RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

---

UNIVERSITE ECHAHID HAMMA LAKHDAR - EL OUED
FACULTÉ DES SCIENCES EXACTES
Département D'Informatique

Mémoire de Fin D'étude
Présenté pour l'obtention du Diplôme de

LICENCE ACADEMIQUE

Domaine : **Mathématique et Informatique**
Filière : **Informatique**

Spécialité : **Systèmes Informatiques**

Thème

Conception et Réalisation D'une Technique Puissante De Protection du Produit Logiciel

Présenté par :

- Brahim Atia
- Ahmed Boukmih

Proposé et Encadré par :

M. A. Boucherit

Soutenue le 20-05- 2018 Devant le jury :

M. A.K. Laouid	Président
M. K. Gherbi	Rapporteur

Année Universitaire : 2017-2018

Remerciements

Avant tous, je remercie dieu le tout puissant de m'avoir donné le courage

et la patience pour réaliser ce travail malgré toutes les difficultés rencontrées.

Je remercie infiniment tous qui nous a aidé de près et de loin d'avoir compléter

ce travail et dépasser tous les obstacles surtout notre encadreur

*" **Ammar Boucherit** "*

qui n'a pas cessé de nous donner les conseils et les bonnes orientations et nous prive pas de son temps.

*A mes amis **Mounir Beggas** et **Othmani Samir** qui ont proposé de nous aider et nous encourage pour continuer et de ne pas abandonner.*

A nous exceptionnelles collèges du promo Barika Bachir. ,Chathouna Harouna et Fortas Ikrame de leur soutient et encouragement tout au long de cette année.

Et en fin tous qui nous ont aidés de près et de loin pour

La réalisation de notre mémoire.

Dédicaces

Je dédie ce modeste travail

À ma chère mère où elle repose en paix. Après nous avoir appris à vivre dans la dignité et la valeur de l'homme dans sa capacité à donner.

A celui qui a fait des grands efforts pour mon bonheur mon Père.

A ma femme, qui m'a soutenu et encouragé et m'a montré les conséquences de l'étude sans ennui et infatigable.

À mes petits Soulaymane et Maymouna Oum Elkhir qui ont attendus cette moment avec impassion .

A tous la promotion 1996 ainsi 2018 informatique

A. Ibrahim

Dédicaces

Toutes les lettres ne sauraient trouver les mots qu'il faut…
Tous les mots ne sauraient exprimer la gratitude, l'amour,
Le respect, la reconnaissance…

Je dédie ce modeste travail

A MON TRES CHER PERE RABAH…

A MA TRES CHERE MERE MESSAOUDA…

A TOUS MA GRAND FAMILLE…

A MA PEUTITE FAMILLE : MA CHÈRE FEMME ***NORA***

ET MA CHERE PETITE FILLE ***MANAR…***

A TOUS MES AMIS ET MES COLLÈGUES, SURTOUT

MON COLLÈGUE DANS CE TRAVAIL ***BRAHIM ATIA…***

À TOUS MES ENSEIGNANTS DU DÉPARTEMENT

INFORMATIQUE…

A NOTRE ENCADREUR LE DR. ***BOUCHERIT AMAR…***

ملخص

لقد أصبح تطوير البرمجيات من المجالات الحيوية ذات التأثير المباشر والمتنامي على الحياة اليومية، كما أنه يمثل عنصرًا استراتيجيًا لأي لبلد يريد ان ليكون له مكان في العالم.

كما أنه من الواضـح والثابت أن شـركات تطوير منتجات البرمجيات تخسـر أكثر من 35٪ من مبيعاتها بسبب القرصنة كل عام. لذلك، من المهم والضروري ايجاد طريقة جيدة للتغلب على هذه المشكلة.

الغرض من هذا المشروع هو اقتراح وتنفيذ تقنية غير تقليدية لحماية حقوق مطوري منتجات البرمجيات وضمان نسبة عالية من الحماية.

Résumé

Le développement des logiciels est un domaine indispensable à nos jours car elle impact la vie quotidienne d'une façon croissante, et elle devient un élément stratégique pour qu'un pays peut avoir une place dans le monde.

Il est aussi clair et *ennuyeux* que les sociétés de développement des produits logiciels perdent plus de 35% de ses ventes à cause du piratage chaque année. Par conséquent, il est aussi crucial de trouver un bon moyen pour pallier ce problème.

Le but de ce projet est de proposer et de réaliser une technique non classique pour assurer la protection des droits des développeurs des produits logiciels et de garantir un taux élevé de protection.

Sommaire

Liste des figures

Introduction générale :

C’est vrai que la protection juridique joue un rôle très important et s'intéresse largement à la protection des droits d'auteurs, mais elle n'est pas suffisante dans le monde de l'informatique pour des raison techniques et pratiques. Alors il vient le rôle de protection logiciel par des outils informatique car elles accompagnent les produits et garantissent le contrôle automatique et instantané.

Dans le contexte de notre projet de fin d'étude et afin de réaliser un outil de protection logiciel, nous allons en premier pas expliquer et donner les définitions des termes liés et les types de contrats relatifs à l'exploitation des programmes distribués dans différents types et domaines d'une part, afin de préciser la catégorie bien concernée par la protection et d’autre part, pour connaitre les différentes catégories de distribution. À la fin de cette étape, on a essayé de catégoriser les techniques de protection selon une option que nous considérais la plus importante d’un point de vue technique.

En deuxième étape, on a donné plus de définitions et explications concernent la protection physique par dongle et les raisons pour lesquelles elle a été choisi. Elle contient aussi la définition des besoins fonctionnelles et non fonctionnelles pour cette catégorie en générale et aussi pour notre outil a développé comme une première étape dans la voie de développement. On la terminant par une description littérale du principe de fonctionnement de notre dongle.

Le pas suivant ; c’était la description détaillée de la solution proposée. L’idée est d’utiliser un flash disque simple pour contenir des informations élémentaires et nécessaires pour le lancement du programme avec des caractéristiques bien précises de tel façon qu'il soit unique pour but d'interdire l'utilisation multiple de ce flash disque sur différentes machines.

Les informations qualifiées pour être utilisé comme des valeurs de référence sont groupées dans des ensembles chaque une reliée avec un objectif bien précise ; commençant par l’attachement total du fonctionnement du programme avec la présence de ce flash disque précisément. Ceci implique la présence des informations identifiant le dongle original pour que le programme en train du lancement puisse assurer qu'il s’agit du dongle légitime et approprié. En plus, tous ces données doivent être protégé contre la consultation et la modification intentionnellement ou par erreur.

La troisième étape c'était l'implémentation de la solution en code ; Microsoft Visual studio qui a été choisi comme environnement de développement. Pour plusieurs raisons parmi les quelles sa disponibilité ; sa compatibilité totale avec l'environnements ciblé Microsoft Windows. Aussi, par ce qu'il est l'outil le plus utilisé sur notre marche a fin d’assuré le maximum de possibilités d’aide d’’autres développeurs si c’est nécessaire.

La dernière étape, c'était la mise en place et le teste par l'intégration de notre outil avec un programme d'essai, et puis on a appliqué des différents scénarios de piratages imaginés pour dépasser la protection, les résultats obtenus nous ont montré l'efficacité très acceptable de notre technique surtout au niveau du marché local ciblé essentiellement.

Chapitre I

Protection

des logiciels

1. Introduction :

Pour quoi protégé son code ?

Le développement du code et la réalisation des produit informatique – les logiciels- est coûteuse du côté des efforts allouée ou bien du cout total.

Les revenues sont liées directement avec le taux de la disruption de ces logiciels d'une manière garantisse les revenues attendues. Mais cela est loin d'être réalisées avec la possibilité qu'elle soit distribuée et exploiter par une simple copie ou bien en partage les clés associées.

Protéger peut empêcher l'utilisation du code source de notre programme dans le but de tirer profit. Ou d'accéder au technologies mis en œuvre dans le logiciel. C'est aussi un bon moyen de s'assurer qu'un client ne fasse pas de mise à jour son autorisation. D'autre part cela permet d'empêcher des personnes males intentionnées de trouver un fail par une analyse de code source et de s'en server contre le programme.

Pour réaliser une protection il faut d'abord comprendre les termes et les aspects utilisées pour définir les éléments reliés directement avec cet objectif. Ensuite décrite et catégoriser les différentes techniques utilisées afin de Sitter ces avantages et inconvénients pour objectif de choisir le plus réponds avec nos exigences et capacités.

