Le fichier analysé à un format PE non valide : un message "Format PE non valide"
- Le fichier analysé à un format PE valide : les résultats ; EntryPoint, PE détail, les sections PE, la liste des fonctions API importées et les éventuels packers utilisés seront affichés dans un nouveau champ.

La figure 4.17 représente les résultats d'analyse d'un fichier PE valide :



FIGURE 4.17 – Le parseur PE de l'antivirus .

4.4.2.3 Maintenance

Pour effectuer une mise à jour, l'utilisateur doit cliquer sur le champ "Maintenance" de la page menu, cette action va faire apparaître une nouvelle page (Figure) comportant :

- Deux boutons, le premier intitulé "mise à jour de la base de signatures" le deuxième "mise à jour du programme"
- Des renseignements concernant la version actuelle de la base de signatures et de programme ainsi que le nombre de signatures
- Un message encourageant la mise à jour.

La mise à jour se déroule par un clic sur l'un des boutons cités précédemment, deux cas sont possibles :

- Il n'existe pas une nouvelle mise à jour : dans ce cas un message "votre antivirus est déjà mis à jour" est affiché
- Il existe une mise à jour : une barre de progression, qui déclare le taux d'avancement de l'opération, une fois cette dernière est terminée, un message "mise à jour effectuée avec succès" est affiché, en cas d'échec, ce dernier sera déclaré via un message.

La figure 4.18 présente la page de mise à jour de l'antivirus :



FIGURE 4.18 – La mise à jour de l'antivirus .

4.4.2.4 USTHB_AV

Pour avoir les dernières actualités sur l'antivirus, télécharger le fichier d'installation de l'antivirus, guide d'utilisation,...

L'utilisateur doit cliquer sur le champ "USTHB_AV" de la page Menu, cette action permet à l'utilisateur d'accéder au site web de l'antivirus <http://usthbav.jimdo.com/> (Figure 4.19) :



FIGURE 4.19 – Le site web de l'antivirus .

4.4.2.5 Fichier suspect

Pour envoyer un fichier suspect, l'utilisateur doit cliquer sur le champ "Fichier suspect" de la page Menu, cette action va faire apparaître une nouvelle page (Figure) portant un message décrivant les étapes obligatoires à suivre pour envoyer un fichier suspect, qui sont :

- Archiver le fichier suspect à envoyer en ".rar"
- Définir un mot de passe pour le fichier archivé
- Cliquer sur le bouton "..." afin d'accéder au chemin du fichier suspect à envoyer, après validation (par un clic sur le bouton OK) le chemin est déclaré dans le champ "fichier à envoyer"
- Décrire le fichier suspect au niveau de champ "Sujet"
- Accompagner le fichier archivé par le mot de passe de décompression, au niveau de champ "mot de passe de décompression"
- Valider l'envoi du fichier suspect par un clic sur le bouton "Envoyer".

Après la validation de l'envoi du fichier suspect, une barre de progression sera affichée, une fois cette dernière terminée, un message "Fichier envoyé avec succès" est affiché, en cas d'échec, ce dernier sera déclaré via un message.

La figure 4.20 décrit l'envoi d'un fichier suspect, nommé "suspect.rar", sujet "fichier suspect" et un mot de passe de décompression "Ss000000".

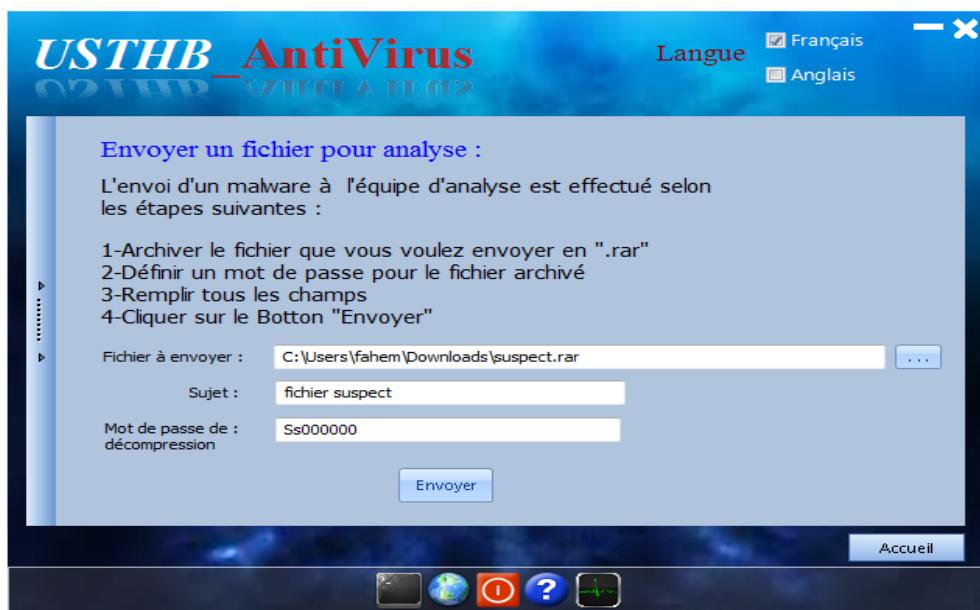


FIGURE 4.20 – L'envoie d'un fichier suspect .

4.4.2.6 Protection

Pour visualiser les informations relatives à l'antivirus, l'utilisateur doit cliquer sur le champ "Protection" de la page Menu, cette action va faire apparaître une nouvelle page, portant trois paramètres, détection, informations et quarantaine, indiqué ci-dessous :

- **Détection :** permet à l'utilisateur de voir la liste des malwares classée par leur nom et type, détectés par l'antivirus(Figure 4.21)



FIGURE 4.21 – Les malwares détectés par l'antivirus .

- **Information :** Cette partie sert à indiquer les informations relatives à la version actuelle de l'antivirus, qui sont : version de la base virale, nombre de signatures, nombre de malwares détectés et la version du moteur (Figure 4.22)



FIGURE 4.22 – Les informations relatives à l'antivirus .

- **Quarantaine :** Cette option permet à l'utilisateur de visualiser la liste des malwares, déjà mis en quarantaine, la figure 4.23 représente une quarantaine de trois malwares.



FIGURE 4.23 – Quarantaine .

4.5 L'accès à l'interface de l'antivirus

L'utilisateur peut accéder a l'antivirus via trois méthodes différentes :

- **A partir du Menu démarrer** : la figure 4.24 présente l'apparence de l'antivirus dans le Menu démarrer.

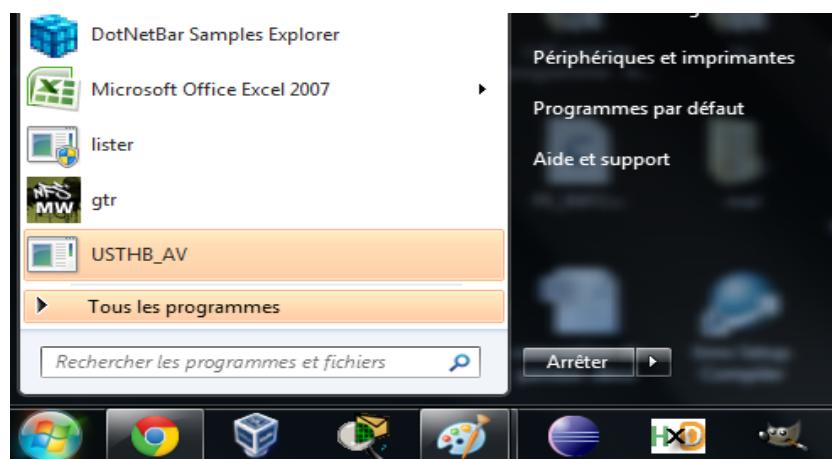


FIGURE 4.24 – Menu démarer.

- **A partir d'un raccourci** : voir la figure 4.25



FIGURE 4.25 – Raccourci au Bureau.

- **A partir de la zone de notification :** la figure 4.26 représente l'apparence de notre antivirus au niveau de la zone de notification ainsi que les tâches que l'on peut effectuer depuis cette zone.

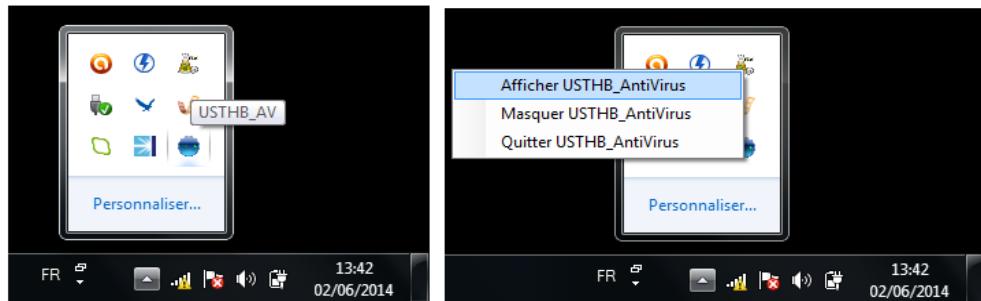


FIGURE 4.26 – Zone de notification "Systray".

