République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mouloud Mammeri de Tizi-Ouzou (UMMTO)

Faculté de Génie Electrique et Informatique

Département d'Electronique

Mémoire de Fin de cycle

En vue de l'obtention du Diplôme de Master Académique

Domaine : Sciences et Technologies

Filière : Télécommunication

Spécialité : Réseaux et Télécommunications

Thème :

Le cracking et ses différentes techniques

Encadré par :

M^R^ LAHDIR Mourad

Réalisé par :

COULIBALY Lasseni

KONE Mamadou

2017/2018

Remerciements

Nous tenons avant tout à remercier DIEU pour nous avoir accordé sa clémence et nous avoir permis de bien passer notre séjour en parfaite santé.

Nous tenons également, à remercier l'Université Mouloud Mammeri de Tizi-Ouzou, de nous avoir accueillis en son sein pour y passer nos études universitaires.

Nos remerciements les plus réels vont à l'endroit de l'ensemble des enseignants qui nous ont accompagnés depuis le début jusqu'ici, pour leurs efforts dans notre formation.

Et plus particulièrement, nous remercions le département d'Electronique, le personnel de l'administration et le corps professoral pour l'attention qu'ils nous ont porté tout au long de notre cursus.

Nos plus sincères remerciements vont à l'endroit de notre promoteur **$M_{R.}$ Lahdir Mourad** qui a su nous encadrer avec enthousiasme, attention et implication. Nous lui sommes particulièrement reconnaissants de nous avoir placé sa confiance pour faire ce travail, d'avoir toujours été à notre écoute et de s'être montré patient face à nos nombreuses questions. Nous garderons un excellent souvenir de toutes les heures où nous avons travaillé ensemble et pendant lesquelles nous avons pu profiter de ses connaissances et de ses conseils mais aussi de son courage exemplaire.

Nous remercions également **M_{R}. Ouallouche Fethi** et **M_{R}. Mohia Yacine** pour leur sympathie et d'avoir été à notre écoute toutes les nombreuses fois que nous avons eu besoin de leurs services. Recevez ici toute notre gratitude.

Nous remercions vivement nos camarades de classe, qui nous ont facilités notre intégration et notre épanouissement dans le milieu scolaire.

ET enfin, nous remercions sincèrement les membres du jury d'avoir accepté d'évaluer notre travail en dépit de leurs occupations.

Dédicace

Je dédie ce modeste travail ,

A mon cher père et ma chère et tendre maman pour tout son affection et son soutient indéfectible et indescriptible depuis toujours,

A mon très cher professeur de math du lycée : ***Michel Ouattara*** *qui m'a toujours apprécié et soutenu, et dont le courage, la bonté et l'intelligence sont pour moi une source d'inspiration,*

A mes très chers jeunes frères : Yaya, Drissa et Souleymane, qui m'animent de courage chaque jours et pour qui, j'ai l'intime détermination de devenir un grand frère exemplaire,

A toute ma famille de près ou de loin sans exception,

A tous mes amis avec lesquels j'ai passé des moments inoubliables,

Enfin, une dédicace spéciale à mon cher frère et binôme ***Mamadou KONE****, d'avoir donné le meilleure de lui-même pour la réalisation de ce travail.*

Lasseni Coulibaly

Dédicace

Je dédie ce travail a :

Mon père qui est mon mentor, mon modèle de courage et de réussite. Lui qui n'a ménagé aucun effort matériel et moral pour me permettre d'atteindre cet objectif.

Ma chère et tendre mère qui est tout ce que j'ai de plus cher au monde, qui a toujours veillé sur moi et qui m'a inculqué les valeurs de l'humanité. Elle qui m'a accompagne, soutenu et partage mes moments d'échec, de difficulté et de réussite tout cela dans le seul souhait ardent de voir son enfant heureux.

Mon professeur de math du lycée, le feu ***Hamidou Cissé*** *qui de par ses principes, sa rigueur intellectuelle et surtout le souci de voir une jeunesse instruite au service du pays pour un avenir meilleur, a mis son temps et sa personne au service de notre formation scolaire. Que Dieu ait son âme.*

Egalement à mon grand frère Seydou Koné qui est mon modèle de scolarité et dont j'ai toujours admire l'intelligence, mes deux jeunes frères, mes oncles, mes tantes, mes amis, ainsi que tous ceux qui de prêt ou de loin m'ont soutenu et ainsi contribue a la réalisation de cet objectif.

Enfin, une dédicace spéciale à mon cher frère et binôme ***Coulibaly Lasseni****, d'avoir donné le meilleure de lui-même pour la réalisation de ce travail.*

Koné Mamadou

Résumé

Nous somme aujourd'hui dans un soucie de sécurité des informations à travers des attaques qui sont perpétrées à longueur de journée par les pirates informatiques contre les systèmes informatiques (réseaux, systèmes d'exploitation, logiciels, etc.). Après moult recherches, nous avons constaté que la plupart de ces attaques sont faites via internet et principalement à travers des programmes malveillants liés aux logiciels que nous installons sur nos machines. En effet, ces programmes malveillants sont conçus pour des objectifs bien définis telles que : vol d'informations, espionnage des utilisateurs, altération de données, etc.

Pour un simple individu, ces attaques peuvent ne pas représenter une grande gravité. Par contre, l'enjeu de telles attaques est accru dans le cas d'un réseau d'entreprise ou d'une entité gouvernementale où la sécurité des informations est vitale. Dans ce sens, il est nécessaire de s'assurer de la sécurité des logiciels avant de les installer sur nos machines en analysant pas à pas le code source de ces derniers pour désactiver les programmes malveillants si présence y est. Ce processus d'analyse de code source est rendu possible par les techniques de cracking, d'où le choix du thème « **le cracking et ses différentes techniques** ».

Dans ce mémoire, nous traitons dans un premier temps les pirates informatiques, leurs techniques d'attaques et leurs motivations. Ensuite, les techniques de cracking et leur application sur des logiciels.

Liste des mots clés

- Sécurité des systèmes informatiques
- Pirates informatiques
- Hackers
- Crackers
- Logiciels
- Cracking

Sommaire :

Sommaire

Sommaire

Liste des figures :

Chapitre I :

Chapitre II :

Chapitre III :

Liste de table :

Introduction :

Nul n'est ignorant de l'importance que représente le logiciel dans le monde informatique. En effet, le monde informatique est composé de matérielles (hardware) et de logiciels (software). Le logiciel est un ensemble de programmes de traitement automatique de l'information. Il est défini aussi comme étant un programme contenant les procédures et les données nécessaires à faire fonctionner une machine. C'est ce qui place le logiciel au cœur de l'informatique, dont la sécurité est prépondérante, car sa sécurité implique celle des informations qu'il traite. L'arrivée de l'internet a apporté une grande complexité du point de vu sécuritaire en amenant des notions telles que : pirates informatiques, virus, fausses informations, accès non autorisé aux données, etc. C'est ce qui nous amène à nous demander comment sécuriser le logiciel afin de protéger nos informations.

De cette problématique en découle un certain nombre d'interrogation à savoir :

- Qu'est ce qui menace la sécurité des informations ?
- Par quels moyens ces menaces peuvent-elles aboutir ?
- Et comment se protéger face à ces menaces?

Autant de questions qui nous ont poussés à faire le choix de notre thème : « le cracking et ces différentes techniques ». Le cracking est le processus qui vise à contourner la protection d'un logiciel ou d'un système informatique afin d'y accéder sans autorisation. En effet, elle représente la base de toutes intrusions illégales dans des systèmes informatiques. C'est pourquoi nous nous sommes donnés comme objectif dans ce travail, de comprendre les différentes techniques de cracking afin d'être à mesure d'apporter des solutions face à ces problèmes.

Afin de pouvoir apporter des éléments de réponses à notre problématique, nous avons articulé notre travail autour de trois chapitres.

Nous avons décrit dans le 1er chapitre, le piratage informatique en explicitant les différents groupes de pirates informatiques, leurs motivations à s'attaquer aux systèmes, leurs différents modes d'attaques et les moyens dont ils disposent pour mener ces attaques et ensuite quelques dispositifs de sécurité.

Dans le 2ème chapitre, nous avons présenté les différentes techniques de cracking, les outils et les connaissances de base en programmation qui sont indispensables pour cracker un logiciel. Ensuite, une mise en application de ces différentes techniques sur des logiciels dans le 3ème chapitre.

Nous terminons notre mémoire par une conclusion pour résumer les apports essentiels de notre travail ainsi que par des perspectives ouvertes par celui-ci.

Chapitre I: Généralités sur le piratage informatique

I.1. Préambule :

Nous avons l'habitude de protéger nos maisons pour empêcher les voleurs de nuire à nos biens. De ce fait, nous avons besoin de connaitre les motivations des voleurs et de savoir comment parviennent-ils à s'introduire dans la maison afin de protéger cette dernière. Par analogie, dans le monde informatique il est nécessaire de connaitre les pirates informatiques, leurs motivations et leurs modes opératoires pour construire des systèmes de protection en conséquence. Ainsi la connaissance de leurs limites passe par la connaissance des techniques de piratage.

Le terme piratage informatique désigne toutes actions visant à s'introduire dans un système informatique sans en avoir l'autorisation. En effet, les personnes actrices de ces actions sont qualifiées de pirates informatiques.

Dans ce chapitre, nous étudions les pirates informatiques, leurs motivations et leurs manières de procéder.

I.2. Pirates informatiques :

Le terme pirate informatique est souvent utilisé pour désigner un hacker. C'est une personne dotée de compétence en programmation et aimant exploré les détails du fonctionnement des systèmes informatiques. La finalité de cette quête de compréhension du fonctionnement des systèmes informatiques définit, la qualification du hacker [1].

Le nom hacker a eu plus d'une signification depuis son apparition dans les années 50. A l'origine, ce nom était utilisé de façon méliorative pour désigner les personnes chevronnées en informatique et les programmeurs émérites dont l'objectif était de rendre publique et accessible toutes informations pouvant servir à l'Homme de comprendre son environnement sans l'autorisation d'une quelconque autorité.

C'est au cours des années 70 que ce même nom servit pour décrire les révolutionnaires de l'informatique, qui pour la plus part sont devenus les fondateurs des plus grandes entreprises informatiques.

Il faut attendre les années 80 pour que ce nom soit utilisé pour catégoriser les personnes impliquées dans le piratage de jeux vidéos, en désamorçant les protections puis en en revendant des copies. Puis qu'il était jugé que personne n'était capable de tels exploits si ce n'était que les hackers. C'est alors que les hackers ont été classés en trois catégories : les bons

hackers (**white hat hackers**), les mauvais hackers (**black hat hackers**) et les hackers gris (**grey hat hackers**). Puis d'autres groupes sont apparus au fil des années. Ainsi, les pirates informatiques sont classés en groupes selon leurs motivations et leurs expériences [2] [3].

I.2.1- White hat hackers:

Ce sont des hackers au sens noble du terme, ce sont des professionnels de sécurité, généralement employés dans des organismes officiels pour assurer la sécurité des systèmes informatiques. Leur but est d'aider à l'amélioration et à la manipulation virtuose des systèmes et technologies informatiques. Ils adoptent une attitude expérimentatrice qui a sous-entendu la plupart des grandes innovations des principaux protocoles, systèmes d'exploitation dont Linux et outils informatiques que nous utilisons aujourd'hui [1].

Les white hat hackers sont motivés en règle générale pour une des raisons suivantes :

- L'apprentissage ;
- L'optimisation des systèmes informatiques ;
- La mise à l'épreuve des technologies jusqu'à leurs limites afin de tendre vers un idéal plus performant et plus sûr [2].

I.2.2- Black hat hackers:

Plus couramment appelés pirates informatiques, ce sont des personnes ayant une connaissance parfaite des outils informatiques et qui les utilisent en s'introduisant dans des systèmes informatiques dans un but nuisible.

Les motivations des black hat hackers sont multiples et peuvent être les suivantes :

- L'attrait de l'interdit ;
- L'intérêt financier ;
- L'intérêt politique ;
- L'intérêt éthique ;
- Le désir de la renommée ;
- La vengeance ;
- L'envie de nuire (détruire des données, empêcher un système de fonctionner) [2].

I.2.3- Grey hat hackers :

Il existe également une catégorie de hackers dit hackers gris (Grey hat hacker), ils sont généralement des white hackers mais peuvent virer au black hat hacking, à la faveur de circonstances diverses (idéologiques, politiques, intérêt financier, etc.) [3].

I.2.4- Script kiddies ou « gamins du script »:

Ce sont des jeunes utilisateurs des réseaux utilisant des programmes trouvés sur internet sans en avoir de connaissances réelles de ces derniers, pour vandaliser des systèmes informatiques, généralement de façons maladroite, afin de s'amuser. Ils sont aussi surnommés crashers, lamers ou encore ''packet monkeys''. La plupart des attaques qui ont lieu sur internet sont l'œuvre des script kiddies [1].

I.2.5- Phreakers :

Les phreakers sont des personnes qui s'infiltrent dans des systèmes téléphoniques afin de téléphoner gratuitement ou sur le compte d'une entreprise. Forts de leurs connaissances dans les réseaux téléphoniques commutés et en électronique, ils peuvent construire des systèmes électroniques (appelés « Box ») ayant des fonctions biens définies telles que :

- Le Blue-Boxing : Les phreakers téléphonent à des numéros verts gratuits, à qui ils envoient un son de 2600 MHz (équivalent au signal qui indique qu'un appel est terminé) grâce a des logiciels. Ce son envoyé fait croire au central du numéro que l'utilisateur a raccroché, alors qu'il est toujours en ligne. Ensuite ils peuvent effectuer des appels à l'extérieur et ils téléphonent où ils veulent gratuitement.
- Les Box: Il s'agit de trafiquer des branchements électriques du téléphone pour obtenir des fonctions qui facilitent la tâche. La plus célèbre des boxes est la Black Box : elle permet à celui qui vous appelle de ne pas payer les communications. Il existe plusieurs types de boxes (beige, gold, etc.) [1] [2].
- Etc.