De notre part on a choisi ce thème car il présente l'option nécessaire pour chaque programme de profit en informatique et bien sûr le premier élément assurant la survivre et continuité des entreprises en conjoint avec le bon marketing. Aussi elle présente la porte à ouvrir pour découvrir et participer à un couloir très important en informatique moderne ; attacher à l'Ingénierie inverse et la Security d'information.

2. Définition d'un Logiciel :

C'est un ensemble des instructions forment un programme ou un ensemble de programme qui s'exécutent séquentiellement ou en parallèles et manipulent des données bien précises afin d'offrir un service ou réaliser un objectif. Ce composant et mener avec une documentation afin d'expliquer la manière d'utilisation et le principe de fonctionnement de ce logiciel.

Créer un logiciel est un travail intellectuel qui prend du temps. La création de logiciels est souvent le fait d'une équipe, qui suit une démarche logique et planifiée en vue d'obtenir un produit de bonne qualité dans les meilleurs délais. Le code source et le code objet des logiciels sont protégés par la convention de Berne concernant les œuvres littéraires[1].

Les logiciels sont créés et livrés à la demande d'un client ou pour atteindre une catégorie des clients ou alors ils sont créés sur l'initiative du producteur, et mis sur le marché, parfois gratuitement. Mais en générale pour réaliseras des bénéfices par les vend aux concerner selon des contrats qui décrites les droits d'utilisation et l'exploitation de ces produits afin de garantie une transaction légale et juste.

3. Licence d'utilisation de logiciel.

3.1. Principe

Une licence de logiciel est un contrat par lequel le titulaire des droits du logiciel autorise un tiers à bénéficier de son produit [2]. Autrement dit, une licence sert en effet à définir ceux qui ont le droit d'utilisation d'un logiciel, à protéger les droits d'auteur ou au contraire à permettre aux utilisateurs d'accéder au travail de cet auteur.

D'autre part, le moyen technique de valider l'installation est d'utiliser une clé d'activation qui est un ensemble de code peut être donné par l'organisation de développement à ces clients afin de les permettre d'activer leur produit.

En plus, l'obtention d'une licence ne confère que des droits d'utilisation du logiciel. Le problème qu'on confronte est que le licencié souvent distribue ou même revendre le logiciel à plusieurs autres clients, …etc. C'est pourquoi le client doit toujours bien lire le contenu de sa licence de produit pour n'est pas commettre le crime de voler le droit d'auteur.

3.2. Types de licences [3]

Tant qu'il existe plusieurs sortes de licences, on se concentre ici de citer les plus connues.

3.2.1. Shareware

La licence **shareware** attribue un droit temporaire et/ou avec des fonctionnalités limitées d'utilisation pour permettre aux utilisateurs de voir l'efficacité du produit et s'il s'agit ou non du produit voulu. Après une période d'essai, l'utilisateur devra payer l'auteur un prix pour continuer à utiliser le logiciel ou avoir accès à la version complète. Dans ce type on trouve :

3.2.2. Licence fixe

La licence fixe est conçue pour être installée sur un ordinateur particulier. Elle peut utiliser une caractéristique spécifique à cet ordinateur, comme son adresse *Media Access Control* (MAC) pour vérifier et contraindre la conformité de l'usage de la licence.

3.2.3. Licence nominative

La licence nominative, qui à remplace la licence fixe, a pour but de permettre l'utilisation d'un logiciel sur un ordinateur définie avec une adresse MAC par exemple

3.2.4. Licence flottante

La licence flottante fonctionne avec un ordinateur serveur de licence(s). Celui-ci décompte le nombre de licences utilisées à un instant « T » sur le réseau. Tant qu'au moins une licence reste disponible, tout ordinateur du réseau réclamant une licence de la verra affecter temporairement durant le temps d'utilisation du logiciel concerné.

3.2.5. Licences libres

Une **licence libre** est la licence par laquelle l'auteur autorise tout ou partie de ses droits aux utilisateurs de son produit. En même temps, ces libertés peuvent être soumises à conditions, notamment l'interdiction d'utilisation de l'application à des fin commerciales.

4. Clé de produit [4]

4.1. Définition

Une **clé de produit** est un ensemble de code, une chaîne de caractères spécifique (chiffres et/ou de lettres.) ou un fichier bien préparé pour activer une copie d'un logiciel. Généralement, la clé de produit doit être fournie au logiciel lors de son installation pour que le logiciel soit activé. La vérification de la clé se fait avec ou sans connexion à Internet selon les cas. Une vérification avec connexion à Internet permet à l'éditeur du logiciel de s'assurer que plusieurs utilisateurs n'utilisent pas la même clé.

Il n'est pas nécessaire que tous les logiciels doivent posséder de clé de produit, car certains éditeurs peuvent choisir d'utiliser une méthode différente pour protéger leurs droits d'auteur ou, dans certains cas, tels que les logiciels libres ou les gratuiciels, la protection des droits d'auteur n'est pas utilisée.

4.2. Types de clés

4.2.1. Clé de licence en volume

Une clé de licence en volume (en anglais, *volume license key* ou VLK) est une clé de produit utilisée lors de l'installation d'un logiciel sous licence en volume, ce qui permet d'utiliser une seule clé de produit pour plusieurs installations.

Cette forme de licence s'applique généralement aux entreprises, aux gouvernements et aux établissements d'enseignement. Les prix des licences en volume varient en fonction du type, de la quantité et du terme d'abonnement applicable.

Par exemple, les logiciels Microsoft disponibles dans le cadre des programmes de licences en volume comprennent Windows XP, Windows Vista, Windows 7 Enterprise, Windows 8 Enterprise, Windows Server 2008, Microsoft Office 2007 et bien d'autres.

4.2.2. Clé d'activation multiple [4]

À partir de Windows Vista, Microsoft a remplacé les clés de licence en volume par des clés d'activations multiples (en anglais, *Multiple Activation Keys* ou MAK) et des serveurs de gestion de clés (en anglais, Key Management Server (en) ou KMS). Les

logiciels activés via un serveur de gestion de clés doivent faire rapport à ce serveur une fois tous les 180 jours.

4.2.3. Efficacité et Méthodes alternatives

Les clés de produit induisent de nombreuses contraintes aux utilisateurs. Non seulement elles doivent être entrées chaque fois qu'un programme est installé, mais l'utilisateur doit également être sûr de ne pas les perdre. La perte d'une clé de produit signifie généralement que le logiciel est inutilisable une fois qu'il a été désinstallé ou que l'ordinateur sur lequel il a été installé est perdu, volé ou en panne.

Aussi, les clés de produit ne sont pas suffisantes pour arrêter la violation des droits d'auteur de logiciel, car ces clés peuvent être distribuées. De plus, avec l'avènement de l'Internet, des attaques plus sophistiquées sur des clés telles que le *software cracking* (la suppression de la validation de la clé) et les générateurs de clés sont devenues courantes.

Pour cette raison, les éditeurs de logiciel se tournent de plus en plus vers des méthodes alternatives pour s'assurer que les clés sont à la fois valides et non compromises. Une méthode, la validation du produit, associe la clé de produit à une caractéristique unique du matériel informatique de l'acheteur (comme son adresse MAC) qui ne peut pas être facilement dupliquée.

D'autres méthodes plus récentes nécessitent une validation périodique de la clé auprès d'un serveur web (pour les jeux avec un composant en ligne, cette validation se fait chaque fois que l'utilisateur se connecte au jeu). La validation peut ainsi être effectuée sur le côté du serveur, ce qui empêche le craquage de logiciel (qui se fait sur le côté client). Le serveur peut également conserver une liste noire des clés connues comme étant invalides et refuser l'accès aux utilisateurs de ces clés.

5. Protection de logiciels.

La protection d'un logiciel c'est ne que l'interdiction de tout exploitation illégale de ce logiciel c'est à dire sans avoir la permission du producteur original, que ce soit d’une manière directe par le partage de clé et par installation multiple ou bien indirect à travers le copiage des certaines parties réutilisable.

5.1. Protection basée logicielle

5.1.1. Méthode classique

Au départ on dispose donc d'un exécutable non protégé. On passe cet exécutable non protégé à un "protector" (exécutable qui protège un autre exécutable). Ce "protector" met en place une technique dites de "Wrapping" (executable wrapping ou binary wrapping, etc.). Techniquement, il s'agit de mettre autour du code original une enveloppe (le "wrapper") contenant la protection. i.e, en ne créé par un nouvel exécutable, on utilise l'exécutable original en y "injectant" un nouveau code (la protection).