Lancement de l'antivirus au démarrage de Windows

La figure 4.27 représente la liste des programmes à lancer au démarrage de Windows

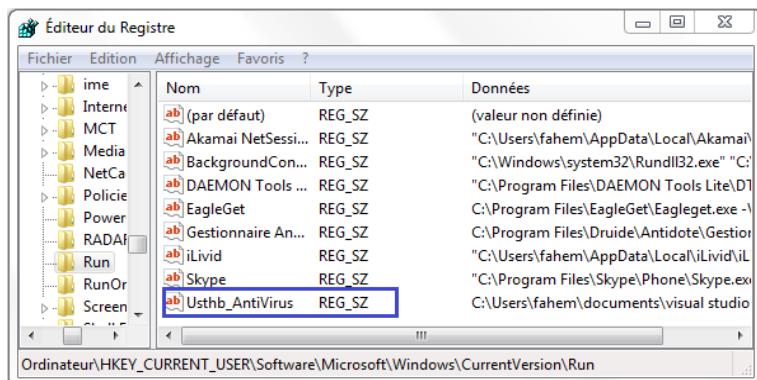


FIGURE 4.27 – Lancement de l'antivirus au démarrage de windows.

4.6 Conclusion

Dans ce chapitre nous avons vu l'ensemble de fonctionnalités de l'antivirus, côté administrateur "gestion de la base de signatures", côté utilisateur "antivirus".

Puis, à travers les impressions écrans, nous avons vu les interfaces, de l'application, illustrant chacune une fonctionnalité.

Chapitre 5

Analyse de malwares

5.1 Introduction

L'analyse de malwares (malware analysis) est aujourd'hui devenue une activité très importante dans le cadre de la gestion des incidents de sécurité informatique. Les organisations qui prennent sérieusement en compte la sécurité de leurs réseaux sont souvent confrontées à des fichiers suspects capturés grâce à leurs antivirus, IDS et systèmes de supervision sécurité, ou encore lors d'analyses forensics. Il est alors nécessaire d'analyser rapidement ces fichiers pour déterminer s'il s'agit de code malveillant connu, d'une attaque ciblée ou bien d'un faux positif. Connaître le comportement d'un malware est capital pour déterminer si d'autres machines sur le réseau peuvent être compromises ou non, et pour décider comment réagir.

L'analyse statique de code exécutable par désassemblage a beaucoup de succès et permet d'obtenir d'excellents résultats, cependant la complexité de ce processus demande un long apprentissage, beaucoup de temps et d'énergie. De plus il n'est pas rare de rencontrer des codes malveillants protégés contre le désassemblage et le débogage, ce qui peut ralentir considérablement l'analyse.

L'analyse dynamique de code malveillant est une autre méthode qui consiste à faire exécuter un échantillon de code sur une plate-forme conçue pour observer toutes ses actions. Dans la plupart des cas, c'est une approche très efficace et rapide pour déterminer la nature et le comportement du code malveillant, au moins dans un premier temps.

5.2 Objectifs de l'analyse de malwares

Voici quelques-uns des objectifs habituels d'une analyse de malware :

- Vérifier si un fichier suspect est effectivement malveillant ou s'il s'agit d'un faux positif
- Déterminer s'il s'agit d'un code malveillant générique (connu ou non) ou bien d'une attaque ciblée
- Obtenir des informations sur la source de l'attaque
- Déterminer toutes les actions du code malveillant sur le système local : fichiers créés/modifiés/supprimés, processus lancés/stoppés, clés de registre, services installés, ...
- Déterminer toutes ses actions sur le réseau : connexions sortantes, réPLICATION, serveur en écoute, ...
- Déterminer ses "fonctionnalités" malveillantes : virus, ver, cheval de Troie, porte dérobée, bombe logique, rootkit, keylogger, téléchargement, relais de spam, déni de service, botnet,

- Déterminer quelles sont les machines ciblées et lesquelles pourraient être vulnérables ou déjà compromises sur le réseau
- Fournir des informations pour mieux cibler une éventuelle analyse forensics d'autres machines (noms de processus, empreintes MD5 de fichiers, clés de registre, mots clés, ...)
- Décider quelles actions entreprendre pour gérer l'incident
- Comprendre comment la compromission s'est déroulée afin d'améliorer les protections à l'avenir.

5.3 Où trouver des malwares ?

Il existe toute une communauté autour de l'analyse de malwares : forums, sites, blogs, etc. Certains de ces sites mettent à disposition des échantillons permettant d'effectuer des analyses. Quelques adresses :

- **KernelMode** : Forum d'échange et d'analyse de malwares. Met à disposition des lecteurs des échantillons [28]
- **Malware.lu** : Excellent site mettant à disposition des échantillons de malwares et proposant des analyses [29]
- **Contagio Exchange** : Site permettant d'échanger et de télécharger des malwares [30]
- **CrowdRE** : Plate-forme collaborative d'analyse de malwares [31]
- **VadeRetro Sales malwares** : Articles et tutoriaux sur les malwares [32]
- **Malekal's Site** : Excellent site consacré à l'entraide informatique. De nombreux exemples d'analyses comportementales [33]
- **Malware Analysis & Diagnostics** : Analyse de malwares [34]
- **FireEye Malware Intelligence Lab** : Site de recherche et d'analyses de malwares [35]
- **Evil3ad** : Site d'analyse de malwares [36]
- **M86SecurityLabs** : Site consacré aux attaques (malwares, cybercriminalité) [37]
- **Fun in malwares analysis** : Tutoriaux sur l'analyse de malwares [38].

Les blogs des compagnies antivirales publient aussi de nombreuses analyses :

- **Blog de Kaspersky** : [39]
- **Blog d'Eset** : [40]
- **Blog de Norton** : [41]
- **Blog de Sophos** : [42].

5.4 Construire un laboratoire sécurisé d'analyse de malwares

Dans la section précédente, nous avons vu différentes méthodes pour récupérer des malwares. Maintenant comment les étudier en toute sécurité ? Il est bien sûr "impensable" de les exécuter et de les analyser sur un poste de travail. Il convient donc de créer un environnement spécial permettant de les analyser en toute sécurité.

5.4.1 La virtualisation

La virtualisation a révolutionné de nombreux domaines de l'informatique dont le domaine de l'analyse des malwares. Il devient ainsi simple et rapide de tester des malwares dans des environnements contrôlés afin de découvrir leurs actions.

Parmi les outils les plus utilisés dans le domaine de la virtualisation personnelle, nous pouvons citer principalement **Vmware** [43] (payant) ainsi que **VirtualBox** [44] (gratuit). Par contre certains malwares possèdent des fonctionnalités de détection d'environnements virtuels et stoppent leur exécution.

5.4.1.1 Avantages de la virtualisation

- **Coût et flexibilité** : une seule machine physique permet de simuler deux machines ou plus, et de construire différentes architectures réseau suivant les besoins.
- **Sécurité** : les machines virtuelles peuvent être connectées par un réseau virtuel totalement indépendant de tout réseau opérationnel, sans risque d'infecter d'autres machines.
- **Rapidité et efficacité** : une machine virtuelle peut être stoppée, restaurée et redémarrée en quelques secondes.
- **Facilité d'emploi** : n'importe quel état de fonctionnement peut être sauvegardé dans un "snapshot" en quelques secondes. Il est ensuite très simple de restaurer tout snapshot précédent, et de comparer plusieurs états entre eux.
- Les machines virtuelles peuvent être facilement copiées, dupliquées et modifiées pour constituer une bibliothèque de toutes les versions d'un système d'exploitation ou d'une application.

5.4.1.2 Inconvénients de la virtualisation

- Il existe de nombreuses techniques connues utilisables par des malwares pour détecter s'il s'exécutent dans une machine virtuelle. Dans ce cas ils peuvent éviter de s'exécuter ou perturber l'analyse.
- Certains environnements de virtualisation peuvent être vulnérables et permettre à un code malveillant spécifiquement conçu de "s'échapper" d'une machine virtuelle pour compromettre le système hôte. Il est donc utile de prendre quelques précautions lors de toute exécution de code malveillant.

5.4.2 Considérations du système d'exploitation

Les malwares comportent radicalement différent selon le système d'exploitation lequel sont exécutés sur. Certains malwares ne peuvent fonctionner que sur les systèmes d'exploitation Microsoft Windows Server, tandis que d'autres programmes malveillants ne peuvent fonctionner que sur des versions spécifiques du noyau Linux. Malware qui est installé sur un hôte Microsoft Windows Vista peut crasher complètement le système, tandis que le même malware installé sur un système Microsoft Windows 7 pourrait rejoindre un réseau de zombies (botnet). C'est une nécessité d'avoir une variété de systèmes d'exploitation disponibles lors de l'analyse des programmes malveillants. Donc il faut avoir différents systèmes d'exploitation comme Microsoft Windows XP, Microsoft Windows 7, Microsoft Windows Server 2003, et l'une des distributions Linux les plus populaires modernes comme Ubuntu pour servir d'hôtes infectés.

5.4.3 L'isolement du réseau

Des précautions supplémentaires doivent être prises en ce qui concerne l'emplacement des hôtes d'analyse des malwares sur le réseau. Les vers (Worms) et autres types de programmes malveillants peuvent avoir la fonction "auto-réPLICATION", il est donc très probable que l'exécution de malware sur une machine du réseau peut conduire à d'autres hôtes sur ce réseau. Un argument supplémentaire pour isoler les machines de laboratoire de l'Internet est d'empêcher les auteurs des logiciels malveillants de savoir l'existence de l'analyste. Il est tout à fait probable que le malware exécuté est configuré pour "phone home" à une commande & contrôle le serveur qui permet à l'auteur de savoir que vous avez exécuté le malware. À ce stade, l'attaquant pourrait commencer l'exécution de commandes sur le système de laboratoire pour tenter de désactiver ou déjouer les tentatives d'analyse.