I.2.6- Carders :

Ce sont des personnes qui s'attaquent principalement aux systèmes de cartes à puce pour comprendre le fonctionnement et en exploiter les failles. Ainsi, le terme carding désigne le piratage de cartes à puce [2].

I.2.7- Hacktivistes :

Le mot hacktivistes est la contraction de hackers et activistes que l'on peut parfois traduire de cyber militants ou cyber résistants. Ce sont des hackers qui utilisent leurs connaissances pour défendre les droits des autres, par exemple, la non commercialisation de l'internet en divulguant généralement des informations retenues secrètes par les autorités. Les hacktivistes utilisent aussi leurs connaissances à des fins purement idéologiques (politiques, religieuses, etc.) en s'attaquant à des organisations, corporations ou entités dont l'activité est antagoniste à la leur [1][2][3].

I.2.8- Crackers :

Le terme crackers est très couramment utilisé pour désigner les personnes dont la motivation est de casser la protection des logiciels pour en faire les copies et de les revendre. En réalité, le cracker est toute personne qui crée ou qui utilise des outils logiciels permettant de casser les protections contre l'utilisation non autorisée d'un système. Ainsi, tout pirate informatique est à la base un cracker puisqu'ils utilisent tous, des moyens de cassage de protection pour s'introduire dans des systèmes informatiques [3].

Les motivations des crackers sont parfois difficiles à cerner, mais nous trouvons entre autre :

- Pour utiliser un logiciel protégé ou une fonction bridée sans posséder le CD ou la licence correspondante.
- Permettre une interopérabilité, une amélioration du fonctionnement d'un programme non prévue à l'origine par l'éditeur.
- Tricher dans un jeu vidéo.
- Eviter les problèmes d'interopérabilité avec les composants matériels et applicatifs de l'ordinateur (ex : lorsque le logiciel monopolise le lecteur de CD ou refuse de s'installer quand possède légitimement un graveur et un logiciel de gravure).
- Relever un défi pour se faire respecter entre crackers.
- Plus rarement, améliorer une application ou corriger un bogue lorsque l'éditeur ne fournit pas le patch correspondant.
- Et très généralement s'introduire dans un système pour commettre des actions illégales (vol d'informations, utilisation des ressources, etc.) [A].

I.3. Méthodologie d'intrusion :

Pour qu'un pirate puisse s'introduire dans un système, il procède d'abord par une certaine analyse et un repérage de failles de ce système. Cette méthodologie générale repose sur un certains nombre d'étapes à savoir :

- l'analyse du réseau dans lequel se trouve le système ;
- la collecte d'information sur ce réseau afin de déterminer les protocoles, les systèmes d'exploitation et les logiciels que ce réseau utilise ;
- la recherche de vulnérabilités du système qui peuvent être exploitées ;
- l'écoute du réseau afin d'intercepter les informations en fuite ;
- et ensuite l'intrusion dans le système.

I.3.1- Collecte d'informations :

La collecte d'informations ou techniquement qualifiée de prise d'empreinte est un préalable à toute attaque. Elle consiste à rassembler le maximum d'information concernant l'infrastructure du système cible telles que : l'adresse IP ; nom de domaine ; protocoles et services activés.

En cherchant ainsi des informations sur ce système, le pirate est capable d'aller jusqu'à utiliser la naïveté et la gentillesse exagérées des utilisateurs de ce dernier. Ce procédé est appelé ingénierie sociale. Elle consiste à entrer en contact avec un utilisateur, en se faisant passer en général pour une autre personne, afin d'obtenir des renseignements sur le système d'information ou éventuellement pour obtenir directement un mot de passe.

Outre les moyens d'arnaque, le pirate peut aussi avoir accès à des données en fuite en tombant sur une clé USB perdue par exemple ou en allant lui-même voler des équipements (PC ou USB) s'il trouve que l'entité n'est pas assez sécurisée.

Après avoir ainsi collecté les informations, le pirate procède à une recherche des vulnérabilités du réseau ou du système d'information [2] [4].

I.3.2- Analyse du système :

Cette étape consiste à scanner le système à l'aide d'outils spécifiques que le pirate crée ou utilise pour déterminer les ports ouverts sur ce système en envoyant successivement des

requêtes sur les différents ports. Ensuite, il fait une analyse des réponses afin de déterminer les ports actifs.

Ce scanneur permet aussi de détecter les vulnérabilités du système que le pirate peut maintenant exploiter [4].

I.3.3- Les vulnérabilités :

La vulnérabilité d'un système représente son niveau d'exposition face aux menaces connues. Elle représente une faille (brèche) laissée par malveillance ou maladresse, dans les spécifications, la conception, la réalisation, l'installation ou la configuration d'un système, ou dans la façon de l'utiliser [1] [2] [4].

I.3.4- Intrusion :

Une intrusion est le fait de s'infiltrer dans un réseau ou un système d'information auquel on n'est pas autorisé. Le pirate après avoir repéré les failles du système, il développe des outils spécifiques pour exploiter ces failles. Grace à ces outils, le pirate peut se mettre à l'écoute du système et intercepter les informations qui y circulent. Il peut aussi prendre le contrôle du système et l'utiliser pour commettre d'autres attaques [4].

I.4. Les moyens d'intrusions utilisés par les pirates informatiques :

Afin de s'introduire dans un système informatique, le pirate peut utiliser plusieurs moyens d'intrusion (selon les circonstances qui lui sont présentées), à savoir :

- Les moyens physiques ;
- Les moyens logiciels ;
- Les moyens par messagerie.

I.4.1. Les moyens physiques :

En générale il est plus facile pour un pirate informatique de s'introduire dans un système s'il peut avoir un accès physique à ce système. Cela lui permet d'installer manuellement les programmes espions et d'avoir accès à des données qu'il souhaite ou de voler directement des informations sur place.

I.4.2. Les moyens par messagerie :

Fort de sa connaissance sociale, le pirate étudie le plaisir, l'ambition et les émotions des personnes afin de les utiliser comme moyens d'intrusion dans des systèmes en envoyant aux utilisateurs des messages dont le contenu incite à cliquer dessus.

Ces messages peuvent être de plusieurs types :

I.4.2.1- Lettre à la chaine :

E-mail dont le contenu incite à faire suivre et partager à un plus grand nombre de personnes. Cette technique peut être utilisée pour véhiculer des malwares (programmes malveillants) ou encore récupérer un grand nombre d'adresse électronique. *Ex* : Pétitions, messages de soutien, etc. [3].

I.4.2.2- Mail-bombing:

Consiste à l'envoi massif de messages électroniques identiques vers le même destinataire à l'effet de saturer le serveur de mails, saturer la bande passante du serveur et du ou des destinataires ou de rendre impossible aux destinataires de continuer à utiliser l'adresse électronique. La trop grande quantité d'e-mail envoyé simultanément rends inutilisable les services de messagerie [1] [4].

I.4.2.3- Spam ou pourriel :

Message électronique non sollicité reçu de destinataires que l'on ne connait pas ou de qui l'on n'attend pas un e-mail. D'apparence peu offensive, les spams peuvent rapidement devenir très gênant quand les messages reçus atteignent un grand nombre. En effet, certains outils spécialement conçus pour le spam peuvent envoyer des centaines d'e-mails non désirés à la minute. En plus de l'inconfort que cela peut entrainer pour l'utilisateur les risques d'infections par e-mail malware, phishing sont accrus [4] [B].

I.4.2.4- Phishing :

Le phishing est une technique d'arnaque qui consiste à récolter illicitement des données sensibles (mot de passe, identifiant, numéro de carte de crédit, etc.) de personnes tierces, en utilisant la tromperie. L'attaquant procède en envoyant de faux liens cliquables ou un formulaire à remplir, en usurpant l'identité d'un requérant légitime et reconnu (banque,

yahoo, facebook, etc.). Dans la majorité des cas, il s'agit d'un e-mail envoyé au nom d'organisme reconnu, incitant à communiquer des données sensibles via un formulaire ou via un lien qui vous redirige sur un faux site copie presque conforme au site original [4].

I.4.3. Les moyens logiciels:

De nos jours, la plupart des attaques se fait sur internet au moyen des logiciels. De ce fait, les pirates informatiques utilisent les logiciels pour s'introduire dans les systèmes, soit en associant des programmes malveillants aux programmes de ces logiciels soit en usant du cracking de mot de passe contre ces systèmes (systèmes d'exploitation, site web ou logiciels).

Ces programmes malveillants sont nombreux et variés selon leur mode opératoire et leurs objectifs. Nous pouvons en citer :

I.4.3.1-Autorun worm (Vers autorun) :

Logiciels malveillants qui s'exécutent automatiquement en utilisant la propriété «autorun» de Windows. Une fois le périphérique infecté est connecté sur l'ordinateur, le programme malveillant s'exécute automatiquement [1].

I.4.3.2- Backdoor :

Logiciel malveillant qui permet de prendre le contrôle de la machine d'un utilisateur via Internet, sans aucune permission. C'est une sorte de programme qui crée à l'insu du propriétaire de l'ordinateur un accès total à son système. Une fois le backdoor installé, il s'incorpore dans le fonctionnement normal de l'ordinateur et est toujours exploitable même après le redémarrage de l'ordinateur. L'attaquant qui installe un backdoor sur la machine d'une victime dispose ainsi d'un accès large à l'ensemble des activités qui y sont menées [3].

I.4.3.3- Boot sector malware :

Logiciel malveillant qui modifie le secteur de démarrage, portion de disque utilisée par le système d'exploitation pour démarrer. Il modifie le secteur légitime et ordonne au système d'exploitation de démarrer à partir d'un secteur infecté contenue dans le programme malveillant. Une fois que l'ordinateur redémarre, le programme malveillant est lancé et s'exécute donc sur l'ordinateur [1] [4].

I.4.3.4- Bug ou bogue:

Dysfonctionnement d'un programme ou d'un matériel survenant suite à une erreur involontaire de programmation (conception) ou de construction. La gravité peut varier et avoir un impact plus ou moins sur l'intégrité et le fonctionnement du logiciel ou du matériel informatique [4].

I.4.3.5- Browser hijacker: ou détourneur de navigateur

Logiciel malveillant qui modifie les réglages d'un navigateur web sans le consentement de son utilisateur. Il est capable de changer la page d'accueil, la barre de recherche, etc. Cette attaque peut être utilisée par un pirate pour booster les revenus générés par un site web, en modifiant le comportement du navigateur [C].

I.4.3.6- Buffer overflow :

Un buffer overflow se produit lorsqu'un programme stocke les données excédentaires en écrasant les autres parties de la mémoire de l'ordinateur, provoquant des erreurs ou des plantages du système. Les attaques de type buffer overflow exploitent une vulnérabilité d'un logiciel, en lui envoyant un volume de données plus grand que celui auquel il s'attendait. Le programme se voit contraint de traiter un volume de donnée plus grand, que l'espace mémoire prévue à ces fins et efface donc les espaces mémoire utilisés par le système d'exploitation [D].

I.4.3.7- Cheval de Troie ou Trojan Horse :

Logiciel d'apparence inoffensive mais qui cache sous sa face apparente un programme malveillant, qui une fois entré dans le système libère sa charge nocive (logiciel malveillant). Le cheval de Troie se présente comme un programme ne réalisant qu'une simple action inoffensive, mais contient d'autres fonctions malicieuses cachées qui s'activent une fois le logiciel installé sur l'ordinateur de la victime. De nombreux programmes gratuits contiennent des chevaux de Troie de nos jours [1] [2] [3] [4].

I.4.3.8- Exploit :

Dans le langage de la sécurité informatique, un exploit désigne le moyen utilisé pour tirer profit d'une vulnérabilité. Il peut s'agir d'un logiciel ou d'un code malveillant qui est utilisé pour exploiter la faille de sécurité détectée. L'exploit n'a de valeur que tant que la vulnérabilité pour laquelle il est conçu n'est pas corrigée [3] [4].

I.4.3.9- Hoax (canular) :

Fausses rumeurs véhiculées sur Internet. De telles rumeurs peuvent conduire des utilisateurs à utiliser des types spécifiques de matériel ou de logiciel infecté au détriment d'autres. Dans le cas d'entreprises concurrentes, cette technique peut avoir des effets désastreux [1] [4].

I.4.3.10- Keylogger :

Le keylogger est un programme malveillant qui permet à l'insu d'un utilisateur, d'enregistrer toutes les frappes qui sont effectuées sur le clavier d'un ordinateur ou de tout autre périphérique tactile. Ainsi le pirate peut récupérer des informations sensibles (identifiant, mot de passe, etc.) saisies par l'utilisateur d'un ordinateur sur lequel est installé un keylogger [2] [3].

I.4.3.11- Logic Bomb :

Aussi connu sous le nom de Fork Bomb, il s'agit d'un programme résidant sur un système informatique qui une fois lancé, recherche une condition ou un état particulier du système pour exécuter une action illicite une fois cette condition trouvée [2]. Il permet de réaliser une action précise si une condition définie par son auteur est vérifiée. *Ex : éteindre l'ordinateur si la webcam est allumée.*

I.4.3.12- Ransomware ou rançongiciel :

Programme malveillant qui, une fois installé sur une machine, empêche l'utilisateur d'avoir accès à ces fichiers et autres fonctionnalités de son système d'exploitation. Ces programmes exigent le paiement des rançons aux propriétaires via paiement électronique, afin de débloquer l'accès aux fichiers ou autres fonctionnalités, d'où le nom de rançongiciel [4]. De façon imagée, il s'agit d'un logiciel qui prend en otage un ordinateur et exige le paiement d'une rançon afin de le libérer.