Lorsque l'on démarre le programme protégé, au lieu de démarrer sur le code original, c'est l'enveloppe de protection qui est démarrée. Elle vérifie (suivant un ou des algorithmes précis) que le programme peut démarrer :

- Si oui, elle passe le contrôle au code original.
- Si non, elle met fin au processus et le contrôle n'est jamais passé au code original.

Le problème avec ce type de protection c'est que si l'on supprime le wrapper, on retrouve directement le code original et la protection devient caduque.

5.1.2. Méthode avancée

Dans les protections "sérieuses" le "code original" est bien souvent du code très retouché, avec de multiples protections. Dans ce cas, l'enveloppe et le code original sont très entremêlés, ce qui rend difficile la suppression de la protection sans affecter l'algorithme ou la logique métier (ce que fait - par essence - le code original).

Parmi ces protections on retrouve (notamment) [5] :

- Les anti-debugs.
- Les layers de chiffrement.
- Les layers d'obfuscation.
- La "VM-isation" du code assembleur original.
- La redirection d'API.
- L'émulation d'API.

- Les hooks systèmes.
- Les protections physiques (dongles).
- Les protections s'exécutant en mode noyau (kernel).

Si on arrive à mêler tout ça, on obtient une bonne protection apte à ralentir les plus coriaces des crackers.

5.2. Protection basée Matérielle

La sécurité basée sur le matériel est toujours la meilleure contre-mesure pour distinguer un accès autorisé d'un autre non autorisé simplement parce que l'information protégée est plus difficile à être consulté et/ou modifié. Sur cette base, de nombreux producteurs d'applications se réconfortent en pensant que si une stratégie matérielle via un dongle (clé matériel) est implémentée pour protéger le logiciel, alors le piratage de logiciels ne sera pas aussi facile.

La protection du logiciel n'est pas différente, et en ajoutant du matériel via un dongle, un autre lien est ajouté à la stratégie de protection globale. Le maillon le plus faible de cette chaîne est la couche d'intégration entre le matériel et le logiciel, mais jusqu'à présent, de nombreux développeurs pensent que le dongle matériel résoudra tous leurs problèmes de sécurité. En fin de compte, la protection de base pour autoriser l'exécution de l'application vient de la présence (ou non) de la clé, qui est essentiellement un contrôle de comparaison. Voici à quoi pourrait ressembler ce contrôle de comparaison :

```
If (dongleIsPresent () == True)
                    RunApplication();
```

Si nous prenons la même vue de ce code dans l'assembleur, cela ressemblera à ceci :

```
TEST EAX, EAX      ; // Est ce que la clé matérielle est présent ?
JZ 00618496        ; // Lancer l'application (si la clé est présente)
```

Et si le lien le plus faible est attaqué :

```
NOP                ; // le test de présence de la clé ne sera pas fait (sauté)
JNZ 00618496       ; // Lancer toujours l'application (même si la clé n'est pas présente)
```

Donc, la meilleure stratégie pour protéger un logiciel est une approche qui peut être variée avec différents systèmes et techniques. Commençons par une analogie dans le monde de la sécurité physique, comme une maison. Disons alors que le papa

(propriétaire) a acheté le meilleur verrou à peine dormant sur le marché pour protéger les objets de valeur dans sa maison.

Et donc, Est-ce que cela arrêtera les voleurs ?

La réponse logique sera Oui et Non.

Et car cela n'empêchera pas les voleurs qui brisant une fenêtre pour entrer dans la maison, mais il arrêtera un voleur (non professionnel) qui ne tente que des portes.

Nous pouvons augmenter la sécurité globale de la maison via un système d'alarme, des caméras de sécurité et même avec un chien de garde et le même peut être appliqué pour un logiciel.

Quand il s'agit de protéger les logiciels, il est important de se souvenir de ces maximes :

1. Compréhension des mécanismes, tactiques et des outils utilisés (ingénierie inverse, débogueurs et désassembleurs)
2. La sécurité est meilleure d'être en couches (combiner différentes techniques)
3. Les techniques de sécurité doivent être améliorés (aucune technique reste toujours incassable)

5.2.1. Définition du dongle

Traditionnellement, un dongle est un périphérique matériel qui se connecte au PC via l'un des ports à l'arrière. Les plus anciens utilisaient le port d'imprimante parallèle.

Cependant, il n'est pas facile de trouver un ordinateur avec un port parallèle ces jours-ci. Par conséquent, des adaptations ont été faites comme le port USB pour pallier ce problème.

5.2.2. Mécanisme de protection avec un dongle

On se limite de parler ici sur les dongles USB qui sont relativement peu coûteux à produire et sont difficiles à casser (cracker). Le fait est que, selon la façon dont la protection de la clé est incorporée, il peut être impossible de cracker un programme sans la clé matérielle. En fait, ce n'est pas complètement impossible mais, si la protection est mal implémentée, le crack du programme pourrait être aussi simple.

Lorsque l'application protégée démarre, elle vérifie si le bon dongle est en place et ne fonctionne pas si elle n'est pas détectée. Les Dongles sont encore utilisés par de nombreuses applications spécialisées, généralement avec des prix relativement élevés.

En effet, l'implémentation d'un dongle peut être réalisé par deux façons :

La première (la bonne) façon consiste à crypter les programmes et de stocker la clé de cryptage sur le dongle et décrypter ces programmes au moment de l'exécution en testant si la clé est connectée ou non.

La deuxième (la mauvaise) façon serait de vérifier simplement la présence de la clé. Cependant, des pilotes de périphérique peuvent être produits pour émuler la fonctionnalité et la visibilité de n'importe quel périphérique, y compris les périphériques USB et parallèles.

5.2.3 Quelques inconvénients des dongles:

Les utilisateurs n'aiment pas les dongles pour diverses raisons qui ont limité leur utilisation.

1. Les dongles généralement peuvent être difficiles à installer et à utiliser, car ils nécessitent souvent un pilote matériel spécial, et ils peuvent interférer avec l'utilisation de périphériques tels que les imprimantes et les scanners.
2. Si chaque programme protégé nécessite un dongle supplémentaire, cela va provoquer une "pile" peu encombrante de dongles connectés au PC.
3. Les dongles ne facilitent pas non plus la distribution de logiciels sur Internet, car un dongle doit être envoyé à chaque client pour permettre le fonctionnement du logiciel.

5.3. Catégories principales :

En peux catégoriser les types des outils de protections selon l’endroit où se déroule le principal traitement qui décide est ce que la copie est légitime ou pas, nous allons citer deux catégories de vérification :

5.3.1. Interne :

Ce type localise tous les processus de contrôle et de vérifications dans le programme principale ou bien dans un de ces composants, mais ils sont toujours dans le même support que le corps du logiciel. Exemples :

a. Numéro de série stockée dans un fichier au niveau de registre du système.
b. La cohérence du résultat de calcul ou de composition de série des données (ID de processeur, ID disque dure, ID carte mère … etc.) avec le résultat d’un processus.

5.3.1.1. Les avantages de la protection interne

Pour ce type en peux citer suivants :

A- La simplicité d'intégration : peu importe votre choix essaiera de protéger votre logiciel on applique cette catégorie ne demande pas trop d'efforts dans le développement ni dans le déploiement.

B- La souplesse de gestion : une fois vous livrer une copie, le client ne seras plus dépond de vous pour l'activation ou la réactivation de sa copie. Il suffit de saisie le code fourni lors de l'installation.

C- La mobilité totale : la copie d'installation fourni sous forme de fichier sur un support unie ou via internet et ne dépond sur aucun autre matérielle ou conditions ce qui la rendre plus facile a transporter et à fournir.

5.3.1.2. Les inconvénients de la protection interne

A l'autre cote en peux citer les suivants :

A- Pour beaucoup entre ces méthodes les crackers en trouvaient la technique de les dépasser totalement.

B- La facilite de distribuer des clés illégitimes.

C- Le propriétaire ne peut pas contrôler le nombre d'utilisation de son produit par un client.