La figure 5.1 donne un exemple de laboratoire d'analyse de malwares.

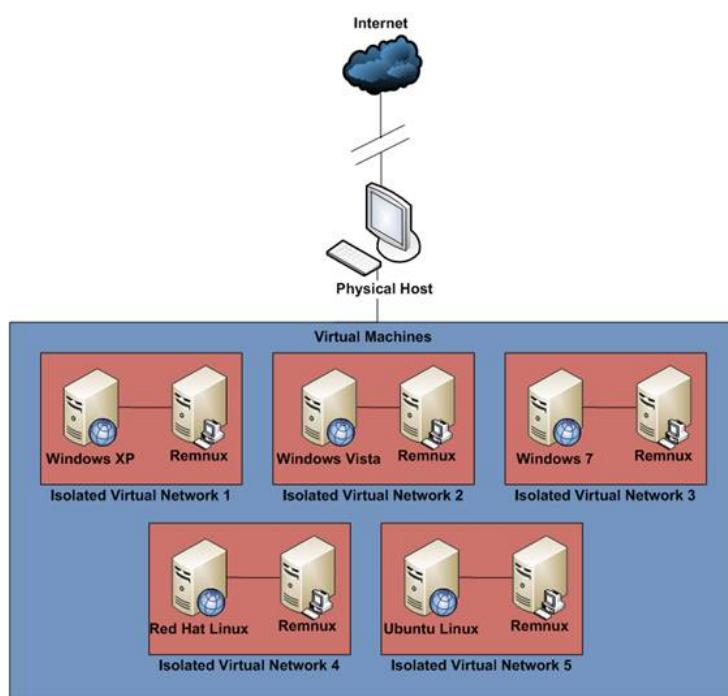


FIGURE 5.1 – Exemple d'un laboratoire d'analyse de malwares.

5.4.4 Les distributions dédiées à l'analyse de malwares

Il existe des distributions consacrées spécialement à l'analyse de malwares, une d'elle est particulièrement connue : **REMnux** (Figure 5.2). C'est une distribution Ubuntu conçue spécialement pour l'analyse de malwares. Elle contient des outils pour analyser la mémoire, des documents pdf vétrolés, le réseau, ...

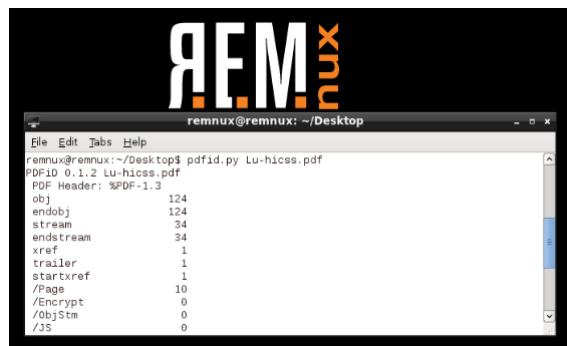


FIGURE 5.2 – Remnux, distribution pour l'analyse de malwares.

Il existe aussi une distribution nommée **Zero Wine**(Figure 5.3) qui permet d'analyser le comportement d'un malware : le malware est exécuté dans un bac à sable (sandbox) puis de nombreux rapports sont générés.

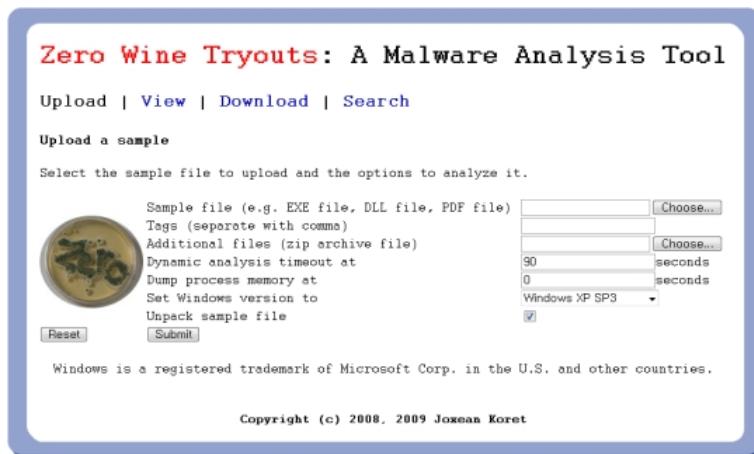


FIGURE 5.3 – Distribution ZeroWine pour l'analyse de malwares.

5.5 Les méthodologies d'analyse de malwares

5.5.1 L'analyse statique

5.5.1.1 Définition

L'analyse statique de malwares consiste à explorer le contenu des fichiers suspects à l'aide de divers outils, dans le but d'extraire le maximum d'informations sans exécuter le code malveillant qu'ils pourraient contenir. A ce stade il s'agit uniquement d'observer le contenu "visible". Le premier outil à employer dans ce cas est toujours un afficheur texte/hexadécimal.

Une analyse directe de ce type permet souvent de révéler beaucoup d'informations utiles grâce aux chaînes de caractères qui apparaissent en clair, aux entêtes de fichiers, aux métadonnées, etc. Il est même parfois possible d'extraire des données camouflées par des méthodes simples (i.e. "XOR"), sans recourir à des méthodes d'analyse sophistiquées. Dans le cas de document malformés (par exemple des exploits pour MS Office, PDF, ...), des outils de "file carving" peuvent souvent extraire les fichiers exécutables malveillants stockés à l'intérieur.

Voici le type d'informations utiles qu'il est souvent possible d'extraire :

- Types de fichiers
- Présence de code malveillant connu ou non
- Adresses IP, URLs, noms de domaines ou de machines (cibles du code malveillant, serveur de téléchargement ou bien connexion retour vers un serveur de contrôle)
- Date de création des fichiers, auteur, organisation et autres informations (méta-données dans les documents)
- Encodages utilisés (qui peuvent trahir une provenance exotique)
- Présence possible de shellcodes ("NOP sleds" : typiquement une longue suite d'octets 0x90)
- Mots clés particuliers qui peuvent indiquer une attaque ciblée : nom de l'entreprise, noms de serveurs, d'applications ou d'utilisateurs spécifiques, mots de passe internes, etc.

5.5.1.2 Le désassemblage

Le désassemblage (reverse engineering) consiste à ouvrir un fichier exécutable avec des outils tel que **IDA pro**, ou **Metasm** par exemple afin de retrouver et d'analyser le code assembleur correspondant au contenu binaire du fichier. Certains outils avancés peuvent même offrir des fonctions de décompilation permettant de reconstruire le code source de haut niveau employé au départ (en général C).

En analysant le code, il est ainsi possible d'étudier le comportement de chaque partie du programme et d'en déduire les fonctionnalités.

L'avantage de cette méthode est qu'en théorie il est possible d'analyser tout le code, même celui ne s'exécutant que sous certaines conditions particulières. De nombreux malwares ont par exemple des fonctions pour ne pas s'exécuter dans un environnement virtuel, ce qui empêche une analyse dynamique. Dans ce cas, seul le désassemblage peut permettre l'analyse du malware. Il est aussi possible de trouver d'éventuelles routines d'obfuscations employées afin de masquer les données sensibles du code malveillant (mot de passe, clés de chiffrement, adresse des serveurs, etc.)

Cependant le désassemblage a quelques inconvénients :

- **Il nécessite de grandes compétences** afin de pouvoir lire et d'analyser du code assembleur. Il faut connaître les correspondances systèmes entre le code, les langages de haut niveau et les appels systèmes
- **L'analyse complète d'un fichier peut représenter beaucoup de temps** (de plusieurs jours à plusieurs semaines), de compétences et d'énergie selon la taille et la complexité du code
- **Certains malwares sont protégés contre le désassemblage** à l'aide de techniques de compression et de chiffrement. Ces protections, jamais parfaites, peuvent néanmoins ralentir l'analyse et demander beaucoup d'efforts pour être cassées.

5.5.1.3 Les outils et les services d'analyse statique

- **La commande file** : file est une commande d'UNIX qui permet de déterminer le type d'un fichier.

Pour chaque fichier valide passé en paramètre, file tente de déterminer le type de données qu'il contient et affiche cette information et éventuellement d'autres informations comme

les dimensions pour une image ou les codecs.

La figure 5.4 donne un exemple d'utilisation de la commande file :

```
root@Mohamed: ~/Desktop
File Edit View Search Terminal Help
root@Mohamed:~# cd Desktop/
root@Mohamed:~/Desktop# file international_petroleum_conference_2014
international_petroleum_conference_2014: PDF document, version 1.7
root@Mohamed:~/Desktop#
```

FIGURE 5.4 – Utilisation de la commande file.

- L'outil **md5sum** : est un utilitaire en ligne de commande qui permet de vérifier l'intégrité d'un fichier. En effet, il permet de récupérer et comparer des empreintes MD5 des fichiers.
- La figure 5.5 donne un exemple d'utilisation de l'outil md5sum :

```
root@Mohamed: ~/Desktop
File Edit View Search Terminal Help
root@Mohamed:~# cd Desktop/
root@Mohamed:~/Desktop# md5sum international_petroleum_conference_2014
e05f7af85ed3d8618a93aa54fdf87d5c  international_petroleum_conference_2014
root@Mohamed:~/Desktop#
```

FIGURE 5.5 – L'outil md5sum.