I.4.3.13- Rootkit :

Portion d'un programme informatique qui cache son activité et les processus qu'il exécute sur un ordinateur, afin de rester totalement indétectable [1]. Ce genre de programme malveillant est utilisé pour cacher les activités malveillantes menées par l'attaquant. Un rootkit peut cacher des keyloggers ou des renifleurs de mots de passe, installés sur un ordinateur.

I.4.3.14- Sniffer :

Outil logiciel dont l'objet est de capturer les trames transitant sur le réseau. Ce genre de dispositif est utilisé afin d'intercepter les informations sensibles qui circulent dans le réseau, sous formes de paquets de données. Il peut s'agir de mots de passe, identifiant, numéros de carte de crédit, courriers électroniques, etc. Lorsque les informations échangées entre les différents segments du réseau (utilisateurs) ne sont pas cryptées, un attaquant peut aisément intercepter et reconstituer les paquets afin de lire en clair les informations échangées [2][3][E].

I.4.3.15- Spyware ou espiogiciel :

Programme malveillant qui est destiné à espionner, épier les habitudes de navigation, etc. de l'utilisateur sur l'ordinateur de qui il est installé. Cette technique est utilisée afin de récolter des informations telles que ; les sites web visités, mots clés saisis, achats effectués sur le web ; afin de dresser un profil de l'utilisateur. Cette technique est majoritairement utilisée par les entreprises commerciales, afin d'épier les utilisateurs et proposer des produits adaptés aux habitudes des utilisateurs, conférant un avantage certain dans un contexte concurrentiel [1].

I.4.3.16- SQL injection ou injection SQL:

Exploit (technique de hacking) qui s'appuie sur les requêtes envoyées vers les bases de données de logiciels, applications, qui ne contrôlent pas suffisamment les accès vers ces dites bases de données. Cette technique permet d'accéder à l'ensemble des données stockées dans la base de données d'un site, en passant par l'interface du site web.

Dans les champs de saisis du site web, l'attaquant saisis des commandes (requêtes SQL) afin d'exécuter des actions non prévues par le concepteur du site. En effet, en lieu et place de ses noms et prénoms, l'attaquant peut saisir une commande qui lui permettra d'extraire les noms et prénoms de l'ensemble des utilisateurs stockées dans la base de données. Les WAF (web application firewall) permettent de diminuer les risques liés à cette attaque en analysant les requêtes SQL envoyées au serveur web via l'interface web [4].

I.4.3.17- Virus :

Programmes malveillants extrêmement nuisibles et dont l'action peut avoir des conséquences très graves sur le système infecté, tel que : vol, suppression de données,

désactivation de logiciels ou fonctions essentielles d'un système, crash, destruction totale du système, etc. Ces programmes ont la capacité de se rependre d'un ordinateur à un autre, d'un réseau à un autre en créant des copies d'eux même sans que l'utilisateur n'en soit conscience [4].

I.4.3.18- XSS (cross-site scripting):

Technique qui consiste à injecter du code malveillant dans des pages web. Ces attaques visent les sites web qui ne contrôlent pas suffisamment le contenu que les utilisateurs du site web sont autorisés à transmettre via les formulaires. Si dans les SQL injections c'est la base de données qui est ciblée, dans les attaques XSS le site web est forcé à exécuter ou afficher le code HTM ou les scripts saisis par les utilisateurs [F]. Ainsi, un utilisateur malveillant peut en injectant du code malicieux dans les pages d'un site vulnérable, déclencher de nombreuses actions, telles que :

- Redirection des utilisateurs qui visitent le site légitime vers un faux site
- Vol de session
- Etc.

I.4.3.19- Zero-Day :

Faille de sécurité présente dans un logiciel et qui n'est pas connue du grand public et pas même par l'éditeur dudit logiciel. La faille et les moyens de l'exploiter ne sont connus que par un nombre restreint de personnes, ce qui augmente le potentiel destructeur de l'attaque [1].

Quand une faille informatique n'est pas publiée, ou connue du grand public, et donc corrigée (patchée) par l'éditeur du logiciel mis en cause, le hacker qui exploite la faille **zero-day** bénéficie d'une relative facilité à mener son attaque, étant donné qu'aucune mesure de protection n'est prévue. Il peut par conséquent prendre le contrôle d'un ordinateur, d'un logiciel ou d'un réseau, voire effectuer une attaque par déni de service, sans que l'utilisateur visé n'ait eu le temps de s'y préparer [E].

I.4.3.20- Drive by download:

Attaque qui consiste à compromettre un site web légitime avec un malware, afin d'infecter les ordinateurs des utilisateurs qui viendront se connecter au site web compromis. Les pirates injectent du code malveillant dans les pages d'un site web légitime et généralement très visité. Une fois que l'utilisateur se connecte à se site web compromis, le code malveillant exploitent des vulnérabilités présentent dans le navigateur web et s'installe

sur la machine de sa victime [1]. Ainsi, le pirate peut prendre le contrôle de la machine et commettre de nombreuses infractions (vol et modification de données, etc.).

I.5. Les types d'attaque :

Les pirates informatiques utilisent plusieurs techniques d'attaques pour atteindre leurs cibles. Ces attaques peuvent se faire selon trois manières, à savoir :

- Les attaques directes.
- Les attaques indirectes par rebond.
- Les attaques indirectes par réponses.

I.5.1- Attaques directes :

C'est la plus simple des attaques. Le pirate attaque directement sa victime à partir de son ordinateur. La plupart de ces types d'attaques sont l'œuvre des "script kiddies". En effet, les programmes de piratage qu'ils utilisent ne sont que faiblement paramétrable, et un grand nombre de ces logiciels envoient directement les paquets à la victime.

Les pirates n'utilisent pas cette technique pour mener des attaques de grande envergure, car il y a de grandes chances pour que la victime puisse remonter à l'origine de l'attaque, identifiant par la même occasion l'identité de l'attaquant [5].

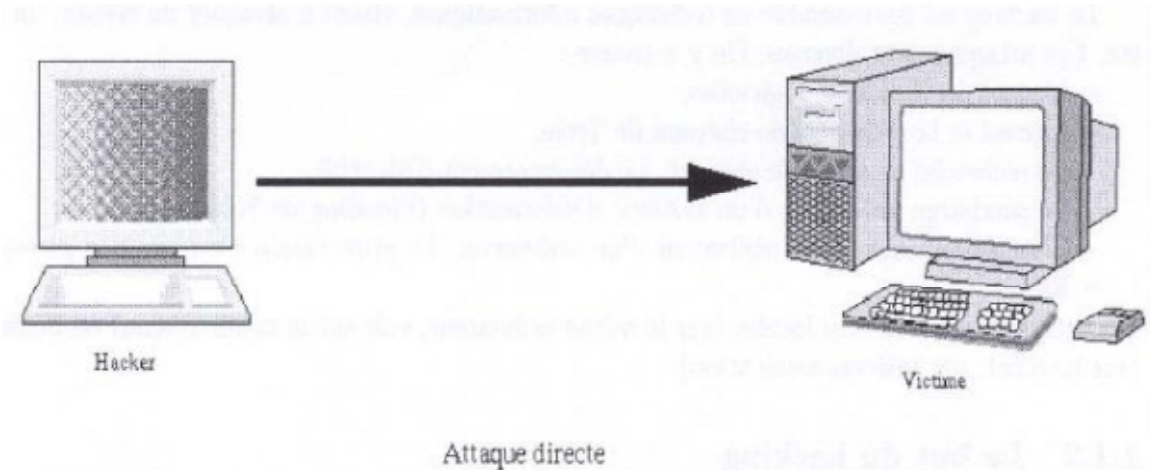


Figure I.1 : Attaque directe

I.5.2- Attaques indirectes par rebond :

Cette attaque est très prisée des hackers. Ce type d'attaque est une sorte d'attaque en série. Elle consiste à attaquer une machine ou un réseau pour en prendre le contrôle et l'utiliser comme rebond pour attaquer une autre machine ou un autre réseau. Ainsi, la machine nouvellement attaquée peut aussi être utilisé pour attaquer une autre et ainsi de suite jusqu'à ce que le pirate atteigne sa cible finale.

En effet, le rebond a deux avantages pour le hacker:

- Masquer l'identité (l'adresse IP) du hacker pour qu'on ne puisse pas remonter jusqu'à lui.
- Éventuellement, utiliser les ressources de l'ordinateur intermédiaire car il est plus puissant (CPU, bande passante...) pour attaquer.

Le principe en lui même, est simple : Les paquets d'attaque sont envoyés à l'ordinateur intermédiaire, qui répercute l'attaque vers la victime. D'où le terme de rebond [5].

Figure I.2 : Attaque indirecte par rebond

I.5.3- Attaques indirectes par réponses :

Cette attaque est une dérivée de l'attaque par rebond. Elle offre les mêmes avantages, du point de vue du hacker. Mais au lieu d'envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, l'attaquant va lui envoyer une requête. Et c'est cette réponse à la requête qui va être envoyée à la machine victime [5].

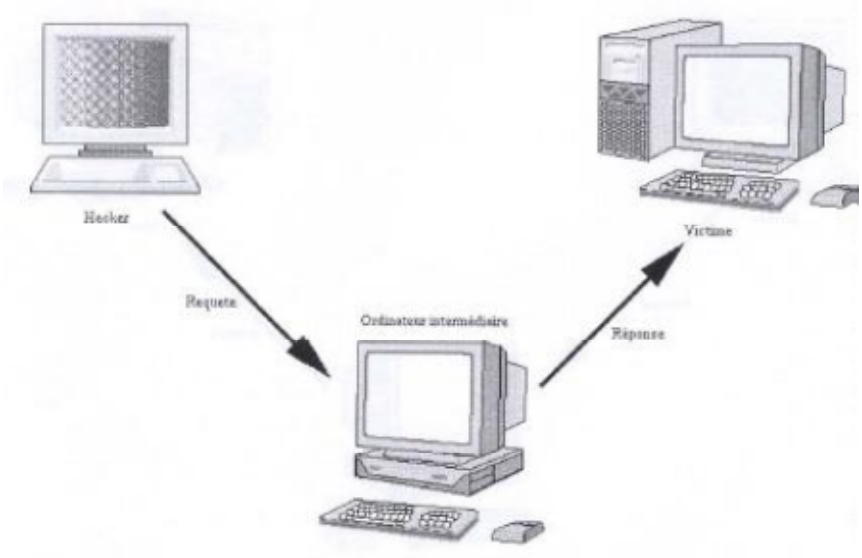


Figure I.3 : Attaque indirecte par réponse

I.6. Les techniques d'attaque :

Il ya plusieurs moyens mises en œuvre pour protéger les systèmes informatiques imposant ainsi à toute personne voulant accéder à ces derniers de s'identifier et s'authentifier afin d'être reconnu autoriser à les utiliser. Le pirate n'ayant aucune autorisation d'accès à ces système veut tout de même s'y introduire pour voler des informations sans être repérer. Pour cela, il utilise un certain nombre de technique d'attaques différentes à savoir :

I.6.1- Attaque de mot de passe :

Lors de la connexion à un système informatique, celui-ci demande la plupart du temps un couple identifiant/mot de passe pour y accéder. L'identifiant n'étant pas secret, seule la connaissance du mot de passe par une personne non autorisée à utiliser ce système peut compromettre la sécurité de ce dernier. Le pirate n'ayant pas connaissance du mot de passe peut s'introduire de quelques manières à savoir :

- L'attaque par force brute : Elle consiste à tester toutes les combinaisons possibles de mot de passe à l'aide d'outils spécifiques pour chaque système. Cette attaque peut s'avérer inefficace car elle peut prendre des heurs, des jours voir même des mois de calcule surtout avec l'adoption des outils de cryptage pour chiffrer les mots de passe. Face aux limites de cette technique, les pirates préfèrent adopter l'attaque par dictionnaire ou hybride.

- L'attaque par dictionnaire : En générale, le choix du mot de passe est laissé libre aux utilisateurs du système et ces derniers utilisent généralement des mots de passe qu'ils peuvent retenir facilement en rapport avec leur nom, année de naissance ou le nom des choses ou des personnes qu'ils aiment. Ainsi, une alternative du pirate consiste à constituer un dictionnaire de mot de passe possibles d'un utilisateur et ensuite procéder au teste de ces combinaisons. Cette attaque est plus efficace que celle précédente.
- L'attaque hybride : Cette dernière attaque consiste à associer les deux attaques précédentes [4].

A l'heure actuelle, ces techniques sont moins utilisées puisque la plupart des systèmes sont configurés de manière à se bloquer automatiquement après un certain nombre de tentatives de connexion infructueuses. Cela a poussé les pirates à chercher d'autres moyens plus efficaces pour obtenir des mots de passe des utilisateurs. Ces moyens peuvent être :

- ✓ Les keyloggers : Ce sont des logiciels, qui une fois installés sur la machine de l'utilisateur, permet d'enregistrer les frappes de claviers saisies par ce dernier.
- ✓ L'ingénierie sociale : Elle consiste à exploiter la naïveté de l'utilisateur pour obtenir son mot de passe en se faisant passer pour une autre personne.
- ✓ L'espionnage : Il représente la plus vielle méthode pour obtenir le mot de passe de l'utilisateur en observant ces faits, ces papiers ou même un coup d'œil par-dessus son épaule lors de la saisie du mot de passe au cas où le pirate fait partie de l'entourage de la victime [2].