5.3.2. Externe :

Dans ce type, pour obtenir le ok afin de poursuivre l'exécution, le contrôle ce fait à l'extérieur du support ou le logiciel a été installée, ou il obtient une partie nécessaire pour continuer l'exécution (partie de programme, données spécifiques), exemple :

c. Activation online ou via internet.
d. Utilisation du dongle.
e. Vérification de l'existence des certaines conditions.

5.3.2.1. Les avantages de la protection externe :

Parmi les avantages de cette catégorie :

A- Le mécanisme d'activation est loin de la porter du cracker ce qui rend l'opération de dépassement très difficile et couteuse.
B- Le propriétaire contrôle le nombre des copies et d'activation par client ou par clé.
C- La souplesse dans le changement des variables de complexité.

5.3.2.2. Les inconvénients de la protection externe :

En peuvent citer les inconvénients suivants :

A- Coût supplémentaire des dispositifs dongle accompagneront le logiciel.
B- Plus de complexité a l'implémentation.
C- Mobilité très faible du la condition de présence du dongle à chaque lancement.

Chapitre II

Conception

1. Conception générale :

Une description générale de tous qu'il faut pour que l'outil attient son objectif, commençons par la numération des besoins fonctionnels et non fonctionnelles, en suite la description du fonctionnement souhaité en cite toutes les observations et les contraintes.

2. Définition des besoins :

2.1. Les besoins fonctionnels :

1- La technique de protection qui sera proposée doit générer une information qui identifieras un flash disque (par exemple) d'une manière unique.

2- La technique doit être la condition principale pour l'activation d'un copy du logiciel.

3- La technique doit protéger le logiciel contre l'activation ou de son utilisation illégale d'un pc à un autre par l'attachement du fonctionnement du programme avec la présence du flash disque clé.

4- Elle doit attacher le fonctionnement d'un copy du logiciel actif avec un flash disque unique et spécifique.

5- La technique doit contient un plan de recouvrement et de re-attachement avec un flash disque légitime en cas ou le dongle originale est endommagé.

6- La technique doit se protéger soi-même contre le copiage illégal sur plus qu'une flash disque-machine.

7- Elle doit limite le nombre d'installation avec le même flash disque.

2.2. Les besoins non-fonctionnels :

1- Elle doit être une partie du logiciel à protéger.

2- Fonctionne en arrière-plan, n'est pas forcement au lancement du programme uniquement mais à des intervalles du temps, nombre d'utilisation et/ou à des endroits aléatoires spécifiques.

3- En cas de détection d'illégitimité, le logiciel continuera son fonctionnement mais avec le strict minimum de fonctionnement.

3. La technique proposée :

1. Vue qu'il n'y a pas deux flash disque identique au volume compter par bit (une réalité réel industriel) ce que veut dire chaque flash est unique et peut être identifier par sa capacité en bit.
2. En tenant compte de l'option précédente, en veux lier le fonctionnement du programme avec la :
    - La présence d'un flash caractérise par sa taille en bit, et pour rendre la situation plus nuance afin d'être plus loin à analyser et être dépasser par un cracker, on va ajouter quelques options :
        - Utilisation d'un flash diffèrent pour chaque copie du logiciel.
        - Ajout d'une clé stocker sur le flash écrit par la technique (Alternante Data Stream) ADS du Microsoft pour qu'elle soit invisible.
        - La clé à ajouter n'est que le GUID (Globale Unique Identifier) générée pendant la compilation du logiciel. Cela garantie que pour chaque copie auras sa propre clé.
        - Pour camoufler la relation entre les options, un fichier de grosse masse seras enregistrer sur le flash sur la plupart de la surface, afin de protéger contre la modification des informations direct ou bien indirect.

Alors en présume qu'à chaque lancement tentative du programme il doit simuler le diagramme suivant :

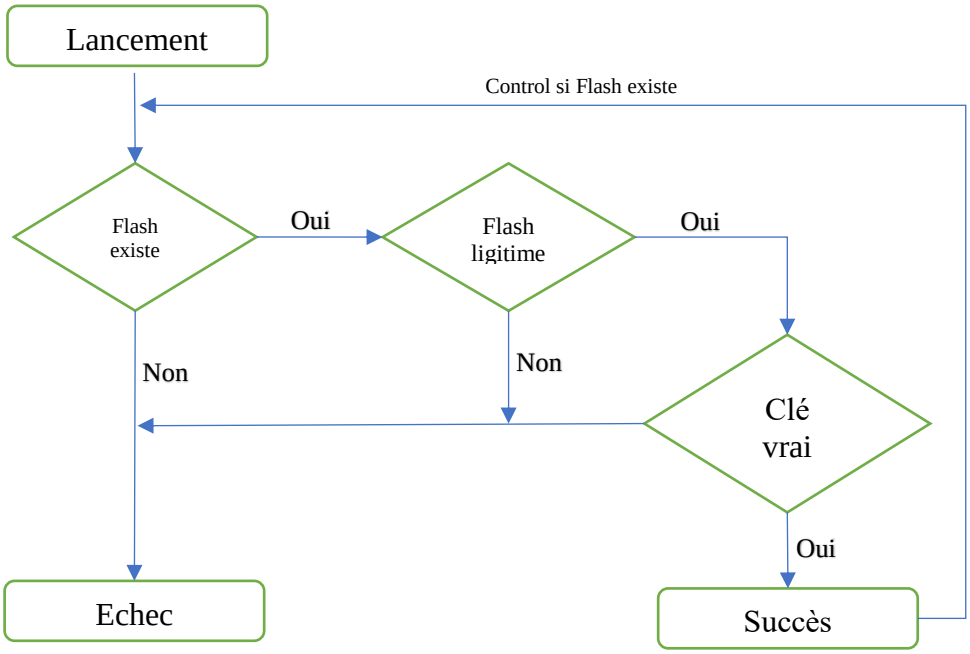

Figure 1: Principe de fonctionnement general de la technique

4. Conception détaillée :

4.1. Contraints :

1. La génération de la bibliothèque faite par l'éditeur d'outil de protection.
2. En mis aux dispositions du développeur un dongle contenant la bibliothèque de protection.
3. Le développeur doit se référencier à l'outil et l'intègre dans son code source pour encapsuler des informations qui conduisent au fonctionnement total de son logiciel.
4. Les traitements du protecteur seront faits en parallèle (dans un thread séparer) pour but à ne pas poser des charges supplémentaires sur le logiciel à protéger.
5. La bibliothèque doit encapsuler des informations nécessaires pour le fonctionnement total de son logiciel.

4.2. Les étapes de fonctionnements :

6. L'outil appelée à partir du logiciel, doit :
    1. **Vérifier l'existence** du débogueur afin de prévenir l'analyse d'exécutions et l'échange des données.
    2. **Examiner** l'existence du dongle.
    3. **En cas** du succès elle vitrifiera sa légitimité par
        - Lire les caractéristiques du dongle.
            - L'espace vide sur le flash en bites.
            - La taille du fichier de nuance (crée avec une taille arbitraire bondant la création du dongle).
            - Le **Timestamp** [6] du fichier nuance.
            - Le hash stockée dans le Data Stream du fichier de nuance (générer en fonction avec la composition des informations précédentes lors du création du dongle).
        - Construire les clés nécessaires (par composition des informations)
        - Compare le Hash de la clé avec le hash lu de dongle.
        - En cas de succès cet clé seras utiliser dans le déchiffrement de la clé stockée dans les Stream du fichier nuance pour décrypter des informations.
    4. Vitrifiera la relation entre le dongle et la copie installée par l'extraction d'une clé du Stream (seconde clé générer lors du première lancement et stockée dans le Stream) et l'utilise pour faire une comparaison avec un clé générer à partir des informations extraites du répertoires du logiciel à protéger. (Ce clé a été générer pendent le premier lancement du programme pour relier le dongle avec la copie et sauvegarder dans le deuxième Stream).

4.3. Les principaux composants de la bibliothèque :

- **Device watcher** : cette partie ce charge de surveiller la présence du dongle attacher à la machine.