- **Antivirus scaninig** :Après la récupération de hash md5 de malware qui va nous permettre de savoir si ce malware a été déjà analysé par une autre personne. Pour ce faire, on peut trouver des sites qui permettent d'accomplir cette tâche. Il suffit d'entrer le fichier et il va vérifier s'il s'agit d'un malware connu ou non. On peut citer comme exemple le site suivant www.virustotal.com qui permet d'analyser le fichier avec 52 antivirus différents. Cette étape est très importante dans le but de ne pas perdre du temps dans l'analyse d'un malware déjà connu.

La figure montre la page web VirusTotal.com :



FIGURE 5.6 – Aperçu sur l'interface du site www.virustotal.com.

- **Strings de Sysinternals suite** : Recherche pour ANSI et UNICODE chaînes dans les fichiers.

La figure 5.7 montre un exemple d'utilisation de l'outil strings :

```
C:\Windows\system32\cmd.exe
C:\Users\Admin\Desktop>strings malware.exe
Strings v2.51
Copyright <C> 1999-2013 Mark Russinovich
Sysinternals - www.sysinternals.com

MZP
InnoeR
This program must be run under Win32
^B*
CODE
'DATA
BSS
idata
.tls
.rdata
P.reloc
P.rsrc
string
$>@
Free
@)>
InitInstance
L)e
CleanupInstance
h<@_
ClassType
```

FIGURE 5.7 – L'outil strings de Sysinternals suite.

- **PEiD** : Il détecte les packers et les compilateurs les plus courants pour les fichiers PE. Il est actuellement possible de détecter plus de 470 signatures différentes dans les fichiers portable exécutable.

La figure 5.8 montre le programme PEiD.

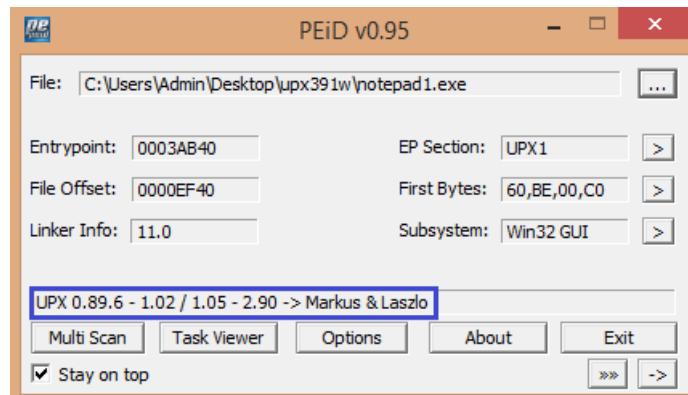


FIGURE 5.8 – L'interface de PEiD.

- **Les désassemblateurs** : Parmi les outils les plus connus pour le désassemblage, il y a bien sûr **IDA**. IDA est un désassembleur commercial très utilisé en rétro-ingénierie. Il supporte une grande variété de formats exécutables pour différents processeurs et systèmes d'exploitation.

IDA permet de passer du code binaire du malware vers son code assembleur. De plus, on peut lui ajouter le plugin Hex-Ray qui va nous permettre de faire la décompilation (c'est-à-dire revenir au code C).

La figure 5.9 montre le programme IDA :

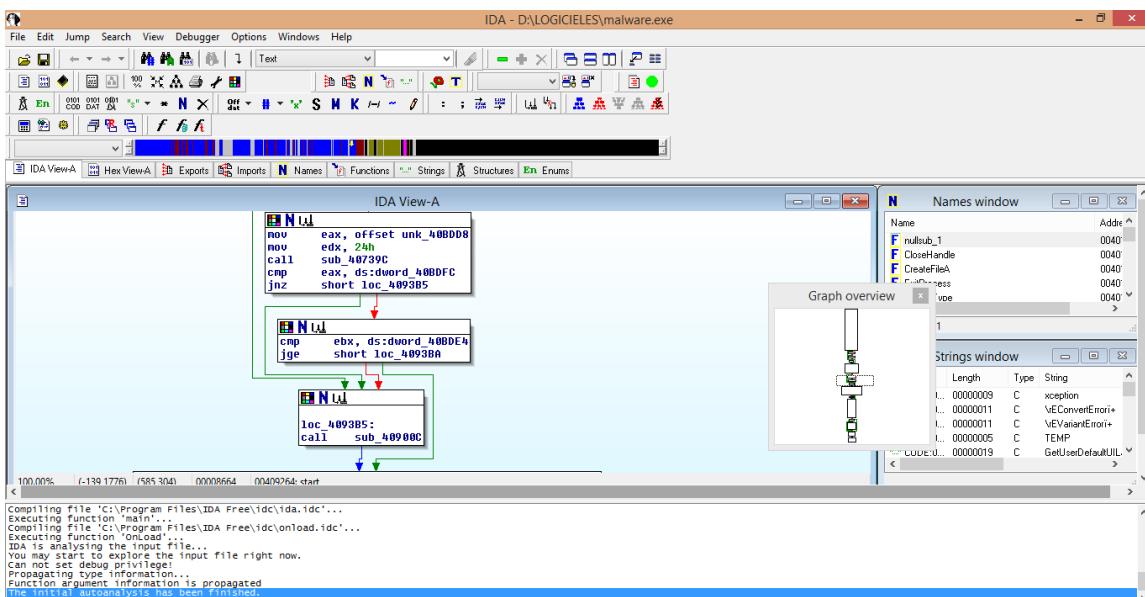


FIGURE 5.9 – Interface d'IDA Pro Free.

5.5.2 L'analyse dynamique

5.5.2.1 Définition

Contrairement à l'analyse statiques, l'analyse dynamique consiste à exécuter réellement le code malveillant dans un environnement adéquat afin d'observer toutes ses actions et d'en déduire son comportement global. Ce type de méthode est parfois appelée analyse comportementale.

Cette méthode apporte plusieurs avantages par rapport au désassemblage statique :

- Elle est beaucoup plus rapide. En quelques minutes, il est déjà possible d'avoir un aperçu des principales fonctionnalités du malware
- Elle ne nécessite pas de connaissances en assembleur ou en programmation.
- Elle n'est pas sensible aux techniques de protection des malwares (obfuscation, anti-désassemblage, etc).

Par contre, l'analyse comportementale n'est pas efficace dans certains cas :

- Si certaines fonctionnalités du malware nécessitent des conditions particulières pour être exécutées, elles peuvent ne pas être détectées lors de l'analyse
- Certains malwares contiennent des fonctionnalités permettant de détecter qu'ils sont dans un environnement contrôlé et de ne pas s'exécuter dans ces cas là
- L'analyse comportementale ne permet pas toujours d'accéder aux informations sensibles contenues dans le malware.

5.5.2.2 Les outils d'analyse dynamique

- **Process Monitor et Capture BAT** : Permettent de surveiller l'activité du système de fichiers, du Registre, des processus, des thread et des DLL en temps réel.. Ces outils peuvent aider à comprendre comment les tentatives de logiciels malveillants à intégrer dans le système lors de l'infection.

La figure 5.10 montre l'interface de programme Process Monitor :

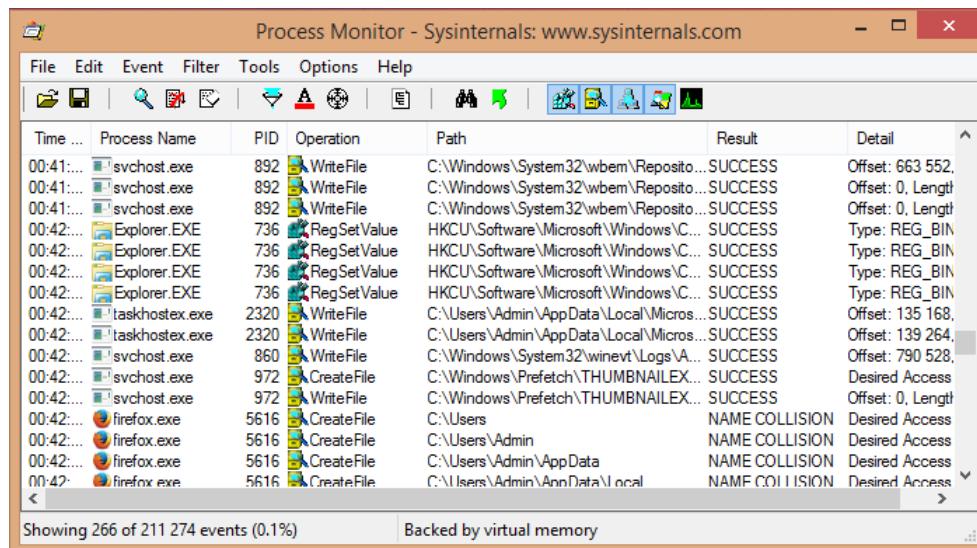


FIGURE 5.10 – L’outil Process Monitor de Sysinternals suite.

- Process Explorer et Process Hacker :** Ils indiquent quels fichiers ont été ouverts par les clés de registre et autres processus d’objet, quelles DLL ils ont chargées, le propriétaire de chaque processus, et bien plus encore.