I.6.2- Attaque par usurpation d'adresse IP :

Lorsqu'un pirate souhaite s'introduire dans un réseau qui est protégé par un système de filtrage (par feu) qui ne laisse passer que des paquets provenant des machines autorisées. Cela représente un obstacle énorme pour le pirate. Ainsi le pirate utilise des outils spécifiques pour écouter le réseau et intercepter des paquets afin de les exploiter. Une fois que le pirate est en possession du paquet, il peut modifier son contenu en remplaçant l'adresse IP de l'expéditeur par l'adresse IP d'une autre machine du réseau pour que le par feu n'intercepte pas le paquet. Ce procédé est appelé usurpation d'adresse IP [2] [G].

I.6.3- Attaque par déni de service :

Elle consiste à envoyer des paquets IP ou données de tailles afin de provoquer une saturation ou un état instable des machines victimes. Il s'agit la plupart du temps d'attaques à l'encontre des serveurs d'une entreprise, afin qu'ils ne puissent être utilisé [2] [G].

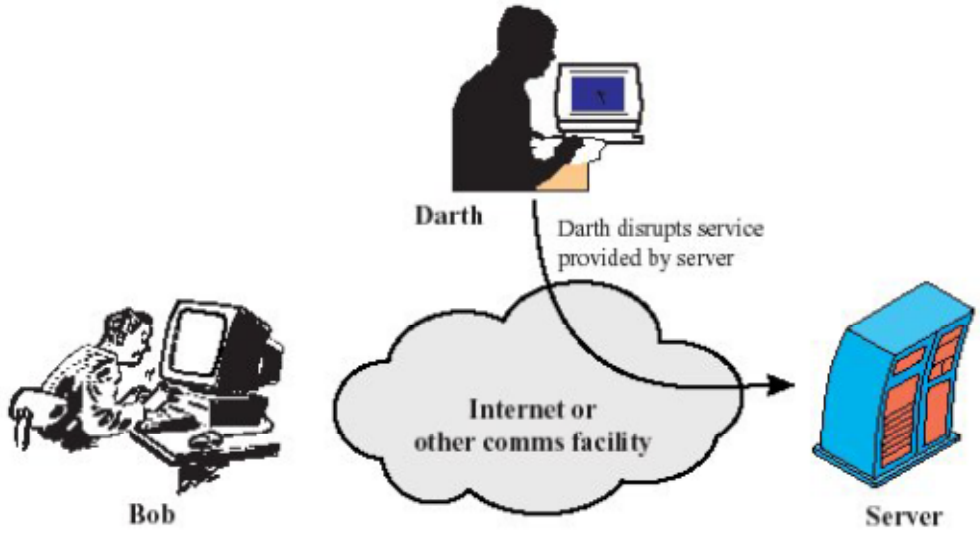


Figure I.4 : Attaque par déni de service

I.6.4- Attaques homme du milieu « man in the middle » :

Cette attaque est aussi appelée attaque de l'intercepteur. C'est un scénario d'attaque dans lequel un pirate écoute une communication entre deux interlocuteurs et falsifie les échanges afin de se faire passer pour une des parties sans que ni l'une ni l'autre ne puisse se douter que le canal de communication a été compromis [6] [G].

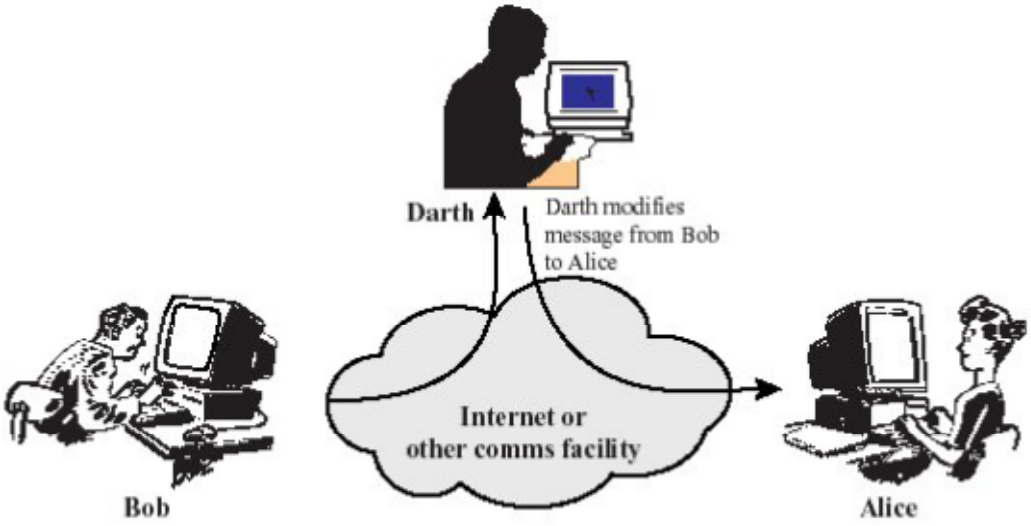


Figure I.5 : Attaque homme du milieu

I.6.5- Attaques par débordement de tampon :

Une attaque par débordement de tampon ou « buffer Overflow » est une attaque informatique qui exploite une faille de dépassement de mémoire tampon par lequel un programme en cours d'exécution, au moment de l'écriture à l'intérieur du tampon, écrit à l'extérieur de l'espace destinée au tampon en écrasant les autres informations indispensables au processus.

Elle consiste pour un utilisateur malintentionné à détourner volontairement un programme comportant cette faille en lui faisant exécuter un code arbitraire ou des instructions qu'il a introduites dans le processus par l'envoie d'une quantité de donnée qui dépasse largement la limite de celle que le programme lui-même est sensé recevoir. Ces types d'attaques peuvent viser un routeur, un serveur de cache ou des logiciels [1] [G].

I.7. Les dispositifs de sécurité :

Afin de contrer les attaques des pirates informatiques, un certain nombre de dispositifs de sécurité sont utilisés à savoir :

I.7.1- Antivirus :

Un antivirus est un programme capable de détecter la présence d'un programme malveillant de type virus dans une machine, et dans la mesure du possible, de désinfecter ce dernier. On parle ainsi d'éradication de virus, la procédure de nettoyage de cette machine infectée [2].

I.7.2- Système par feu « firewall » :

Le système par feu est un système logiciel, reposant parfois sur un matériel réseau dédié, constituant un intermédiaire entre le réseau local (ou la machine locale) et un ou plusieurs réseaux externes. Il permet ainsi de protéger le réseau d'ordinateurs des intrusions provenant d'un réseau tiers (notamment internet), en filtrant les paquets de données échangés avec ce dernier [5] [H].

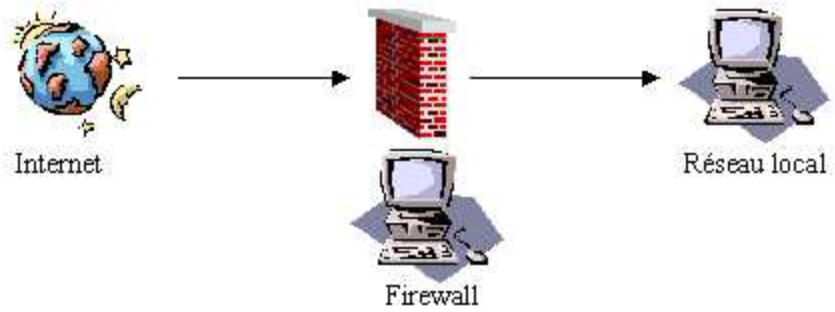


Figure I.6 : Système de par-feu

I.7.3- Serveurs mandataires « proxy » :

Un serveur proxy ou serveur mandataire est une machine sur laquelle est installé le logiciel proxy, qui sert d'intermédiaire entre les ordinateurs d'un réseau local et l'internet [H].

I.7.4- Systèmes de détection d'intrusion :

C'est un mécanisme qui écoute le trafic du réseau de manière furtive afin de détecter les activités anormales ou suspectes sur le réseau en cas de présence d'intrus. Il en existe deux grandes familles distinctes :

- Les **N-IDS** (Network based Intrusion Detection System), ils assurent la sécurité au niveau du réseau.
- Les **H-IDS** (Host based Intrusion Detection System), ils assurent la sécurité au niveau des hôtes [2].

I.7.5- Cryptographie :

A défaut de moyens sures pour empêcher l'interception de données et l'intrusion des pirates dans les réseaux, on adopte la cryptographie comme moyen de sécurité pour assurer la sécurité des données informatiques. Ainsi, la cryptographie est l'art de dissimuler le contenu d'une information afin de la rendre incompréhensible aux yeux d'une personne étrangère à son mode d'emploie. Elle permet de répondre à un certain nombre de conditions à savoir :

- ✓ L'intégrité, qui est le fait qu'une autre personne en dehors des personnes autorisées ne doit pas pouvoir modifier l'information.
- ✓ L'authentification, l'origine de l'information doit être sure.
- ✓ La non répudiation, l'émetteur d'une information ne doit pas pouvoir nier son envoie.

Il y a deux types de cryptographie à savoir :

- **La cryptographie symétrique ou à clé privée:** qui consiste à utiliser une même clé tenue sécrète pour le chiffrement et le déchiffrement d'une information. Cette technique s'est avéré plus efficace à utiliser mais peu fiable puisse qu'elle nécessite un partage de clé pouvant être interceptée par une personne non désirée.

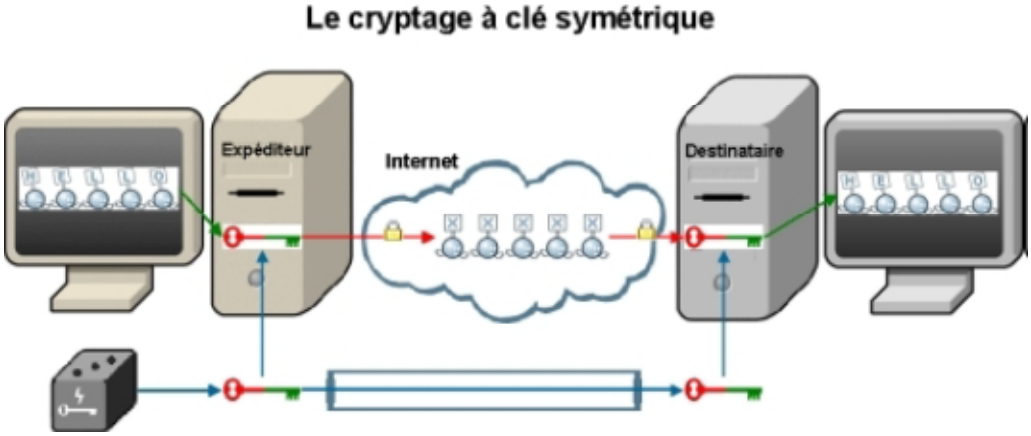


Figure I.7 : Cryptographie à clé privée

- **La cryptographie asymétrique ou à clé publique:** qui consiste à utiliser une clé rendue publique pour le chiffrement d'une information et une autre clé tenue sécrète pour le déchiffrement. Elle est plus solide du fait du non partage de la clé de déchiffrement, mais plus complexe à réaliser et lente pour le déchiffrement [3] [5] [6].

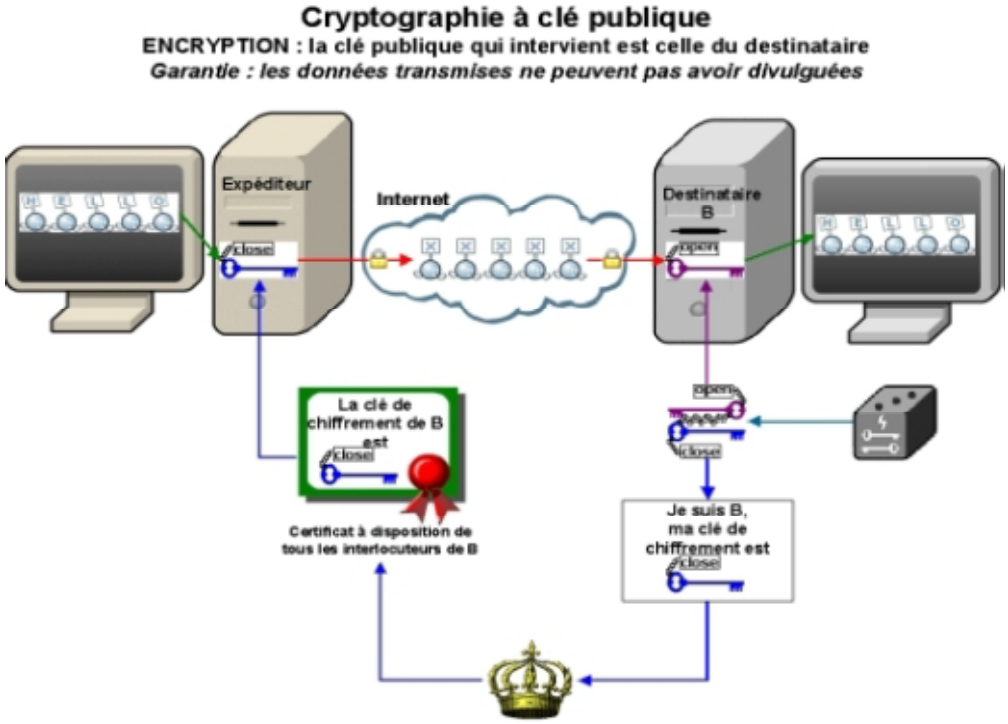


Figure I.8 : Cryptographie à clé publique

I.8. Discussions :

Ce chapitre nous a permis de comprendre de façon globale les menaces réelles qui planent sur le monde informatique, mais la liste n'est pas exhaustive, il y a beaucoup d'autres menaces dont nous n'avons pas eu connaissance et d'autres qui ne sont pas encore connues du grand public.