- **Dongle Identifier** : de valider la cohérence de dongle, à partir des caractéristiques spécifiques il calcule un hash et le compare avec un hash stocke dans le Stream d'un fichier du dongle. Ces caractéristiques sont : l'espace vide du flash disque, le volume du fichier nuance, timestamp du fichier nuance.
- **Composition Créateur :** cette unité est chargée de création de deux fichier le premier d'une taille aléatoire compris entre un quart et un tiers d'espace de dongle, avec trois Stream secondaires comprennent des data comme valeurs d'initialisation. Le second fichier ni pour que remplir le reste du dongle afin de protéger l'espace restant d'être modifier.
- **CryptsAndHash** : cette unité a pour réalise le calcule de hash et de crypter le hash initial pour plus de protection.
    1. DataEncapsulation : a le rôle de combiner les informations d'une manière prédéfinie pour calculer un hash. Ce dernier sera comparé avec le hash stockée dans le Stream.
- **ADS unit :** pour lire and écrire les informations dans les Stream.

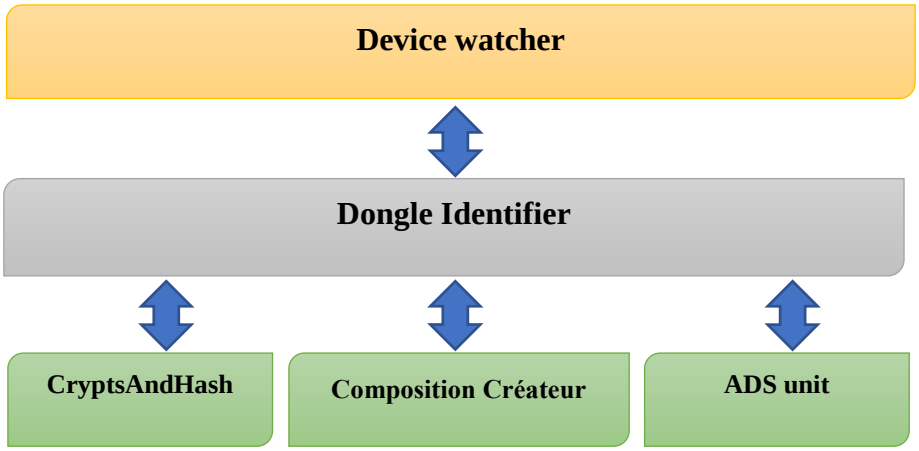

Figure 2 Les principaux composants de la bibliothèque

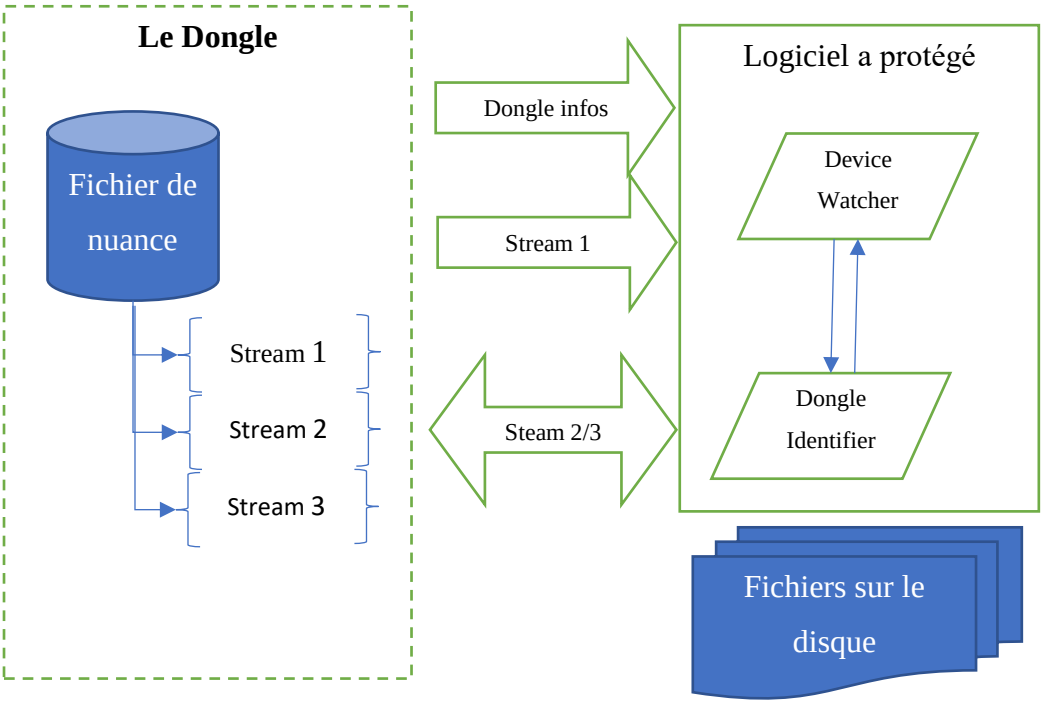

Figure 3 Flux d'informations Dongle-Logiciel

5. La création du dongle :

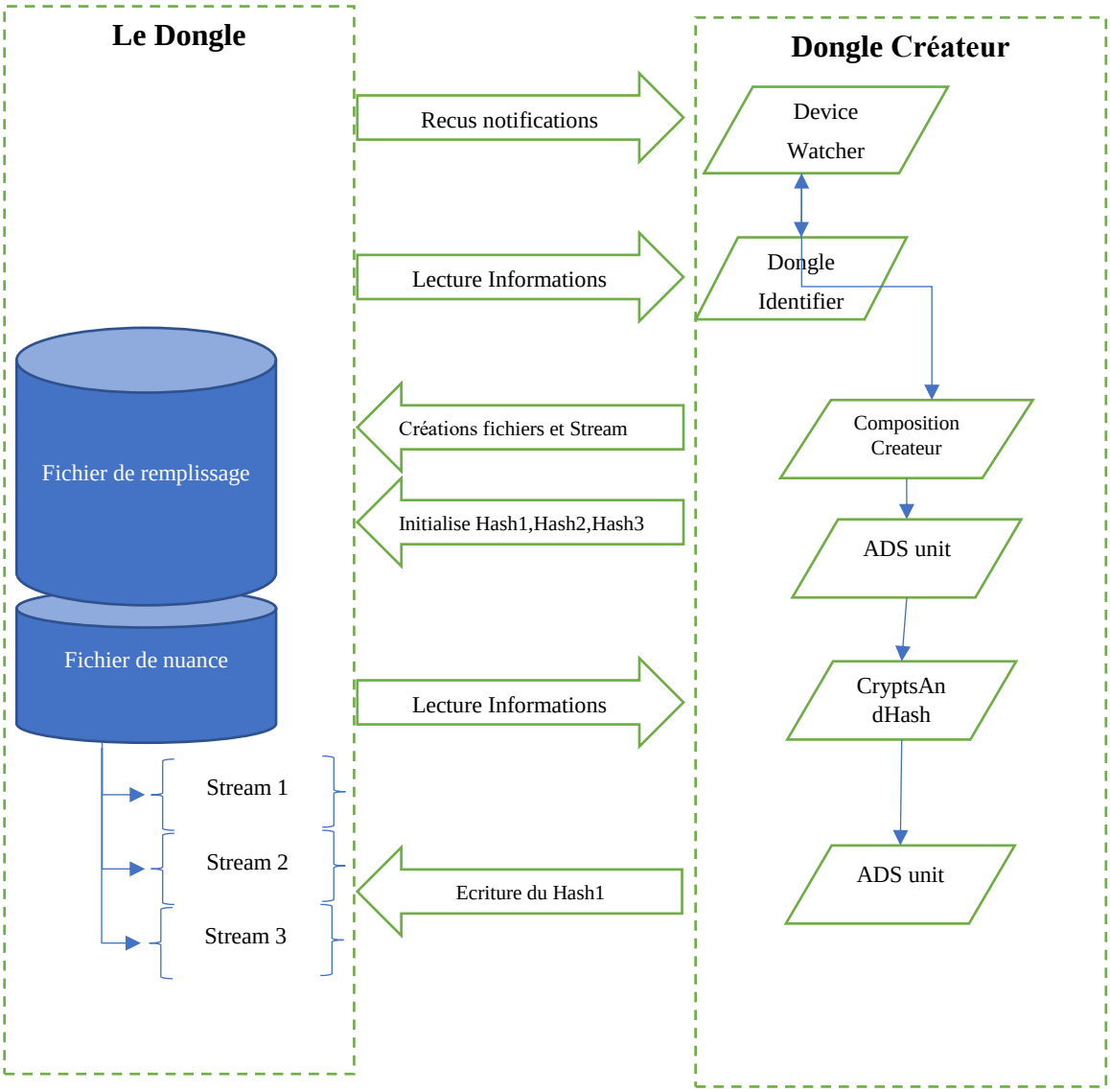

Figure 4 Sequence de fonctionnement du createur du dongle

Chapitre III

Implémentation

1. Préface l'implémentation de la solution :

Dans cette partie nous avons cité pour chaque composant de notre technique son principe de fonctionnement, ainsi une définition de ses sous-parties et les relations entre eux afin d'avoir une idée claire des différentes techniques et approches utilisées au cours du développement.