La figure 5.11 montre l’utilitaire Process Explorer :

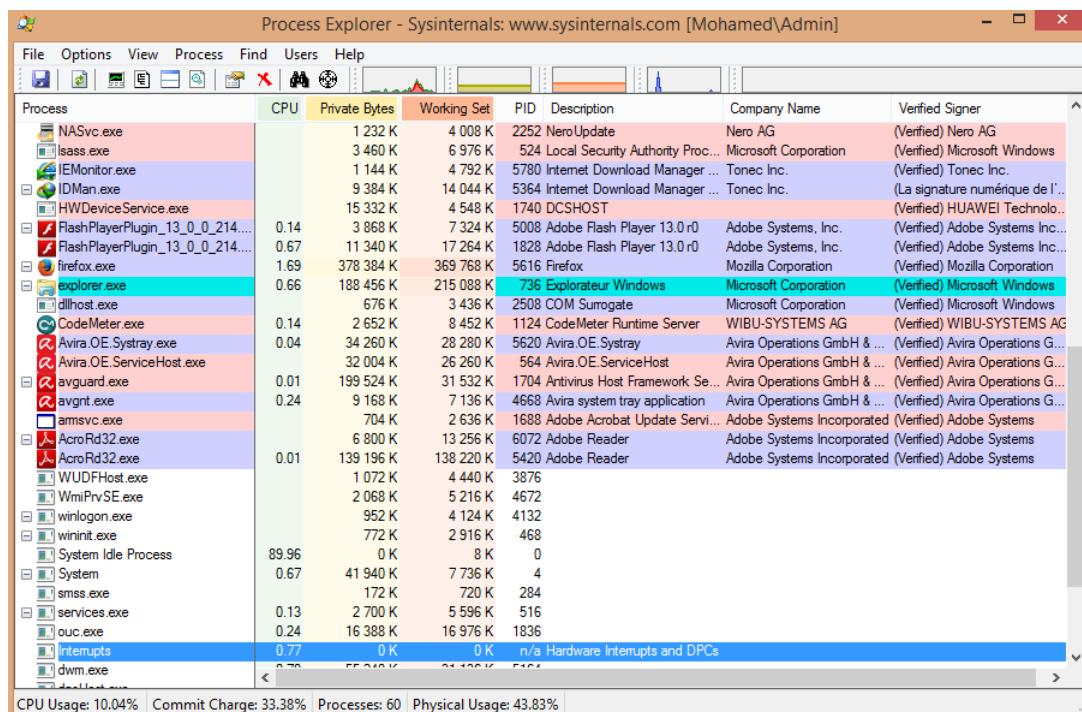


FIGURE 5.11 – L’outil Process Explorer de Sysinternals suite.

- Wireshark et SmartSniff :** Ils sont des analyseurs réseau, qui peuvent capturer le trafic réseau de laboratoire pour les tentatives de communication malveillants, tels que les demandes de résolution de DNS, le trafic de bot, ou des téléchargements.

La figure 5.12 montre l’interface de Wireshark :

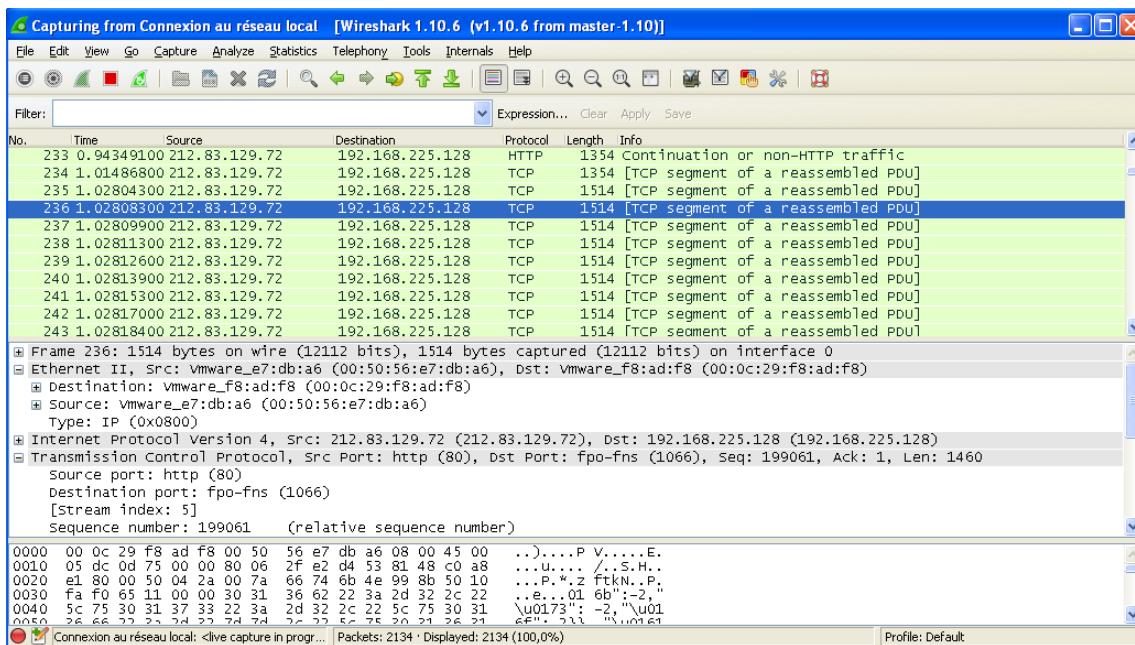


FIGURE 5.12 – L'interface de Wireshark.

- **Regshot** : Est est un outil léger pour comparer l'état du système avant et après l'infection, de mettre en évidence les principaux changements apportées par le malware au système de fichiers et les Registres.

La figure montre le programme Regshot :

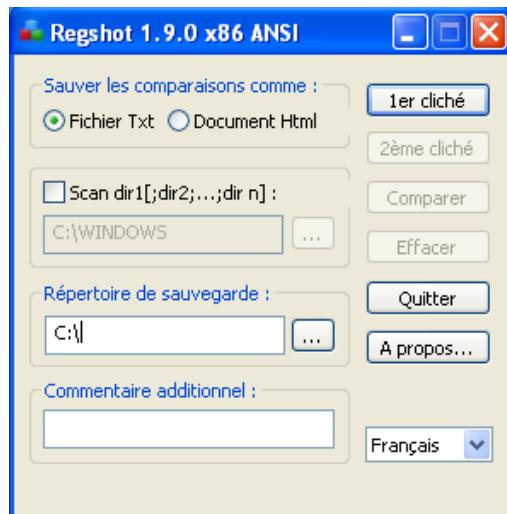


FIGURE 5.13 – L'interface de Regshot.

- **Cuckoo Sandbox** : Ce bac à sable (sandbox) permet en quelques minutes d'obtenir une première estimation des capacités d'un malware et de ses communications avec l'extérieur ou encore de connaître les fichiers créés sur le système. Gratuit, il nécessite d'être installé sur une station hôte saine et de lui soumettre des malwares qui seront analysés de façon totalement automatisée dans une machine virtuelle.

La figure 5.14 montre l'interface Cuckoo Sandbox :

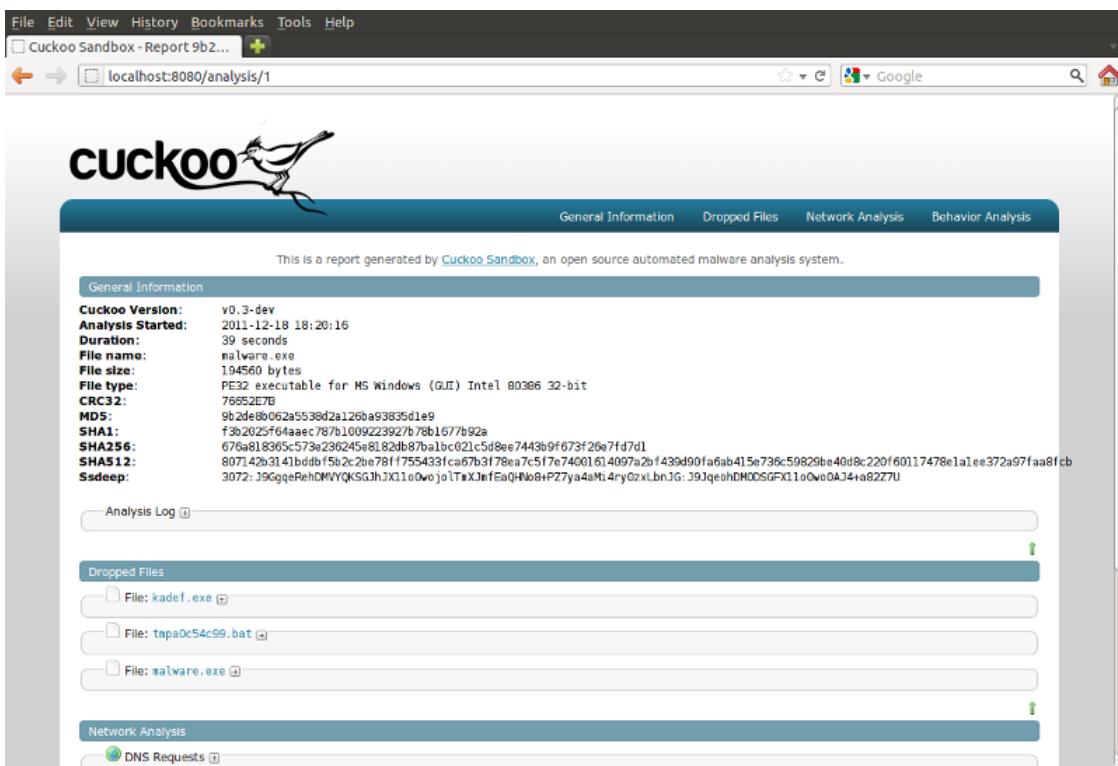


FIGURE 5.14 – L’interface de Cuckoo Sandbox.

Par rapport à d’autres services en ligne de ce type, Cuckoo permet d’analyser très rapidement en première approche des binaires ou documents suspects dans un environnement sécurisé tout en gardant la confidentialité des données ce que ne permettent pas les services de ligne.