Aussi, les dispositifs de sécurité mis en œuvre en prévention de ces menaces ont montré leurs limites face aux pirates informatiques. En effet, un pirate étant une personne parfaitement doté de connaissances en informatique, peut aussi concevoir et vendre un dispositif de sécurité avec une faille qu'il peut utiliser à son avantage. Par exemple :

- Le pirate peut concevoir un serveur proxy et le proposer comme fiable à un administrateur d'un réseau, une fois que celui-ci l'utilise, Le pirate pourra ensuite collecter toutes les informations que ce réseau échange avec internet.
- Il peut concevoir un logiciel en injectant un programme malveillant indétectable qui lui permettra d'avoir accès aux réseaux ou aux systèmes de ces utilisateurs une fois que ses derniers l'auront installé.
- Il peut aussi créer des virus et les propager et ensuite proposer des antivirus pour se faire de l'argent.
- Etc…

De ce point de vu, les pirates ont un certains avantage sur les responsables de sécurité puisse qu'ils étudient méticuleusement les systèmes avant de s'attaquer à ces derniers, ce qui fait que nous devons aussi apprendre leurs techniques pour mieux sécurisé nos systèmes. Pour cela, nous étudions dans le chapitre suivant, les différentes de cracking que les pirates utilisent pour s'introduire dans les systèmes.

Chapitre II: Les différentes techniques du cracking

II.1. Préambule :

Aujourd'hui, le monde informatique est soumis à un certains nombre de problèmes qui menacent la sécurité des informations qui y véhiculent. Ainsi, pour mieux sécuriser ces informations, nous devons tout d'abord nous préoccuper de la sécurité des systèmes qui les abritent. Pour cela, nous devons nous attaquer à la menace qui se joue contre ces systèmes, qui s'appelle « le hacking de systèmes informatique » communément appelé « **le cracking** ».

Le cracking est l'art d'enlever, de casser ou de contourner la protection d'un système informatique afin de s'y introduire sans autorisation. Il peut se manifester à trois niveaux à savoir :

- Le cracking d'un site web afin de s'introduire dans un réseau via sa page internet.
- Le cracking d'un système d'exploitation afin de s'introduire dans une machine.
- Et le cracking d'un logiciel protégé ou d'un logiciel de protection d'une information.

En effet, Le système informatique est composé de matérielles et de logiciels. Le logiciel défini comme un ensemble de programmes de traitement automatique de l'information représente les trois quart (3/4) du monde informatique, regroupant ainsi les systèmes d'exploitation, les protocoles et même les sites web. D'où l'importance à faire de la sécurité des logiciels une priorité.

Afin de pouvoir sécuriser les logiciels, il est nécessaire de comprendre le cracking. De ce fait, nous traitons dans ce chapitre, les différentes techniques de cracking logiciel et les outils qui en sont nécessaires.

II.2. Les différentes techniques du cracking :

Dans le cracking, on utilise plusieurs techniques pour cracker un logiciel, mais il y a principalement trois (3) grandes techniques qui sont très utilisés dans la plupart des cas. Elles sont les suivantes: le patching (modification de programme), le serial fishing (pêche au sérial) et le keygenning (génération de clé).

II.2.1. Patching:

C'est une technique qui permet de modifier les parties d'un programme. Nous pouvons le modifier de telle sorte que le programme exécute une tâche spécifique que nous souhaitons

et qui n'était pas prévue par l'éditeur du programme. Par exemple, nous enregistrer dans un logiciel avec n'importe quel code, ou qu'un jeu démarre sans disque CD.

Cette technique est relativement simple car en général, il suffit de modifier un saut ou d'inverser une valeur pour que le logiciel soit cracker sans pour autant avoir besoin de vraiment comprendre le code dans les moindres détails.

Cette technique est la plus simple et la plus utilisé par les débutants, mais seulement sur les programmes simples. Dans des programmes plus complexes, il serait plus commode d'utiliser d'autres techniques que le patching, puis que le saut que nous devons modifier pour nous enregistrer va se répéter plusieurs fois dans le programme, d'où la complexité d'en trouver le bon [I].

II.2.2. Sérial fishing:

Cela signifie littéralement « Pêche au Sérial ». Dans cette technique, nous devons analyser le code source du logiciel afin de trouver le code valide dans le programme en rapport d'un nom entré. Cela se complique un peu plus car nous devons retrouver un sérial en valeur hexadécimale dans un registre ou une adresse mémoire [I].

II.2.3. Keygening:

Le keygenning signifie littéralement « Génération de Code ou Clé ». Cette technique consiste à retrouver l'algorithme de génération du Sérial dans tout le code du programme. Quand nous rentrons un code dans le logiciel, le programme crée un bon code et le compare au sérial entré. S'il ya des différences, le code est faux et s'il y a égalité, le code est juste.

Le keygenning consiste à retrouver la petite partie du programme qui crée le code et ensuite de la reproduire pour créer un « générateur de codes valides ».

Elle est assez compliquée, car elle requiert de nombreuses connaissances en programmation assembleur et aussi beaucoup de logique.

La difficulté principale est de comprendre l'algorithme et ensuite de le reproduire dans n'importe quel langage de programmation [I] [J].

Cette technique est plus difficile que les précédentes, mais elle reste la plus utilisé par les crackers, car elle a l'avantage de générer un code valide à partir de n'importe quel nom et permet donc un enregistrement pour tous.

II.3. Qu'est ce qu'un crack ?

Un crack est un programme qu'un pirate crée pour pouvoir modifier les octets d'un programme afin de changer son comportement. Cela peut être pour :

- Modifier la partie de protection d'un logiciel pour qu'il lui laisse entrer sans avoir le vrai mot de passe.
- Modifier une partie du programme pour qu'il exécute une tâche spécifique.
- Ou modifier une partie du programme pour l'empêcher d'exécuter une tâche.

II.4. Quelques outils utilisés pour cracker un logiciel:

Pour cracker un logiciel, il nous faut des programmes multiples et divers ayant des fonctions spécifiques. En voilà quelques uns :

II.4.1. Un Désassembleur:

Le désassembleur comme son nom l'indique, permet de désassembler ou d'afficher le code source d'un logiciel en assembleur. Cela nous permet d'analyser le code source du programme afin de voir la partie ou les octets qu'il faut modifier et à quel endroit ils se trouvent, sans pour autant nous permettre d'apporter une modification quelconque. Ce programme est quasi indispensable dans le cracking [K].

Exemples : WinDasm, IDA, …[L]

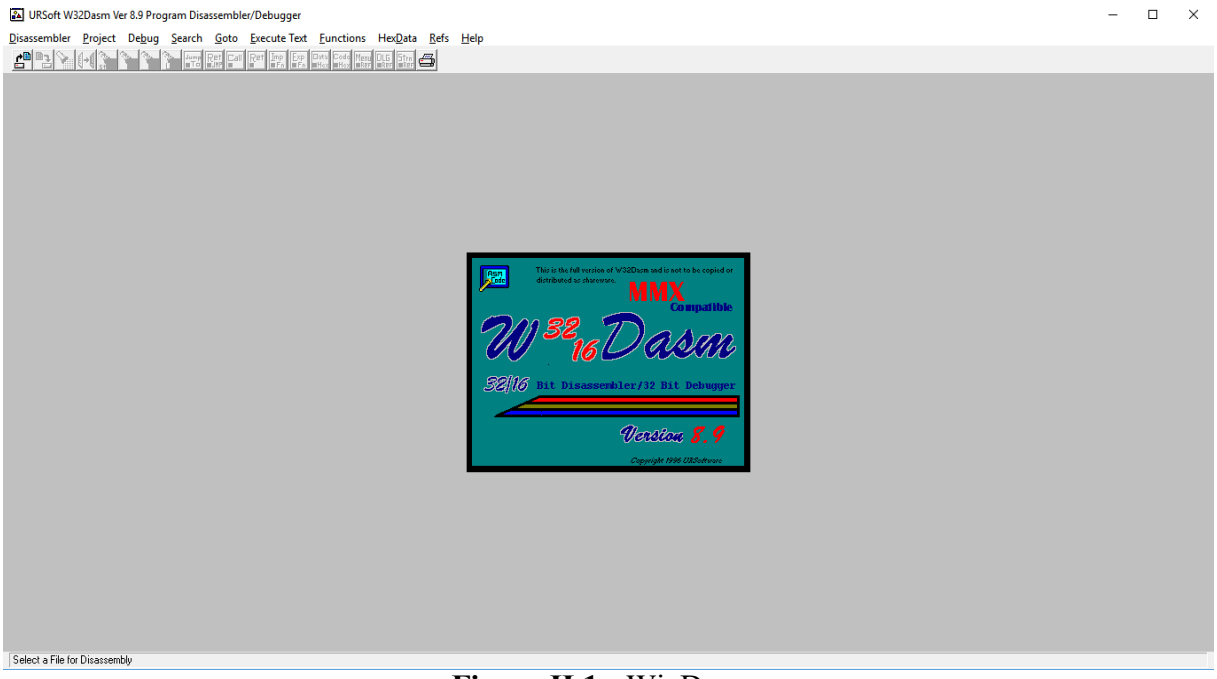


Figure II.1 : WinDasm

II.4.2. Un Débuggeur:

Le débuggeur permet d'analyser le programme lors de son exécution, afin de voir les valeurs des registres et des opérations que le programme effectue. En générale, le programme de débuggeur et désassembleur sont regroupés dans un même outil [K].

Exemples : OllyDBG, SoftIce, … [M]

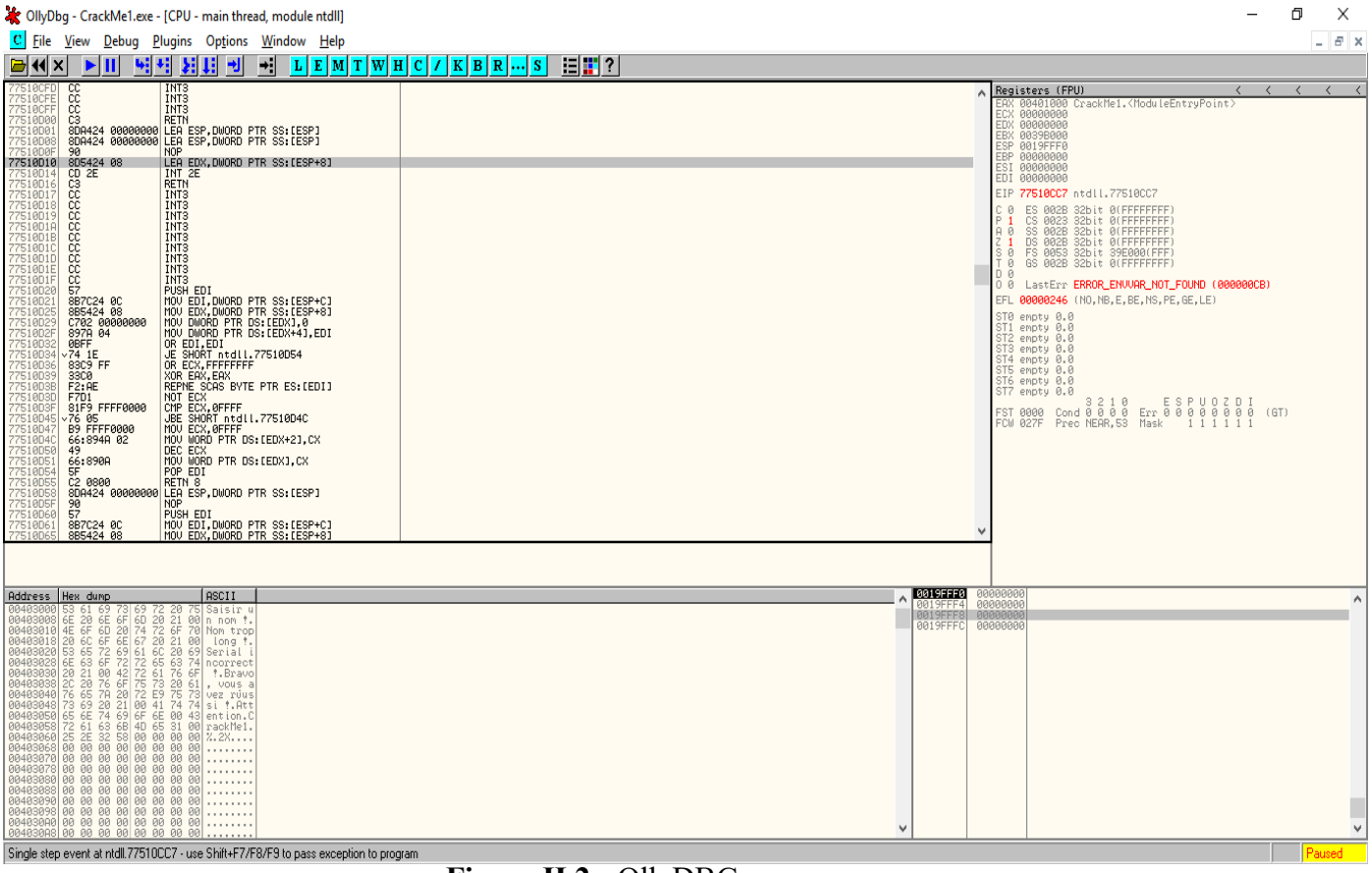


Figure II.2 : OllyDBG

II.4.3. Un Editeur Hexadécimal:

Il permet d'afficher le contenu du programme en hexadécimal et en ASCII. Il permet également de faire les modifications des parties du programme qu'il faut cracker [K].