Pour le premier point, nous avons suivie strictement les étapes de la conception précédemment expliquée par les schémas globale et détaillée. Car elle respecte les principaux de développement le plus possible en ce qui concerne la séparation des traitements dans des unités a travail bien détermines.

L'outil de développement choisie été Microsoft Visual Studio la version 2015, car elle est très connue de notre part et aussi elle a l'avantage d'être la plus proche et la plus intégrable aux systèmes utilisés dans notre marche ciblé.

2. Device Watcher :

Le principal objectif de cette unité pilote est de surveiller les notifications system indiquent qu'un flash disque est placé ou au contraire est retiré de la machine, cela doit être faite en parallèle pour ne pas ralentissez le fonctionnement du programme a protégée.

Alors que l'objet ManagementEventWatcher dans l'espace des noms System.Management permet de s'abonner à des événements dans le contexte WMI - Windows Management Instrumentation. Un changement dans le statut d'un service Windows est un tel événement et il est possible d'être averti quand cela arrive.

```
//
  ManagementEventWatcher watcher = new ManagementEventWatcher();
//
```

Pour que n'obtient que les évènements concernent les périphériques du stockage USB en mis une requête mener d'une condition de filtrage.

```
//
  WqlEventQuery query = new WqlEventQuery("SELECT * FROM
  Win32_VolumeChangeEvent WHERE EventType = 2");
//
```

En doit définir la procédure a exécuter lors de l'évènement ainsi que la requête a lancer pour extraire les informations

```
//
watcher.EventArrived += new EventArrivedEventHandler(watcher_EventArrived);
watcher.Query = query;
//
```

L'étape final est de lancer le watcher dans un thread séparé.

```
//
  watcher.Start();
//
```

3. Dongle Identifier:

Cette unité de qu'on reçut une notification de l'existence, qu'un flash disque est connecte ou bien un des flashs disques a été déconnecté, elle commence à faire une suite de vérifications dont le but final est : de confirmer que le dongle légitime est encore connecté avec la machine ou bien le contraire.

Afin de pouvoir travailler avec les fichiers disques et les informations ainsi qu'avec les volumes nous avons l'espace des nom System.IO.

1- Le premier test est de savoir est ce qu'il y a au moins un flash disque connecte cela est fait simplement par la collections des informations concernent les drivers de type Removable par la logique suivante :
   a. var RemovableDrv = DriveInfo.GetDrives().Where(x => x.DriveType == DriveType.Removable);
   b. return (RemovableDrv.Count<DriveInfo>() != 0)
2- La première caractéristique du dongle est qu'il contient deux fichiers seulement, la somme de ces deux fichiers doit donner la capacité du dongle en bytes, cette option seras vérifier par la séquence :
   a. string[] MyFilesListe = Directory.GetFiles(Drv.Name, "Nawel.txt");
   b. if (MyFilesListe.Length == 2)
   c. {

```
// Calculates the sum of the two files which must equal to the total dongle space.
Int64 Space = 0;
foreach (var item in MyFilesListe)
{
  Int64 fileSizeInBytes = new FileInfo(item).Length;
  Space = Space + fileSizeInBytes;
}
if (Drv.TotalSize != Space)
  isOk = false;
}
```

3- En passe à l'étape pour voir légitimité du dongle par la cohérence entre le has de composition des informations sur lui et le hash inclue dans le Stream1.
   a. Demande à l'unité ADS unit de lire le Stream1.
   b. Composition de la clé à partir des informations acquises du contenue du dongle (tailles des fichiers, leur timestamps).
   c. Calcul du hash de la clé générer et lui comparer avec la valeur provenant du "a".
   d. Si la comparaison est positive alors elle demande au CryptsAndHash unit de fournir une fausse

4. ADS unit :

C'est là ou se regroupe toutes les routines destinées pour les opérations du :

1. La création et l'initialisation du Stream1 du fichier Volume1 (ce Stream contiendra le hash de la clé composée de multiples informations du flash disque) qui sert à identifier si le dongle est légitime.
2. La création et l'initialisation du Stream2 et Stream3 du fichier Volume1(ces Stream ont pour l'attachement du dongle avec la copie de logiciel).
3. La lecture des Stream demandée par l'unité dongleIdentifier.

Pour plus d'informations voire Annex 1.

5. Composition Créateur :

Cet unité set la partie principale du création et configuration du dongle. Elle a la capacité de :

1. Formate un flash disque, en définiront le NTFS comme son system de fichier, et lui donnant un nom de volume "Dongle".
2. Cri un fichier de masse, s'appelle Volume2 d'une taille aléatoire comprise entre un quart et un tiers de la taille du flash disque. Ce fichier sera l'host des données des Stream 1 et 2 et 3.
3. A travers les unités "CryptsAndHash" et "ADS unit" elle crier et initialise le contenue du Steam1, Stream2 et Stream3.

4. Créer un fichier de masse, s'appelle Volume 2 d'une façon à remplir le reste de la surface du flash disque. C'est pour camouflage.
5. La dernière opération à faire c'est d'écrire le hash d'une clé composée par la position des informations retenues du flash disque qui sont :
    - La taille du fichier Volume1.
    - Le Timestamp du Volume2.
    - La taille du flash disque en bytes.

6. CryptsAndHash :

C'est la dernière unité dans l'outil, elle est chargée de :

1. Génère la clé d'identification du dongle, qui est compose de la concaténation de la taille du fichier Volume1 et la timestamp du fichier Volume2 ainsi que la taille totale du flash disque. Ces informations sont de haut degré d'identiques (spécifiquement la taille du flash disque ce qui rond l'union des trois plus efficace pour mettre le dongle un et unique).

   //

   public string ComposeKey(string vol1, string vol2, DriveInfo drv)

   //
2. Produire le Hash de la clé générée, afin de le sauvegarder comme signature dans le Stream1 du fichier Volume1.

   Le Hash était choisie comme une solution pour crypter la signature, afin de la mettre loin d'être analyser. Et pour exploiter que la clé peut être composer à tout moment. Et que la moindre modification sera détecter :

   //

   public static string HashString(string str)

   //
3. Pour plus de sécurité on a utiliser une fonction pour effacer le contenue du mémoire qui contiendra les informations inclus dans la génération du clé.

   //

   [DllImport("KERNEL32.DLL", EntryPoint = "RtlZeroMemory")]
   public static extern bool ZeroMemory(IntPtr Destination, int Length);

//

Conclusion générale

Tout au long de notre préparation de ce projet, nous avons constaté que la plupart des techniques de protection basée logicielle utilisées sont assez basiques comme elles sont susceptibles à être cassé. Cependant, celles qui sont basée sur le matérielles sont un peu solides par rapport à l'autres, car elles utilisent des méthodes plus élaborées qui mettent leur craque une tâche difficile.

Dans notre projet, on a essayé d'implémenter une technique efficace basée sur différentes facteurs (matériel et logiciel) qui peut ralentir le processus de leur violation à un niveau acceptable, comme il nous semble au même temps illusoire de penser mettre au point une technique inviolable surtout après avoir vu la liste de programmes crackés.

Enfin, nous pensons qu'un programmeur a tout intérêt de livrer son code source. D'une part, car il ne perdra pas de temps en effectuant beaucoup de protections qui au final risque d'être contournées par les reverser les plus expérimentés. D'autre part en rendant libre le code source, chaque programmeur peut corriger des bugs, des failles de sécurité qui au final renforceront le programme. Les groupes de programmation open-source en sont les meilleurs exemples.

ANNEXE

1. ADS unit en C # avec le client CodeFluent Runtime [7]:

Des flux alternatifs NTFS ? Également connu sous le nom de flux nommés ou ADS (Alternate Data Stream). Eh bien, c'est une fonctionnalité utile dans les systèmes de stockage NTFS. Il étend le concept de flux de fichiers et de données.