5.5.3 L’analyse des documents malveillants

Les documents malicieux se sont énormément développés ces dernières années et constituent un vecteur d’attaque très utilisé par les malwares. Nous nous focaliserons ici sur les documents de la famille Microsoft Office et les documents PDF, ceux-ci pouvant se révéler très complexes. Du point de vue de l’analyste, la problématique qui se pose est de savoir si le document à analyser possède une charge malicieuse. La génération de documents malicieux est très simple et nombreux sont les outils d’intrusion qui possède de telles fonctionnalités.

5.5.3.1 L’analyse des documents PDF malveillants

Depuis l’année 2009, les attaques au moyen de fichiers PDF (Portable Document Format) malveillants se sont multipliées. Par exemple :

- Pour l’année 2009 le Cert-IST a émis 4 Dangers Potentiels pour avertir notre communauté de nouvelles attaques utilisant des vulnérabilités dans Adobe Reader ou dans Adobe Acrobat
- L’éditeur antivirus BitDefender a placé en tête de son ”Top 10 pour décembre 2009” la menace ”Exploit.PDF-JS.Gen” qui représente 12,04% de l’ensemble des infections. Sous ce nom sont regroupés des fichiers PDF qui exploitent différentes vulnérabilités détectées dans le moteur JavaScript de PDF Reader, afin d’exécuter du code malveillant sur l’ordinateur de l’utilisateur [45].

Les outils d'analyse des documents PDF malveillants

- **pdfid** : identifie les fichiers PDF qui contiennent des chaînes associées à son exécution des scripts et des actions
- **pdf-parser** : examine la structure de fichiers PDF
- **PDF Stream Dumper** : combine de nombreux outils d'analyse de fichiers PDF sous une interface graphique
- **peepdf** : offre un shell interactif en ligne de commande pour examiner les fichiers PDF.

5.6 Partie pratique

5.6.1 Présentation du malware

Le PDG d'une entreprise pétrolière a reçu un mail contenant un document PDF en rapport avec une conférence internationale.

Une fois le PDF ouvert, il s'est rendu compte que le fichier ne correspondait pas à la conférence et que c'était seulement un brouillon de document. Il a alors décidé de l'envoyer à son RSSI (Responsable de Sécurité des Systèmes Informatiques) pour vérifier si le document n'est pas malicieux.

5.6.2 L'analyse statique

Étape 1 : Type de fichier

On ouvrant le document suspect avec un éditeur hexadécimal, on voit dans la figure 5.15 qu'il est bien un document PDF version 1.7.

Offset (h)	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000	25 50 44 46 2D 31 2E 37 OD 25 E2 E3 CF D3 OD OA PDF-1.7.%äö..
00000010	31 38 20 30 20 6F 62 6A 20 3C 3C 2F 4C 69 6E 65 18 0 obj <</Line

FIGURE 5.15 – Vérification de type de fichier dans l'éditeur HxD.

Étape 2 : Récupérer le hash MD5 de fichier

On utilisant l'application "WinMD5" pour récupérer le hash MD5 de document PDF, la figure 5.16 montre le hash du fichier.

File Name and Size: C:\Documents and Settings\MOHAMED\Bureau\pdf_suspect.pdf
Current file MD5 checksum value:
75c94619c4354d0430f0f23bdc3a045a

FIGURE 5.16 – Le hash MD5 de fichier avec "WinMD5".

Étape 3 : Antivirus scanning

Après avoir récupérer le hash MD5 de document PDF, qui va nous permettre de savoir si ce malware a été déjà analysé par une autre personne. On utilisant "virustotal", on voit dans la figure que le document PDF n'a pas été analysé auparavant.

Fichier non trouvé

Le fichier que vous cherchez n'est pas dans notre base de données.

Ramenez-moi à la page principale

Essayer une autre recherche

FIGURE 5.17 – Le scan avec VirusTotal.

Étape 4 : pdfid et pdf-parser

Pdfid identifie les fichiers PDF qui contiennent des chaînes associées à son exécution des scripts et des actions, la figure 5.18 montre que ce document PDF contient un "javascript" et "Automatic Action" donc on peut dire que le document est malveillant. Ce document PDF contient un contenu Flash. Très souvent, les attaquants utilisent Flash intégré afin d'essayer d'exploiter le flash Adobe et exécuter du code arbitraire ou de provoquer un déni de service lors de l'ouverture d'un document PDF malveillant.

```
root@Mohamed: ~/Desktop
PDFiD 0.0.12 international_petroleum_conference_2014
PDF Header: %PDF-1.7
obj 41
endobj 41
stream 6
endstream 6
xref 2
trailer 2
startxref 2
/Page 3
/Encrypt 0
/ObjStm 0
/JS 1
/JavaScript 1
/AA 1
/OpenAction 0
/AcroForm 2
/JBIG2Decode 0
/RichMedia 2
/Launch 0
/EmbeddedFile 2
/Colors > 2^24 0
```

FIGURE 5.18 – L'interface de PDFiD.

Avec pdf-parser, on voit dans la figure 5.19 le code javascript :

```
root@Mohamed: ~/Desktop
File Edit View Search Terminal Help
root@Mohamed:~/Desktop# pdf-parser -s javascript international_petroleum_conference_2014
obj 26 0
Type:
Referencing:

<<
/S /JavaScript
/JS '(var page=1;\\"r\\nvar pdfver=app.viewerVersion;\\"r\\nvar pdftype=app.viewerType;\\"r\\n\\r\\nif\\((pdfver<"10"\\"))\\r\\n{\\"r\\n    page=3;\\"r\\n    if\\((pdftype=="Reader"\\"))\\r\\n        {\\"r\\n            page=1;\\"r\\n        }\\r\\n        if\\((pdfver>"9"\\"))\\r\\n            {\\"r\\n                page=0;\\"r\\n            }\\r\\n        }\\r\\n    }\\r\\n    if\\((pageNum==page;)\")\\r\\n        this.pageNum=page;)'>>

root@Mohamed:~/Desktop#
```

FIGURE 5.19 – L'interface de pdf-parser.

La figure 5.20 montre un exemple d'un fichier intégré dans le document PDF.

```
root@Mohamed:~/Desktop# pdf-parser -o 14 international_petrolium_conference_2014
obj 14 0
Type: /EmbeddedFile
Referencing:
Contains stream

<<
/Length 8216
/Type /EmbeddedFile
>>
```



FIGURE 5.20 – Exemple d'un fichier intégré.

5.6.3 L'analyse dynamique

Dans une analyse dynamique nous devons vérifier le fonctionnement global et le fonctionnement interne du code actuellement en l'exécutant dans un environnement contrôlé. Ceci nous aide à éliminer les faux positifs de la phase d'analyse statique. Des auteurs de malwares incluent du code (techniques d'obfuscation) pour détecter qu'il fonctionne dans les confins d'une machine virtuelle afin de changer son chemin d'exécution.

Pour cela on a mis en place un environnements de test s'exécutant sous MS Windows XP Professional SP3 dans une VirtualBox.

Ce document PDF a des techniques "anti-vm", pour les contourner il faut :

- Arrêter le processus "VBoxService.exe"
- Arrêter le processus "VBoxTray.exe"
- Augmenter la taille de la RAM plus que 512M.

Étape 1 : clés de Registres

On prend un cliché sur l'état de registres avec l'outil "RegShot".

Étape 2 : préparation des outils

- **système de fichiers** : définir un filtre pour l'outil "Procmon", on met "category is write"
- **Surveiller les processus** : on lance "processus Explorer"
- **Surveiller le trafic réseau** : on lance "Wireshark" et "Fakenet".

Étape 3 : ouverture du Document PDF

double clic sur le document pdf puis on fait :

- On prend un 2eme cliché sur l'état de registres
- Arreter les captures dans "Process Monitor" et "Wireshark".

Etape 4 : Interprétation des résultats

Les clés de registre

On fait la comparaison entre les deux clichés de "Regshot" on voit dans la figure 5.21 qu'il y'a un changement de 146 clés de registre.

```

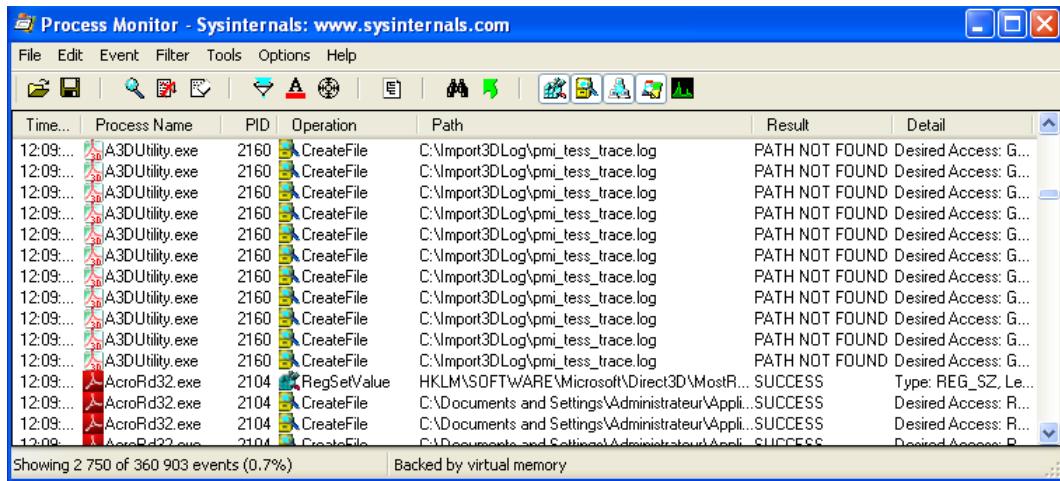
HKU\S-1-5-21-796845957-1958367476-1801674531-500\Software\Microsoft\Windows\CurrentVersion\Explorer
HKU\S-1-5-21-796845957-1958367476-1801674531-500\Software\Microsoft\Windows\CurrentVersion\Explorer
HKU\S-1-5-21-796845957-1958367476-1801674531-500\Software\Microsoft\Windows\CurrentVersion\Internet
HKU\S-1-5-21-796845957-1958367476-1801674531-500\SessionInformation\ProgramCount: 0x00000006
HKU\S-1-5-21-796845957-1958367476-1801674531-500\SessionInformation\ProgramCount: 0x00000007

-----
Total changes: 146
-----
```

FIGURE 5.21 – Exemple d'un fichier créé par le document PDF.