Exemples: Winhex, Hiew, HexDecCharEditor, … [L]

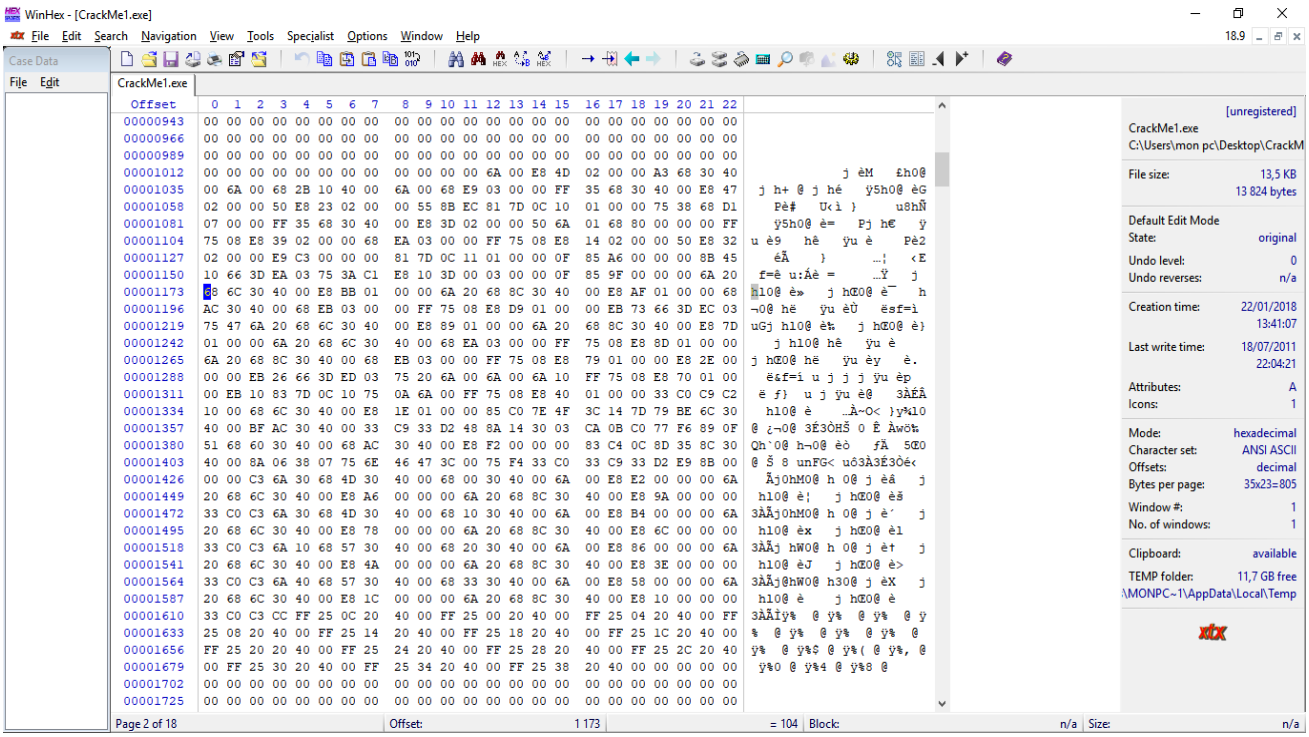


Figure II.3: WinHex

II.4.4. Un Analyseur:

Il permet d'analyser le programme afin de connaitre le langage dans lequel il a été créé ou de connaitre le logiciel utilisé pour le compresser ou le crypter [K].

Exemples: Peid, StudPe, … [L] [M]

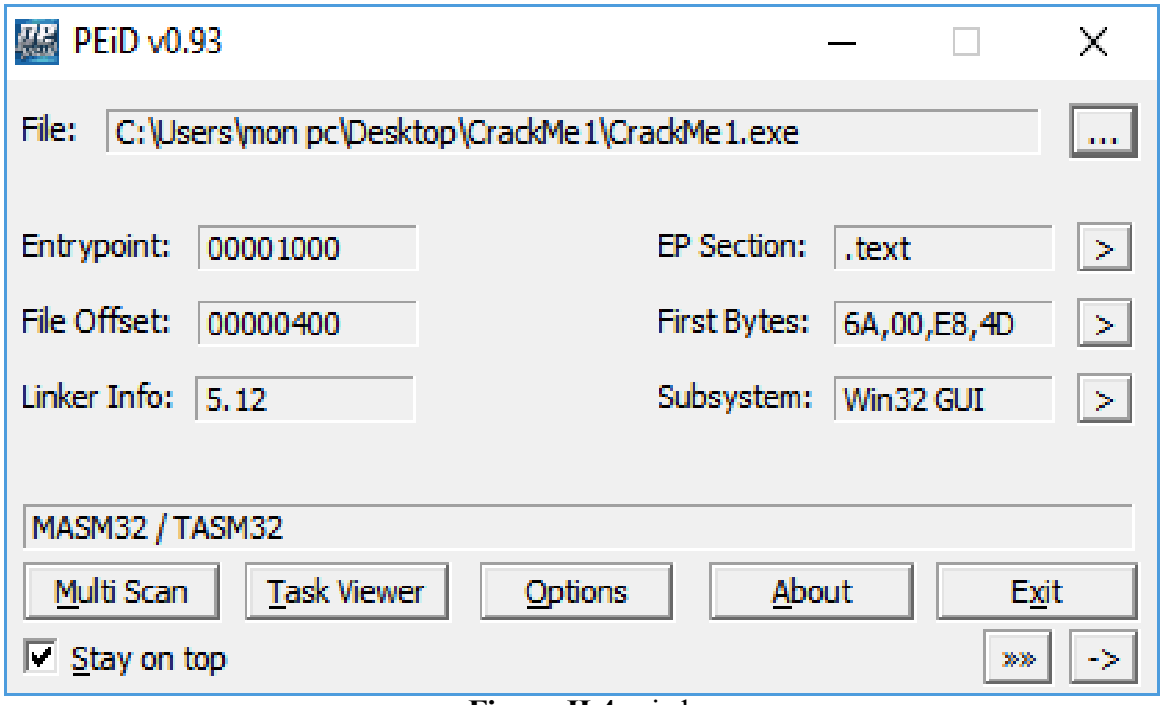


Figure II.4: pied

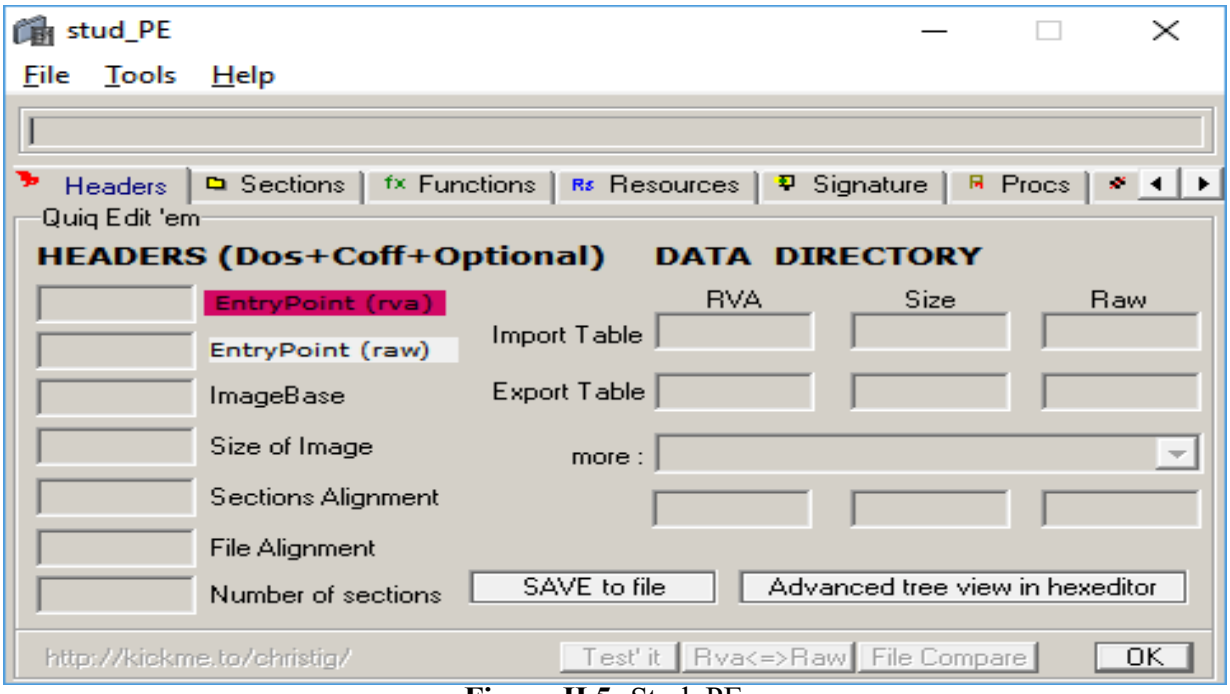


Figure II.5: Stud_PE

II.4.5. Un Patcheur:

Il nous permet de créer un patch (une section de code qu'on ajoute à un logiciel, pour apporter des modifications mineures) pour le programme modifié (cracké). Il compare le fichier original et celui cracké pour ensuite créer un patch adéquat [K].

Exemples : CodeFusion, Graphical-PatchMaker, … [L]

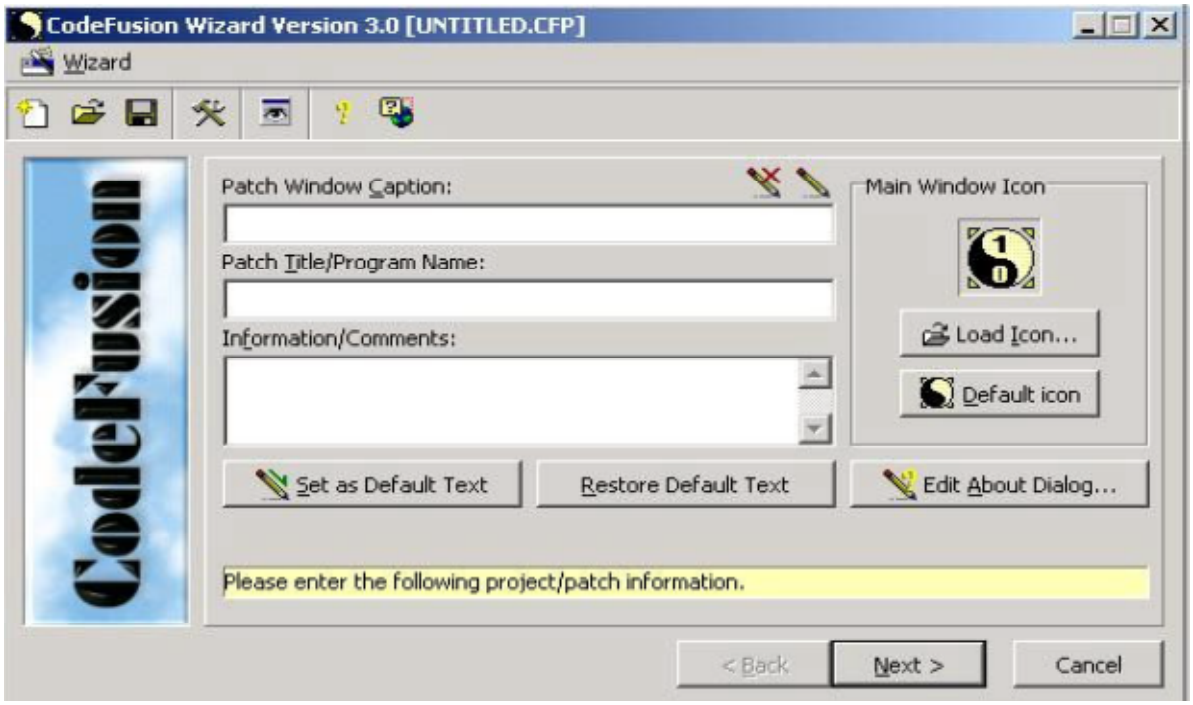


Figure II.6: CodeFusion

II.4.6. Un Unpacker:

L'unpackeur permet de manière automatique de décompresser ou de décrypter les programmes qui le sont. Ainsi, chaque type de compression ou de cryptage demande un programme spécifique [K].