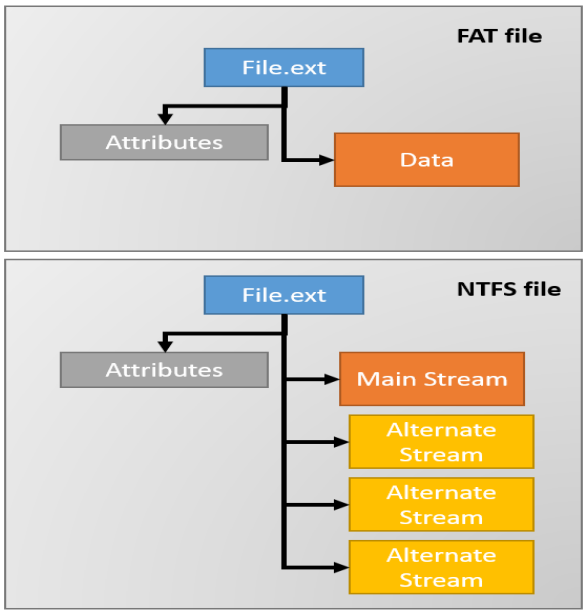

Figure 5: Autres flux de données

Lorsque vous travaillez avec des fichiers NTFS, le flux de données principal (ou flux sans nom) est l'élément central du fichier. Lorsque vous créez un fichier, un flux principal est créé. Lorsque vous créez un flux alternatif et que le flux principal n'existe pas, il est créé. Si vous supprimez le flux principal, le fichier entier est supprimé (donc les flux alternatifs existants).

Lorsque vous lisez un fichier ou que vous écrivez dans un fichier, vous travaillez par défaut avec le flux principal.

Les flux alternatifs suivent la syntaxe : "filename.ext: alternateName"

Vous pouvez stocker n'importe quel type de données dans un ADS (comme vous pouvez le faire avec le flux sans nom), ainsi vous pouvez stocker des données binaires, des données de texte, une image, une vidéo et même un fichier exécutable.

Faisons un test rapide.

Ouvrez une console de ligne de commande (cmd.exe). Créez un fichier texte et écrivez-y du contenu :

```
echo "main content" > test.txt
```

Figure 6: Écriture dans le flux de données principal

Lisons le contenu:

```
more < test.txt
"main content"
```

Figure 7: Lecture du flux principal

Rien d'extraordinaire, nous obtenons le flux de données principal de notre fichier.
Essayons maintenant d'écrire dans un flux de données alternatif (ceci créera le flux alternatif s'il n'existe pas).

```
echo "secret content" > test.txt:hide
```

Figure 8: Inscription à un autre flux de données

Vous avez créé un flux de données alternatif appelé "hide" directement sur notre fichier "test.txt", cela n'aura aucune incidence sur votre flux de données principal. Pour nous assurer que notre flux de données alternatif "hide" a été correctement créé, nous essaierons de le lire.

```
more < test.txt:hide
"secret content"
```

Figure 9: Lecture d'un contenu de flux de données alternatif

Et pour prouver que notre flux de données principal est toujours là, lisons le flux principal.

```
more < test.txt
"main content"
```

Figure 10: Lecture du flux principal

Nous avons effectué quelques tests uniquement avec des flux "texte", mais un flux peut également être une image, un fichier exécutable et tout autre type de flux qu'un conteneur de fichiers peut héberger.

1.1. Qu'en est-il de la manipulation des ADS avec C #?:

Eh bien, cette fonctionnalité n'est malheureusement pas disponible dans .NET, nous aurions besoin d'appeler des méthodes natives si nous voulons manipuler des flux de données alternatifs.

Nous pouvons donc construire de beaux wrappers de méthodes natives afin de manipuler d'autres flux de données ou nous pouvons utiliser le Client Runtime CodeFluent Entities.

CodeFluent Runtime Client est une bibliothèque gratuite qui fournit des assistants très utiles et puissants tels que :

- Utilitaires XML
- Utilitaires IO
- Aide à la conversion de type
- Utilitaires JSON
- ... et beaucoup d' autres

Vous pouvez facilement installer le client CodeFluent Runtime à partir de Nugget .

PM> Install-package CodeFluentRuntimeClient

L'utilisation de CodeFluent Entities Client Runtime pour manipuler des flux de données alternatifs est aussi simple que de manipuler tous les flux de fichiers connus. Nous utiliserons la classe *NtfsAlternateStream* qui se trouve dans l'espace *de* noms *CodeFluent.Runtime.BinaryServices* , elle fournit des méthodes auxiliaires statiques pour manipuler des flux alternatifs car ils étaient des flux "réguliers" (ouvrir, créer, lire, écrire, énumérer, supprimer ...).

Jetons un coup d'oeil à quelques méthodes utiles pour manipuler des flux de données alternatifs (ADS) :

```
//

//Create a stream supporting ADS syntax
FileStream stream = NtfsAlternateStream.Open("test.txt:hide", FileAccess.Write, FileMode.OpenOrCreate, FileShare.None);
stream.Close();
//Writing in to an ADS
NtfsAlternateStream.WriteAllText("test.txt:hide", "Secret content");
//Reading data from an ADS
string text = NtfsAlternateStream.ReadAllText("test.txt:hide");
//Enumerating all the ADS in test.txt
IEnumerable adsStreams = NtfsAlternateStream.EnumerateStreams("test.txt");
foreach (NtfsAlternateStream ads in adsStreams)
{
    Console.WriteLine(ads.Name);
}
//This will not delete the test.txt file
NtfsAlternateStream.Delete("test.txt:hide");
//
```

Un exemple concret de la façon dont ADS est utilisé dans Windows est lorsque vous téléchargez un fichier à partir d'Internet. Lorsque j'ouvre un fichier téléchargé sur Internet avec Word (2013), je reçois un avertissement m'informant que le fichier n'est peut-être pas sécurisé.

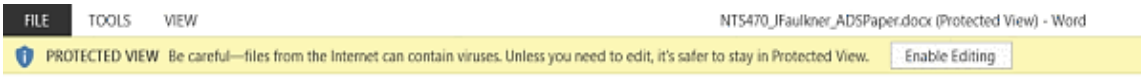

Figure 11: Mot protégé Voir

Comment Word sait-il que j'ai téléchargé le fichier sur Internet? Eh bien, chaque fois que vous téléchargez un fichier, Windows définit un ADS appelé " *: Zone.Identifier* " contenant des données liées à l'origine du fichier. Vérifions cela.

```
        //
        string altFileName =
@"C:\Users\pablo\Downloads\someDoc.docx:Zone.Identifier";
        string content = NtfsAlternateStream.ReadAllText(altFileName);
        Console.WriteLine(content);
        //
```

Ce que nous obtenons est:

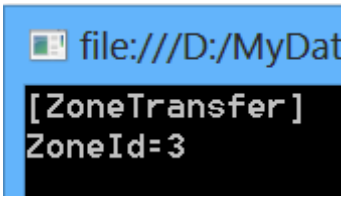

Figure 12: ZoneTransfer ZoneId

Cette valeur " *ZoneId = 3* " signifie que le fichier a " *Internet* " comme origine .

Si nous *supprimons l'* ADS " *: Zone.Identifier* " du fichier:

```
//
string altFileName = @"C:\Users\pablo\Downloads\someDoc.docx:Zone.Identifier";
//this will not delete the file
NtfsAlternateStream.Delete(altFileName);
//
```

2. CryptsAndHash cryptage AES en C # :

Si les fichiers de vos utilisateurs contiennent des informations sensibles, vous pouvez les crypter afin que personne ne puisse ouvrir ce fichier mais l'utilisateur lui-même. Le chiffrement de vos fichiers les rend difficiles d'accès et de lecture sans votre mot de passe. Si vous êtes dans le thème de cryptage dans votre projet, nous allons vous montrer dans cet article comment crypter et décrypter des fichiers en utilisant l'algorithme AES facilement.

Note Bien : Cet article vous montre un moyen de crypter et décrypter des fichiers facilement et rapidement en utilisant des méthodes simples telles que crypter et décrypter. Ils sont le résultat d'une recompilation d'informations provenant de différentes sources comme Stack Overflow, Security Exchange et le site officiel de MSDN.

2.1. Importer les types requis

Afin de gérer l'algorithme de chiffrement AES sur votre projet pour crypter et décrypter les fichiers, importez les 2 types requis suivants :

```
using System.Security.Cryptography;
using System.Runtime.InteropServices;
```

La référence à InteropServices en haut de votre classe vous permettra d'utiliser plus tard le `DllImport` méthode dans notre classe.