Système de fichiers

La figure 5.22 montre que le malware a créé des centaines de fichiers dans le système :

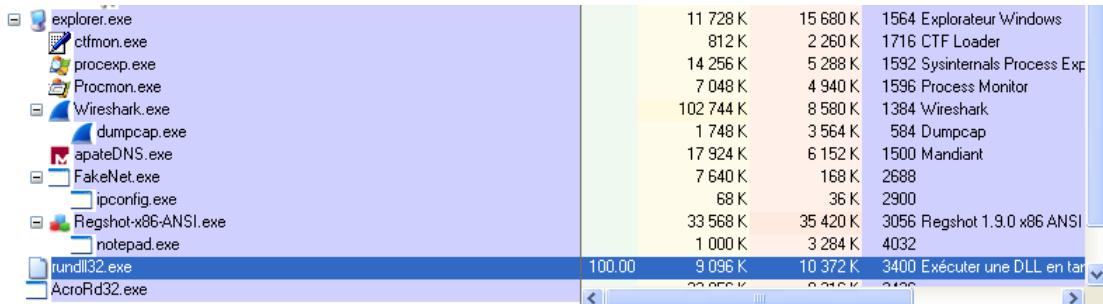


Time...	Process Name	PID	Operation	Path	Result	Detail
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	A3DUtility.exe	2160	CreateFile	C:\Import3DLog\pmi_tess_trace.log	PATH NOT FOUND	Desired Access: G...
12:09...	AcroRd32.exe	2104	RegSetValue	HKEY_LOCAL_MACHINE\Software\Microsoft\Direct3D\MostR... SUCCESS	REG_SZ, Le...	Type: REG_SZ, Le...
12:09...	AcroRd32.exe	2104	CreateFile	C:\Documents and Settings\Administrateur\Appli...SUCCESS	Desired Access: R...	Desired Access: R...
12:09...	AcroRd32.exe	2104	CreateFile	C:\Documents and Settings\Administrateur\Appli...SUCCESS	Desired Access: R...	Desired Access: R...
12:09...	AcroRd32.exe	2104	CreateFile	C:\Documents and Settings\Administrateur\Appli...SUCCESS	Desired Access: R...	Desired Access: R...

FIGURE 5.22 – L'interface Processus Monitor.

Processus suspects

La figure 5.23 montre un processus suspect qui a été lancé après l'ouverture du document PDF :



explorer.exe	11 728 K	15 680 K	1564 Explorateur Windows
ctfmon.exe	812 K	2 260 K	1716 CTF Loader
procexp.exe	14 256 K	5 288 K	1592 Sysinternals Process Exp
Procmon.exe	7 048 K	4 940 K	1596 Process Monitor
Wireshark.exe	102 744 K	8 580 K	1384 Wireshark
dumpcap.exe	1 748 K	3 564 K	584 Dumpcap
apateDNS.exe	17 924 K	6 152 K	1500 Mandiant
FakeNet.exe	7 640 K	168 K	2688
ipconfig.exe	68 K	36 K	2900
Regshot-x86-ANSI.exe	33 568 K	35 420 K	3056 Regshot 1.9.0 x86 ANSI
notepad.exe	1 000 K	3 284 K	4032
rundll32.exe	100.00	9 096 K	3400 Exécuter une DLL en tar
AcroRd32.exe	22 056 K	2 016 K	2 122

FIGURE 5.23 – Processus suspect.

Après voir les propriétés de ce processus, on voit qu'il a créé un virus "sysinit.ocx" (figure 5.24)

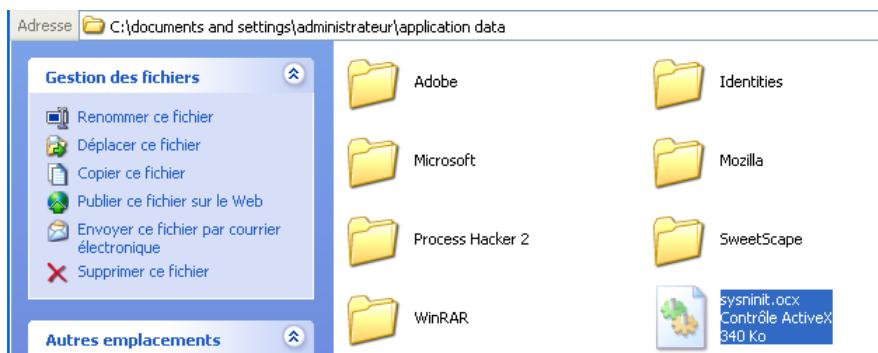


FIGURE 5.24 – Exemple d'un fichier crée par le document PDF.

La figure donne un exemple des fichiers qui se lancent au démarrage de Microsoft Windows :

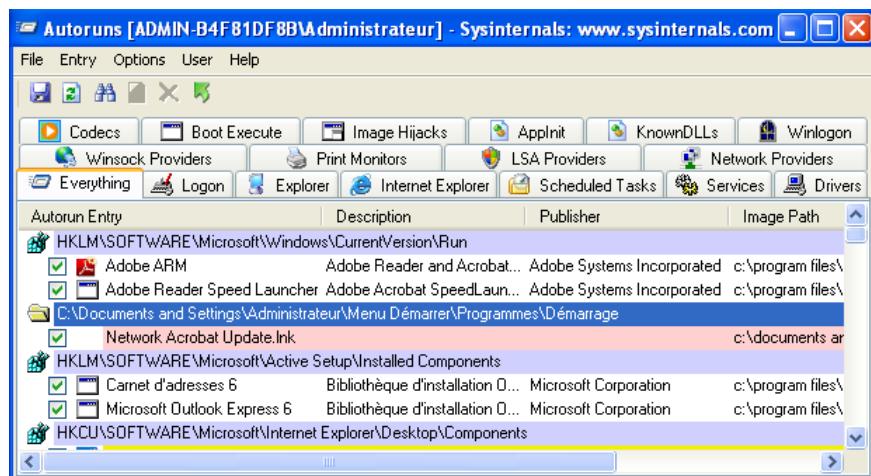


FIGURE 5.25 – L'interface d'Autoruns.

Trafic réseau

Dans la figure on remarque que le malware essaye de contacter le domaine "jhj.wv4.org" et pour tromper la victime que la machine fonctionne bien, il essaye de télécharger les mises à jour de Microsoft Windows.

```
Raccourci vers FakeNet
[New request on port 80.1
GET /test2/serverok.html HTTP/1.1
Accept: image/gif, image/jpeg, image/pjpeg, image/pjpeg, */*
User-Agent: Mozilla/4.0 <compatible; MSIE 8.0; Windows NT 5.2; Trident/4.0; .NET CLR 1.1.4322>
Host: www.jhj.wv4.org
Connection: Keep-Alive
Cache-Control: no-cache

[Sent http response to client.]


[Received new connection on port: 443.1

[Received new connection on port: 80.1
[New request on port 80.1
GET /msdownload/update/v3/static/trustedr/en/authrootseq.txt HTTP/1.1
Accept: */
User-Agent: Microsoft-CryptoAPI/5.131.2600.5512
Host: www.download.windowsupdate.com
Connection: Keep-Alive
Cache-Control: no-cache
Pragma: no-cache]
```

FIGURE 5.26 – Le trafic réseau par "fakenet".

La figure 5.27 montre que le C&C server est "jhj.wv4.org" et il essaye de tester si la victime a un accès Internet ou non avec le serveur "google.com".

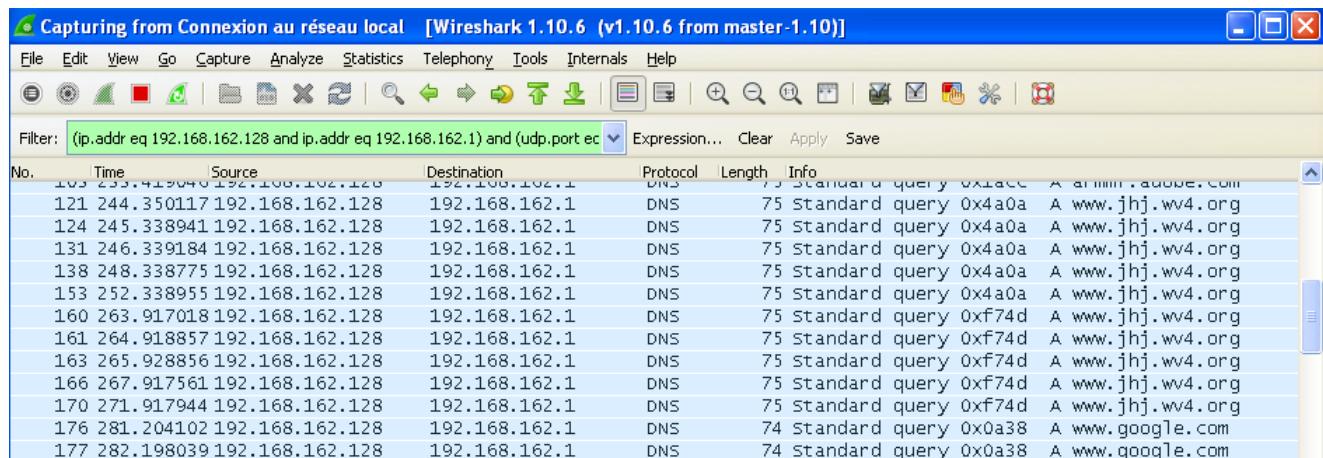


FIGURE 5.27 – Le trafic réseau par "Wireshark".