Exemples : UPX, AsPackDie, … [M]

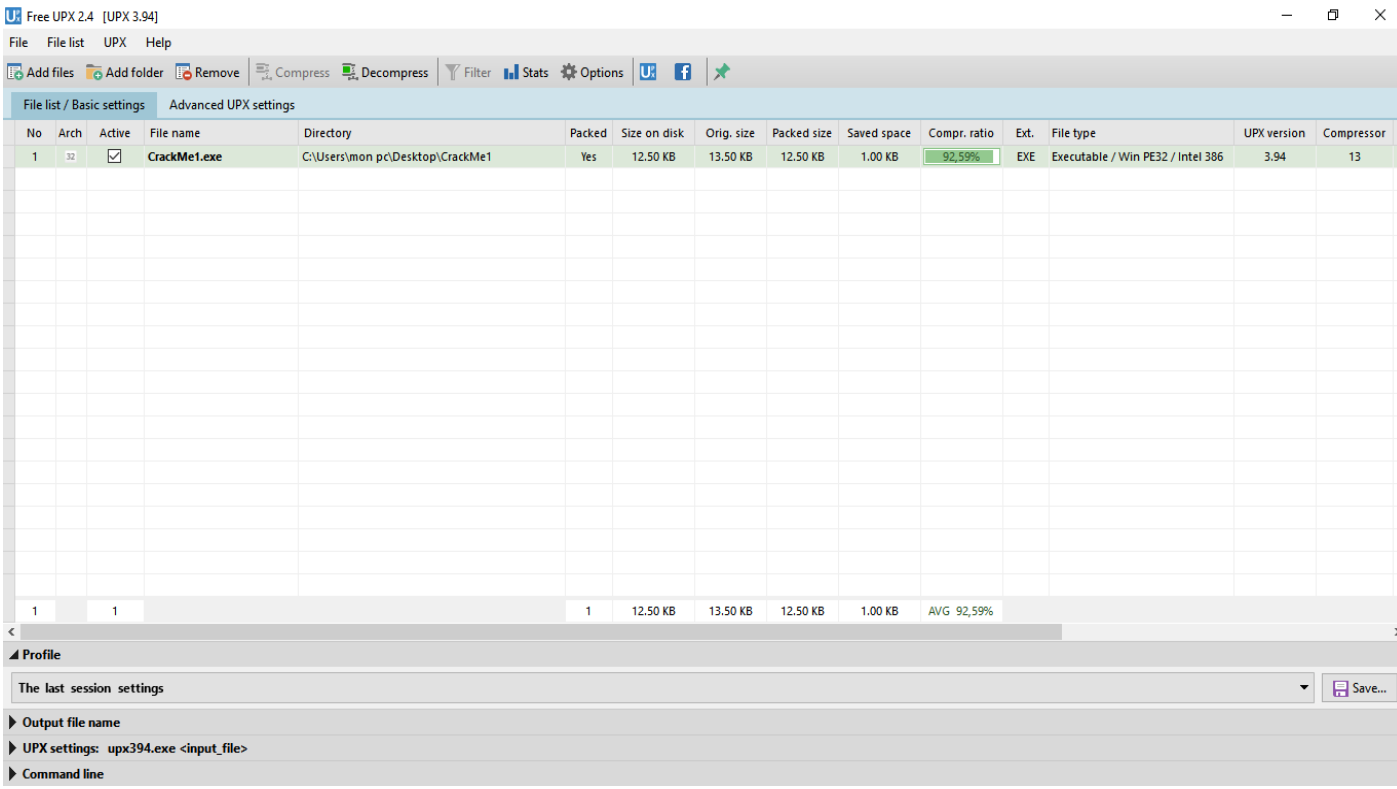


Figure II.7: UPX

II.5. Les bases indispensables pour cracker un logiciel : [E] [N] [O]

Pour cracker n'importe quel logiciel, il est indispensable de connaître le fonctionnement de l'assembleur et ses instructions.

II.5.1. Définition de l'assembleur:

L'assembleur est un langage de programmation transformant un fichier texte contenant des instructions, en un programme que le processeur peut comprendre (programme en langage machine).

Ce langage machine a la particularité d'être difficile à programmer car il n'est composé que de nombres en hexadécimal traduis en instructions. L'assembleur est une "surcouche" du

langage machine, il permet d'utiliser des instructions qui seront transformées en langage machine donc il présente une facilité de programmation bien plus grande que le langage machine. Le fichier texte qui contient ces instructions s'appelle source.

II.5.2. Le langage hexadecimal:

Nous abordons ici un aspect très important de la programmation en assembleur : le système de numérotation en hexadécimal. Ce système est basé sur l'utilisation des chiffres et de certaines lettres de l'alphabet (de A à F).

En assembleur, les nombres décimaux sont suivis d'un "d" (1000=1000d) mais en principe la majorité des assembleurs calculent en décimal par défaut.

La notation hexadécimale implique qu'il faut disposer de 16 niveaux pouvant être alignés dans une représentation et, comme les chiffres ne suffisent plus, on a décidé que les niveaux de 0 à 9 seraient représentés par les chiffres 0..9 et les niveaux manquants pour obtenir 16 niveaux seraient les 6 premières lettres de l'alphabet soit A, B, C, D, E, F avec :

Table1 : Conversion décimal/hexadécimal

DECIMAL	**HEXADECIMAL**
10	A
11	B
12	C
13	D
14	E
15	F

Nous ne pouvons pas utiliser le G et les lettres qui suivent, donc nous augmenterons le premier chiffre ce qui donne 16d=10h. Nous continuons ainsi jusqu'à 255 qui font FF, 256d=0100h (le h indique que ce nombre est en hexadécimal), 257d=101h, 65535=FFFFh. Et ainsi de suite.

II.5.3. Quelques registres généraux:

- **Registres de travail:**

Les quatre (4) registres de travails les plus utilisés sont : AX, BX, CX et DX.

- **AX : accumulateur** -- sert à effectuer des calculs arithmétiques ou à envoyer un paramètre à une interruption.
- **BX : registre auxiliaire de base** -- sert à effectuer des calculs arithmétiques ou bien des calculs sur les adresses.
- **CX : registre auxiliaire (compteur)** -- sert généralement comme compteur dans des boucles.
- **DX : registre auxiliaire de données** -- sert à stocker des données destinées à des fonctions.

Ceci est leur utilisation théorique, mais dans la pratique ils peuvent être utilisés à d'autres usages. Ces registres sont de 16 bits et chacun est divisé en deux parties : L (comme Low pour désigner les bits de poids faible) et H (pour High afin de désigner les bits de poids forts). Par exemple : AX (AL et AH), BX (BL et BH), CX (CL et CH) et DX (DL et DH).

Pour obtenir un registre de 32 bits il faut rajouter un "E" (pour Extended, en français "étendu") devant le registre. Par exemple EAX (32 bits) pour AX (16 bits), (EBX pour BX, ECX pour CX, EDX pour DX). Il existe également des EAH ou EAL, mais la partie haute d'un registre 32 bits ne sera pas directement accessible. Les bits correspondent en fait à leur capacité : 8 bits = 255 (en décimal) au maximum, 16 bits = 65535 au maximum, 32 bits = 4 294 967 295 au maximum.

- **Registres d'offset:**

Ils contiennent une valeur représentant un offset à combiner avec une adresse de segment.

Voici les différents registres :

- **IP : Pointeur d'instruction –** est principalement associé au registre de segment CS (CS : IP) pour indiquer la prochaine instruction à exécuter. Ce registre ne peut jamais être modifié directement; il peut uniquement être modifié indirectement par les instructions de saut, par les sous-programmes et par les interruptions.

- **SI : Index de source --** est utilisé principalement lors d'opérations sur des chaînes de caractères ; il est associé au registre de segment DS.
- **DI : Index de destination --** est utilisé principalement lors d'opérations sur des chaînes de caractères ; il est associé au registre de segment DS ; dans le cas de manipulation de chaînes de caractères, il est associé à ES.
- **SP : Pointeur de pile --** est associé au registre de segment SS (SS:SP) pour indiquer le dernier élément de la pile.
- **BP : Pointeur de base --** est associé au registre de segment SS (SS:BP) pour accéder aux données de la pile lors d'appels de sous-programmes (CALL).

Comme les registres de travail, ces registres peuvent être étendus à 32 bits en ajoutant le "E" : (EIP, ESI, EDI, ESP, EBP). Par contre ils ne possèdent pas de partie 8 bits (il n'y aura pas de EIH ou ESL par exemple). Pour les instructions de chaîne et pour accéder à des blocs en mémoire, les registres SI et DI sont le plus souvent employés.

Ces registres (sauf IP) peuvent être utilisés comme des opérandes dans toutes les opérations arithmétiques et logiques sur 16 bits.

❖ **Registres de segment:**

Pour stocker ou récupérer des données, le processeur utilise des adresses. L'adresse est composée de deux parties de 32 bits (ou 16 bits selon le mode utilisé). La première est le segment et la deuxième est l'offset. L'adresse est présentée comme cela: segment:offset (par exemple : 0A000h:00000h). Voici les différents segments :

- **CS : Registre segment de code --** indique l'adresse du début des instructions d'un programme ou d'une sous-routine.
- **DS : Registre segment de données --** contient l'adresse du début des données du programme. Si le programme utilise plusieurs segments de données, cette valeur devra être modifiée durant son exécution.
- **SS : Registre segment de pile** – se pointe sur une zone appelée la pile.
- **ES : Registre segment supplémentaire (Extra segment) --** est utilisé, par défaut, par certaines instructions de copie de bloc. En dehors de ces instructions, le programmeur est libre de l'utiliser comme il lui convient.
- **FS : Registre segment supplémentaire --** Rôle semblable à ES.
- **GS : Registre segment supplémentaire --** Rôle semblable à ES.

❖ **Registres de flag:**

Il contient des bits qui ont chacun un rôle d'indicateur.

Exemples:

Bit 1 : CF --*Carry Flag* - **Indicateur de retenue**

Bit 2 : 1

Bit 3 : PF --*Parity Flag* - **Indicateur de parité**

Bit 4 : 0

Bit 5 : AF --*Auxiliary carry Flag* - **Indicateur de retenue auxiliaire**

Bit 6 : 0

Bit 7 : ZF --*Zero Flag* - **Indicateur zéro**

Bit 8 : SF --*Sign Flag* - **Indicateur de signe**

Bit 9 : TF

Bit 10 : IF --*Interruption Flag* - **Indicateur d'interruption**

Bit 11 : DF --*Direction Flag* - **Indicateur de direction**

Bit 12 : OF --*Overflow Flag* - **Indicateur de dépassement**

Bit 13 : IOPL

Bit 14 : NT

Bit 15 : 0

Bit 16 : RF

II.5.4. Les premières instructions:

- **MOV**

Cette instruction vient de l'anglais "move" qui signifie DEPLACER mais, le sens de ce terme est modifié car l'instruction MOV ne déplace pas mais place tout simplement une valeur dans un registre.

Cette instruction nécessite deux opérandes (deux variables) qui sont la destination et la source. Ceux-ci peuvent être des registres généraux ou des emplacements mémoire.

Exemple : MOV AX, CX >> Placer le contenu du registre CX dans le registre AX.

MOV AX, 6F5A >> Placer la valeur de 6F5A dans le registre AX.

MOV AX, DS : [SI] >>Placer le contenu de la case mémoire qui se trouve dans le segment DS à l'offset SI, dans le registre AX.

- **LEA**

Pour placer par exemple la valeur de l'offset de MESSAGE dans le registre (SI), il suffit de mettre : MOV SI, Offset-Message. Mais nous ne pouvons pas procéder de cette façon pour les registres DS, ES, FS, GS car ceux-ci n'acceptent pas les valeurs numériques, ils n'acceptent que les registres. Il faudra donc passer par :

MOV AX,Seg-Message puis MOV DS,AX.

Il existe une autre instruction, LEA, qui permet de placer l'offset dans un registre mais en plus court. Par exemple : LEA SI,MESSAGE placera dans SI l'offset de Message. L'utilisation de LEA à la place de MOV rend le code plus clair et plus compact.

- **CALL**

L'instruction CALL permet de faire un appel à un sous programme (sous routine) à un emplacement spécifique dans le programme pour l'exécuter.

Exemple : CALL SI >> Appel la fonction du programme qui se trouve à l'offset contenu dans le registre SI.

CALL 9999 >> Appel la fonction du programme qui se trouve à l'offset 9999.

- **RET**

Retour de sous programme. L'exécution du programme continue à la position récupérée dans la pile. Cette instruction se fait toujours après l'utilisation d'un CALL.

- **CMP**

Cette instruction va servir à tester différentes valeurs et modifier les indicateurs en fonction du résultat. CMP est un SUB qui ne change ni la source ni la destination, seulement les flags. Un CMP BX, CX sera comme un SUB BX, CX à l'exception près que BX ne sera pas modifié.

Exemple : CMP AL, BL >> Comparer la valeur du registre AL à celle de registre BL.

- **JMP**

JMP est une simplification pour exprimer le mot JUMP qui signifie SAUTER en anglais.

JMP va servir dans le programme pour "passer" d'une opération à une autre, de sauter dans le programme à différents endroits pour effectuer d'autres tâches.

Exemple : JMP SI >> Saute à l'offset qui se trouve dans le registre SI.
JMP 9999 >> Saute à l'offset 9999.

II.5.5. Les instructions mathématiques et logiques :

- **instructions mathématiques :**
  - **MUL / IMUL**

L'instruction MUL est utilisé pour la multiplication des nombres non signés et IMUL pour les signés (petit rappel : les nombres signés peuvent être négatifs contrairement aux non signés, par exemple en 16 bits en non signé les nombres vont de 0 à 65535 et en signé ils vont de -32768 à 32767). Contrairement aux opérations précédentes, la multiplication n'a besoin que d'un seul opérande: la source. La destination est donc le nombre qui devra être multiplié se trouvera toujours et obligatoirement dans AX. La source est un registre.

Exemple avec AX = 3 et BX = 5
MUL BX
Résultats : AX = 3*65= 15 et BX = 5

IMUL fonctionne de la même façon, sauf qu'il faut utiliser l'instruction CWD (convert word to doubleword), qui va permettre "d'étendre" le signe de AX dans DX. Si cela n'est pas fait les résultats pourront être erronés.

  - **DIV / IDIV**

Pareil que MUL et IMUL, DIV sert pour faire la division des nombres non signés et IDIV pour les nombres signés. Cette instruction divise la valeur de AX par la source.
Exemple avec AX = 20 et BX = 5 :
IDIV BX
Résultats : AX = 4 et DX = 0 (DX étant le reste)

  - **ADD et SUB**

ADD sert à additionner, et nécessite deux opérandes : une source et une destination. La destination prendra la valeur de source plus destination. De même pour le SUB qui fait la

soustraction. Les règles de combinaison sont les mêmes que pour MOV mais avec SUB et ADD, l'utilisation des registres de segment est impossible.