2.2. Créez des méthodes de chiffrement et de déchiffrement :

Vous devrez ajouter les 3 méthodes suivantes à votre classe (ou les créer dans une nouvelle classe, puis les importer dans la vôtre) :

- `GenerateRandomSalt` : cette méthode crée un sel aléatoire. Cette fonction est personnalisable et vous pouvez la modifier pour créer votre propre sel si nécessaire.
- `FileEncrypt` : cette méthode crypte un fichier existant avec un mot de passe donnée.
- `FileDecrypt` : cette méthode déchiffre un fichier préalablement crypté avec la méthode FileEncrypt en utilisant le mot de passe donnée comme argument.

- `ZeroMemory` : cette méthode sera utilisée pour supprimer le mot de passe de la mémoire, augmentant la sécurité du cryptage. Ce n'est pas nécessaire, cependant recommandable à utiliser.

2.3. En utilisant les méthodes :

À partir des méthodes requises, vous n'aurez besoin que d'en utiliser 2 (FileEncrypt and FileDecrypt) évidemment et 1 d'entre eux facultatif, le quatrième (GenerateRandomSalt) est utilisé en interne par le FileEncrypt méthode.

2.3.1. Encrypt File

Crypter un fichier en utilisant la méthode FileEncrypt qui attend en premier argument le chemin vers le fichier qui sera crypté et en second argument le mot de passe qui sera utilisé pour le crypter. Le mot de passe est genre à partir des informations lues du dongle (la taille du fichier de remplissage et son timestamp), le même sera utilisé pour décrypter le fichier plus tard. Pour tout faire correctement, nous vous recommandons de supprimer le mot de passe de la mémoire en utilisant la méthode ZeroMemory. Appelez cette fonction pour retirer la clé de la mémoire après utilisation pour des raisons de sécurité.

2.3.2. Décrypter le fichier

Pour déchiffrer le fichier, nous suivons le même processus mais en utilisant FileDecrypt à la place. Cette méthode attend en premier argument le chemin vers le fichier crypté et en second argument le chemin où l'informations décrypté doit être placé. Comme troisième argument, vous devez fournir la chaîne qui a été utilisée pour crypter le fichier à l'origine.

Notes finales

Le processus de cryptage / décryptage consomme de la mémoire et prend du temps, il est donc recommandé d'exécuter ces tâches dans un autre thread pour éviter que votre interface principale ne gèle.

3. Conseils de protection

Si vous insistez pour protéger votre application avec un dongle, tenez compte de mon conseil :

1- Premièrement, quand un bon pirate attaque votre logiciel, il ou elle (généralement pas pour discriminer) recherchera les signes révélateurs de vos routines de fabricants de dongles, pour être honnête, il n'y a pas grand-chose que vous pouvez faire pour dissimuler que vous utilisiez un dongle parce que la communication a finalement lieu via les pilotes des vendeurs de dongle. Supposons donc qu'un pirate n'aura pas beaucoup de difficulté à identifier chacune des API du dongle, ce qui réduit sa tâche simplement à l'élaboration de ce que le dongle devrait retourner / contenir.

   Votre première étape devrait donc être de rendre cette tâche plus difficile. Si votre fournisseur de clé électronique fournit un chiffrement, par ex. l'enveloppe HASP ou Sentinel shell puis l'utiliser et vérifier la falsification des fichiers, cela est particulièrement efficace avec dll qui prennent du temps à déballer. Actuellement, toutes les enveloppes HASP 3 peuvent être rompues parce que l'algorithme HaspCode () utilisé est connu, cependant il ajoutera encore du temps à la progression d'un cracker, HASP 4 reste sécurisé mais peut encore être cassé avec des données de requête connues. Tous les obus Sentinel sont vulnérables à une attaque en clair ou à une force brute.

2- Détecter les débogueurs et les outils anti-débogage, par ex. FrogsICE, SoftICE, seul un cracker va lancer le premier et le dernier est improbable sur l'une des machines de vos clients, utilisez au moins 2 routines pour votre détection et NE PAS dire au cracker que vous l'avez trouvé, si vous le faites modifier le flux du programme subtilement, rediriger le pirate vers une routine qu'il ne veut vraiment pas analyser, ne rien faire d'illégal comme le formatage des disques durs, juste une instance de ce qui arrive à un utilisateur légitime rendra votre produit obsolète.

3- Cryptez également vos chaînes de caractères "mauvais gars" sinon un désassembleur les isolera facilement. Comme le conseil 1, nous finirons par trouver un moyen de contourner le problème, mais le temps est ce que vous achetez ;-).

4- Pour la plupart, un cracker ne sera pas familier avec le fonctionnement de votre programme et les marchands de warez ne testent généralement pas très bien, c'est l'arme la plus puissante que vous pouvez utiliser, les mots de votre dongle peuvent

être utilisés comme les clés de chiffrement, les valeurs dans les instructions switch ou les variables critiques pour des fonctions spécifiques, si votre programme effectue des analyses / calculs mathématiques ou statistiques, un mot dongle ne demande qu'à être utilisé et pour l'amour de dieu ne vérifie pas sa valeur :-). Plus vous doutez de la valeur d'un mot, plus il perd de temps à regarder son débogueur et moins un utilisateur commercial illégitime se contentera de la version crackée, si vous le pouvez, utilisez un cryptage propriétaire (par exemple comme les fonctions de hachage) sur vos données de dongle.

5- Parfois, un pirate sera assez chanceux pour avoir un "vidage" du contenu de vos clés matérielles ou il peut posséder le dongle réelle (très improbable dans mon expérience de la scène warez), si vous pouvez épargner l'espace mémoire dongle utiliser un registre requête pour récupérer la clé RegisteredOrganization et la chiffrer avant de l'enregistrer dans la mémoire du dongle (cela pourrait au moins vous aider à identifier votre client déloyal). Tout pirate digne de ce nom ayant accès à un dongle légitime va briser votre protection (éventuellement), essayez donc d'éduquer vos clients un peu en matière de piratage de logiciels, les petites organisations avec 1 ou 2 licences sont beaucoup plus susceptibles de fuir votre logiciel, soutenez-les si vous le pouvez avec des politiques de licence amicales ou arrêtez de charger le plein prix pour un dongle perdu, vous pourriez faire appel à leur conscience :-).

6- Si vous utilisez Hardlock, HASP ou Sentinel alors la plupart des crackers commerciaux ont déjà des fichiers vxd / sys génériques qu'ils vont simplement configurer avec des clés de registre et les distribuer, cela signifie bien sûr qu'ils ne modifieront pas votre programme, attaquant plutôt les fabricants de dongles code, pensez donc à ajouter des moyens de protection supplémentaires tels que des fichiers de clés fortement cryptés, des contrôles de validation de serveur, des clés de registre cryptées, à peu près tout ce qui va ajouter des heures à la progression d'un cracker.

Enfin, faites une petite visite à certains des moteurs de recherche et découvrez si une fissure existe pour votre programme, si vous pouvez réellement l'obtenir, téléchargez et étudiez les liens faibles de votre protection.[8]

On prend compte des conseils précédemment citer du part d’un équipe professionnelle dans la protection nous pouvons éviter des anomalies et des différentes Lacunes techniques dans la solution à développer.

Bibliographies

[1] : https://fr.wikipedia.org/wiki/Logiciel. Consulté le 25/03/2018.

[2] : Pierre-Paul Lemeyre, « Les logiciels libres sous l'angle de la responsabilité civile » (version du 25 avril 2012 sur l'Internet Archive), 2002 [PDF].

[3] : https://fr.slideshare.net/D1clic/130221-type-de-licence-des-logiciels-ok. Consulté le 01/05/2018

[4] : https://fr.wikipedia.org/wiki/Cl%C3%A9_de_produit. Consulté le 25/05/2018.

[5] : https://www.developpez.net/forums/d570600/autres-langages/assembleur/x86-32-bits-64-bits/proteger-executable-l-encapsulant-verifiant-l-adresse-mac 12/05/2018.

[6] : https://en.wikipedia.org/wiki/Timestamp, consulté le 30/04/2018.

[7] : http://learn2android.blogspot.com/2013/12/manipulating-ntfs-alternate-data.html . Consulté le 05/02/18.

[8] : http://www.woodmann.com/crackz/Dongles.htm#tools Consulté le : 01/04/18.