5.6.4 Protection contre ce type d'attaques

Pour se protéger contre ce type d'attaques, il faut mettre à jour Adobe Reader. Et pour la désinfection, il faut supprimer tous les fichiers qui ont été créés par le malware ou d'écrire un script ".bat" qui permet de supprimer les fichiers créés.

5.7 Conclusion

Les malwares sont devenus une réalité pour la majorité des utilisateurs et bien peu d'entre eux y échappent (en particulier sous les systèmes Windows & Android). Connaître leur mode de fonctionnement ainsi que méthodes de propagation restent le meilleur moyen de s'en protéger. La protection contre les malwares se doit d'être globale et ne peut uniquement se contenter de solutions techniques.

Concernant leur analyse, il est possible par l'analyse dynamique d'avoir en quelques minutes les principales caractéristiques d'un malware ainsi que ses fonctionnalités et ce, sans nécessiter de solides connaissances en assembleur. Ce n'est malheureusement pas possible dans certains cas, où les malwares sont capables de détecter les environnements virtuels ou surveillés, et dans ces cas-là, seule une analyse par désassemblage peut permettre d'évaluer la menace.

Conclusion générale et perspectives

Sont cachés dans des e-mails, derrière des liens ou des bannières, dissimulés dans des fichiers et des programmes téléchargés le plus légalement du monde . . . , les malwares, un nom générique qui désigne n'importe quelle forme de code malveillant, qu'il s'agisse d'un virus, un cheval de Troie (trojan), un keylogger, un spyware, un rootkit, etc.

Parmi les outils de protection contre les malwares, on trouve les antivirus. Les antivirus sont des logiciels conçus pour identifier, neutraliser et éliminer des logiciels malveillants. Ces derniers peuvent se baser sur l'exploitation de vulnérabilités des logiciels, mais il peut également s'agir de logiciels modifiant ou supprimant des fichiers, que ce soit des documents de l'utilisateur stockés sur l'ordinateur infecté, ou des fichiers nécessaires au bon fonctionnement de l'ordinateur.

L'analyse de malwares nous permet de bien comprendre le fonctionnement des malwares et pour générer des signatures efficaces afin d'éliminer les souches de malwares .

C'est pour cela que dans notre projet on a développé un antivirus qui sera capable de détecter les souches de malwares par la technique de détection par signature, qui est la technique la plus utilisée actuellement par les antivirus, ainsi qu'un parseur de PE qui nous permet d'observer le contenu d'un fichier exécutable. Avec ce parseur, nous pouvons visualiser et examiner les fichiers de format PE ainsi que leurs structures internes.

Perspectives

Parmi les perspectives qui restent à explorer, nous pouvons citer :

- Concevoir d'autres techniques de détection telle que la technique comportementale
- Détection des malwares en temps réel
- Contrôle Parental
- Ajouter à l'antivirus une fonctionnalité faisant objet d'un pare-feu
- Permettre le chiffrement de fichiers
- Et enfin, faire analyser la conception et le code source du projet par des experts en audit de sécurité, afin de corriger les éventuelles failles de sécurité qui nous ont échappées. Puis, effectuer des tests de pénétration et valider la sûreté de l'application.

Bibliographie

- [1] M. Hoffmann, “Sécurité informatique,” 2008. <http://www.dicodunet.com/definitions/internet/securite-informatique.htm>.
- [2] L. Rogers, “les vulnérabilités,” 2003. <http://usinfo.state.gov/journals/itgic/1103/ijgf/gj7f.htm>.
- [3] M. D. Abrams, “Nims information security threat methodology,” 1998. http://www.mitre.org/work/tech_papers/tech_papers_98/nims_information/nims_info.pdf.
- [4] F. Cohen, *Computer Viruses*. PhD thesis, University of Southern California, 1986.
- [5] L. Adleman, *An abstract theory of computer viruses*, vol. 403, ch. Advances in Cryptology, pp. 354–374. 1988.
- [6] Éric Filiol, *Les virus informatiques :théorie, pratique et applications*. deuxième ed., 2009.
- [7] V. Diego, “Les virus informatiques,” 2004. <http://tecfa.unige.ch/staf/staf-j/diego/staf14/ex8/virus.html>.
- [8] S. S. R. C. Nicholas Weaver, Vern Paxson, “A taxonomy of computer worms,” 2003.
- [9] Orange, “Virus, ver, cheval de troie... quelle est la différence ?,” 2006. <http://www.orange.fr/bin/frame.cgi?u=http%3A//assistance.orange.fr/737.php%3Fdub%3D2%26>.
- [10] S. WACKER, “Les logiciels malveillants.” <http://www.montpellier.iufm.fr/technoprimaire/>.
- [11] wikipedia, “Rootkit,” 2013. <http://fr.wikipedia.org/wiki/Rootkit>.
- [12] wikipedia, “Ransomware,” 2014. <http://fr.wikipedia.org/wiki/Ransomware>.
- [13] cisco, “Cisco 2014 annual security report,” 2014. <http://www.cisco.com/web/offers/lp/2014-annual-security-report/index.html?.keycode=000350063>.
- [14] Alliacom, “Retour sur windigo, le malware qui affecte les systèmes linux et unix,” 2014. <http://www.alliacom.com/nous-suivre/blog/item/retour-sur-windigo-le-malware-qui-affecte-les-systemes-linux-et-unix>.
- [15] M. R. Guillaume CHARPENTIER, Olivier MONTIGNY, “Virus / antivirus,” 2004.
- [16] S. FONTAINE, “Les virus,” 2007. <http://www.authsecu.com/virus-vers-chevaux-de-troie-hoax/virus-vers-chevaux-de-troie-hoax.php>.
- [17] wikipedia, “Portable exécutable,” 2013. http://fr.wikipedia.org/wiki/Portable_Executable.
- [18] Microsoft, “Microsoft pe and coff specification,” 2013. <http://www.microsoft.com/whdc/system/platform/firmware/PECOFF.mspx>.
- [19] I. Tutorial, “Explication complète sur le format pe,” Février 2005. <ftp://ftp-developpez.com/olance/articles/windows/pe-iczelion/luevlsmeyer.zip>.
- [20] M. Sikorski and A. Honig, *Practical Malwar Analysis, The Hands-On Guide to Dissecting Malicious Software*. William Pollock, 2012.

- [21] S. Internet, “Comment marche un antivirus,” 2014. <http://eservice.free.fr/antivirus.html>.
- [22] wikipedia, “Fonction de hashage,” 2014. http://fr.wikipedia.org/wiki/Fonction_de_hachage.
- [23] Commentcamarche, “Signature électronique,” 2014. <http://www.commentcamarche.net/contents/212-signature-electronique>.
- [24] wikibooks, “Les fonctions de hachage cryptographiques,” 2013. http://fr.wikibooks.org/wiki/Les_fonctions_de_hachage_cryptographiques.
- [25] Viruslist.com, “Mise à jour des bases antivirus,” 2014. <http://www.viruslist.com/fr/glossary?glossid=163219184>.
- [26] “Jimdo,” 2014. <http://fr.wikipedia.org/wiki/Jimdo>.
- [27] “Github,” 2014. <http://fr.wikipedia.org/wiki/GitHub>.
- [28] KernelMode. <http://www.kernelmode.info/forum/index.php>.
- [29] Malware.lu. <http://www.malware.lu/pages/company.html>.
- [30] E. Contagio. <http://contagioexchange.blogspot.fr/>.
- [31] CrowdRE. <https://crowdre.crowdstrike.com/>.
- [32] V. S. malwares. <http://www.salesmalwares.com/index.php>.
- [33] S. Malekal’s. <http://www.malekal.com/>.
- [34] M. Analysis. <http://www.internetcopol.fr/wup/?rss>.
- [35] M. FireEye. <http://blog.fireeye.com/research/>.
- [36] Evil3ad. <http://www.evild3ad.com/>.
- [37] M86SecurityLabs. <http://labs.m86security.com/>.
- [38] malwares Fun. <http://fumalwareanalysis.blogspot.fr/>.
- [39] B. Kaspersky. <https://www.securelist.com/en/>.
- [40] B. Eset. <http://blog.eset.com/>.
- [41] B. Norton. <http://www.symantec.com/connect/>.
- [42] B. Sophos. <http://nakedsecurity.sophos.com/>.
- [43] VMware. <http://www.vmware.com/fr/>.
- [44] VirtualBox. <https://www.virtualbox.org/>.
- [45] CERT-IST. http://www.cert-ist.com/public/fr/S0_detail?code=malicious_pdf.