Exemples :
ADD AX, CX (si AX=11 et CX=22 alors nous aurons AX=33 et CX=22)
ADD AX, BX
ADD AX,-123 (l'assembleur autorise les nombres négatifs)
SUB DI, 12

- **SHR et SHL**

Ces instructions permettent de diviser des registres avec SHR et les multiplier avec SHL. Ils demandent deux opérandes : le registre à diviser et le diviseur pour SHR et le registre à multiplier et le multiplicateur pour SHL. Ils fonctionnent en puissance de 2. C'est instructions sont utilisées car elles sont plus rapides en terme d'exécution que les instructions **MULL** et **DIV**.

Exemple : Si nous voulons diviser AX par 4 nous mettrons: SHR AX, 2 (car 2^2 = 4). Pour diviser BX par 16 nous mettrons: SHR BX,4 (car 2^4 = 16). Pour SHL c'est pareil : pour multiplier CX par 256 nous mettrons donc : SHL CX,8 (2^8 = 256).

Pour multiplier ou diviser des nombres qui ne sont pas des puissances de 2 il faudra combiner des SHR ou SHL.

Exemple : Si nous voulons multiplier 5 par 384. Nous voyons que 384 = 256 + 128, qui sont des puissances de 2.

MOV AX,5 >> met 5 dans AX >> AX = 5

MOV BX,AX >> met AX dans BX >> BX = AX

SHL AX,8 >> multiplie AX par 256 (2^8) >> AX = AX*256 = 5*256= 1280

SHL BX,7 >> multiplie BX par 128 (2^7) >> BX = BX*128 =5*128 = 640

ADD AX,BX >> ajoute BX à AX >> AX = AX + BX = 1280+ 640 = 1920

Dans d'autres cas, il est préférable d'utiliser les instructions MUL et DIV pour rendre le programme compact.

- **INC / DEC**

Ces instructions permettent d'incrémenter et de décrémenter la valeur d'un registre.

Exemple : INC AL va faire (AL+1)
DEC AL va faire (AL-1)

- **NEG**

NEG sert à rendre un nombre positif en négatif et inversement. Il ne demande qu'un seul opérande : la destination. Si AX = 10, alors un NEG AX mettra AX = -10. Ou si BX = -14 alors après le NEG BX, nous aurons BX = 14.

❖ **Instructions logiques :**

- **AND** fait un ET logique entre deux opérandes.

Exemple avec AX = 15 et BX = 26 :
AND AX,BX
0000 1111 (AX=15)
AND 0001 1010 (BX=26)
Résultat : 0000 1010 (AX = 10)

- **OR** fait un OU logique entre deux opérandes.

Exemple avec AX = 15 et BX = 26 :
OR AX,BX
0000 1111 (AX=15)
OR 0001 1010 (BX=26)
Résultat : 0001 1111 (AX = 31)

- **XOR** fait un OU exclusif logique entre deux opérandes.

Exemple avec AX = 15 et BX = 26 :
XOR AX,BX
0000 1111 (AX=15)
XOR 0001 1010 (BX=26)
Résultat : 0001 0101 (AX = 21)

- **NOT** fait un complément à 1 de l'opérande.

Exemple avec AX = 15 :
NOT AX = NOT 0000 1111 (AX=15) = 1111 0000 (AX = -16 en signé et 240 en non signé)

- **TEST** Cette instruction teste la valeur d'un ou plusieurs bits en effectuant un ET logique sur les opérandes. Le résultat ne modifie pas les opérandes mais plutôt les indicateurs.

Exemple :
TEST AX,1 >> Effectue un ET LOGIQUE sur le premier bit de AX avec 1, si il est égal à 1, ZF passera à 1 et inversement (si 0 alors ZF=0).

II.5.6. Les sauts conditionnels:

Nous abordons une partie qui est nécessaire lors de la création d'un programme. Souvent, le programme doit faire une action selon le résultat d'une comparaison. Il se base sur les indicateurs pour faire le choix.

Comme l'instruction JMP, s'agissant d'un simple saut vers une autre partie du programme. D'autres instructions comme JMP font des sauts mais selon certains critères, nous les appelons des sauts conditionnels. En voici la liste des ces instructions ci-dessous.

JB - JNAE – JC >> Jump if Below - Not Above or Equal – Carry (Saute si inférieur).
JAE - JNB – JNC >>Jump if Above or Equal - Not Below - Not Carry (Saute si supérieur ou égale).
JE – JZ >> Jump if Equal – Zero (Saute si égale).
JNE – JNZ >> Jump if Not Equal - Not Zero (Saute si inégale).
JO – JNO >> Jump if Overflow - Not Overflow (Saute si débordement).
JP – JPE >> Jump if Parity - Parity Even (Saute si paire).
JNP – JPO >> Jump if No Parity - Parity Odd (Saute si impaire).
JS – JNS >> Jump if Signed - Not Signed (Saute si signé).
JA – JNBE >> Jump if Above - Not Below or Equal (Saute si supérieur).
JBE – JNA >> Jump if Below or Equal - Not Above (Saute si inférieur ou égale).
JG – JNLE >> Jump if Greater - Not Less or Equal (Saute si supérieur).
JGE – JNL >> Jump if Greater or Equal - Not Less (Saute si supérieur ou égale).
JL – JNGE >> Jump if Less - Not Greater or Equal (Saute si inférieur).

JLE – JNG >> Jump if Less or Equal - Not Greater (Saute si inférieur ou égale).

L'utilisation de ces différents sauts conditionnels nécessite toujours l'utilisation d'une instruction de comparaison en amont.

II.5.7. Pile :

La pile (stack en anglais) est un emplacement temporaire où des données de petites tailles peuvent être placées. Cette mémoire est utilisée aussi dans les calculatrices et dans tous les ordinateurs. Le système qu'elle emploie pour stocker les données est du principe du "dernier stocké, premier à sortir" (LIFO-last in, first out). Cela veut dire que lorsqu'on place une valeur (un registre, un nombre) dans la pile, elle est placée au premier rang (placée en priorité par rapport aux autres valeurs dans la pile). Lors de la lecture de la pile pour récupérer la valeur, ce sera la première qui sera prise.

Il y a deux instructions principales à savoir :

- **PUSH**

Nous avons tout d'abord l'instruction PUSH qui signifie POUSSER. Cette instruction permet de placer une valeur au sommet de la pile. PUSH doit être accompagné d'une valeur de 16 ou 32 bits (souvent un registre) ou d'une valeur immédiate.

Exemple: PUSH AX
PUSH BX

En premier lieu, AX est placé en haut de la pile et ensuite BX est placé au sommet, AX est repoussé au deuxième rang.

- **POP**

Passons maintenant à l'instruction de "récupération" qui se nomme POP (sortir-tirer).Cette instruction demande comme PUSH, une valeur de 16 ou 32 bits. Elle prend la première valeur de la pile et la place dans le registre qui suit l'instruction.

Exemple:

Nous avions PUSH AX et PUSH BX et maintenant nous allons les sortir de la pile en utilisant POP BX et POP AX.

II.6. Discussions :

Le cracking peut être illégal suivant l'utilisation que nous en faisons. Il est illégal, et par conséquent punissable par la Loi de, par exemple, modifier un programme protégé par des droits d'auteur, ou un logiciel payant. Par contre il est tout à fait possible d'exercer légalement le cracking sur nos propres logiciels pour tester sa résistance face aux pirates informatiques et aussi sur des logiciels dont la provenance n'est pas sûre (internet par exemple) pour s'assurer de la non présence de programmes malveillants dans le code source avant l'installation du logiciel.

L'ensemble des outils et techniques que nous avons présenté dans ce chapitre, sont les plus utilisés actuellement dans le cracking. Mais, la liste n'est tout de même pas exhaustive, car on pourrait créer d'autres programmes qui soient plus efficace ou un programme qui englobe l'ensemble de ces fonctions réunis, si ce n'est pas déjà en cours, puisque la technologie est en perpétuelle évolution.

En somme, le travail sur ce chapitre nous a permis d'acquérir plus de connaissances en programmation assembleur et de comprendre le fonctionnement des logiciels et leurs vulnérabilités. Dans le chapitre suivant, nous mettons en application nos connaissances acquises sur les différentes techniques de cracking.

Chapitre III: Mise en application des techniques de cracking

III.1. Préambule :

Dans ce chapitre, nous avons choisi des logiciels conçus à titre éducatif appelés « **crackmes** » afin de mettre en application nos connaissances acquises lors de ce travail sur les différentes techniques de cracking. Pour ce faire, nous avons d'abord commencé par la technique **patching**, ensuite la technique **serial fishing** et pour terminer avec la technique **keygenning** qui est la plus importante.

III.2. Le patching :

Après que nous ayons expliqué le processus de fonctionnement, maintenant crackons notre premier logiciel crack-me.

Tout d'abord nous ouvrons le crack-me et il nous demande d'introduire un identifiant et un mot de passe. Nous introduisons un sérial quelconque, ensuite nous cliquons sur Ok et il nous affiche le message d'erreur comme quoi le sérial est incorrect (voir figureIII.1).

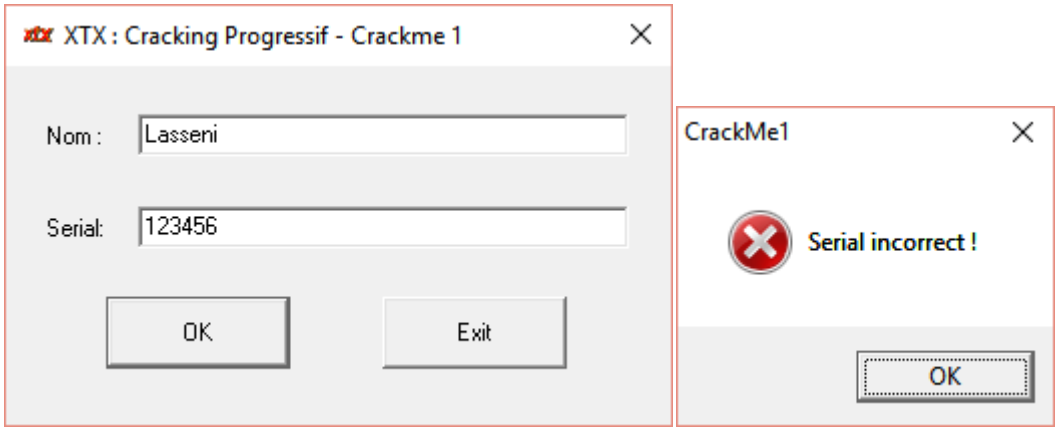


Figure III.1 : Le logiciel à cracker par patching

Cela est dû au système de protection qui est présent dans la plupart des logiciels ou systèmes informatiques.

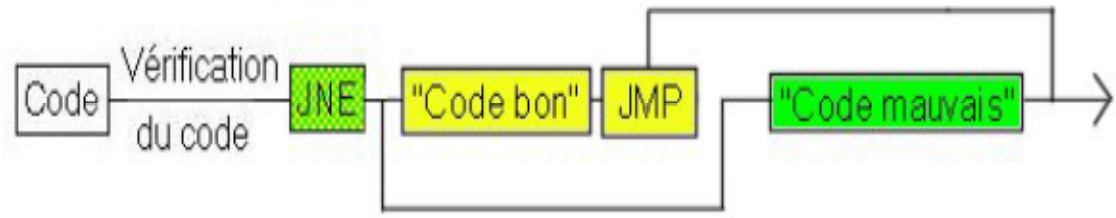


Figure III.2 : Schéma de protection d'un système informatique

Une fois le mot de passe rentré, le système le compare avec le bon code, si ce n'est pas le même il saute pour arriver vers le mauvais message (dans notre cas : Serial incorrect !) sinon il affiche le bon message et continue son exécution en passant au dessus du mauvais message comme nous pouvons le remarquer sur la figure III.2.

JNE est un saut conditionnel. Il saute si le code est faux dans ce cas là (saut si inégal) ou continue sa route si le code est bon.

JMP (= Jump = Saut) est aussi un saut mais inconditionnel puisqu'il saute obligatoirement quelque soit le résultat de la comparaison.

Et ça donne à peu près ça en assembleur:

:00401001 #### Teste si le code est bon
:00401002 75## jne 00401005 saute sinon
Le JNE saute uniquement si le code n'est pas bon sinon il continu avec l'instruction suivante.
* Possible StringData Ref from Data Obj -> **"Le code est bon"**
*Windasm indique que **"Le code est bon"** est le message d'une boîte de dialogue.*
:00401003 EB## jmp 00401005
Le JMP est un saut. Le numéro à sa droite est l'adresse où il va sauter.
* Referenced by a (U)nconditional or (C)onditional Jump at Address:
|:00401002(U)
Il indique que la portion de code suivante est appelée par un saut conditionnel (l'adresse de celle-ci est à gauche du "(U)" : 00401002)
:00401004 #### push ####
* Possible StringData Ref from Data Obj -> **"Le code n'est pas bon"**
:00401005 #### ####

Maintenant, pour pouvoir cracker ce logiciel avec la technique patching, nous avons deux possibilités :

- Soit nous modifions l'instruction de comparaison pour qu'il ne vérifie pas le mot de passe rentré.
- Soit nous modifions l'instruction du saut conditionnel pour qu'il laisse entrer même si le mot de passe est faux.

III.2.1. Modification du saut conditionnel :

Pour cela, nous avons besoin d'un désassembleur (WinDasm) et d'un éditeur haxdécimal (Winhex).

- ✓ WinDasm:

C'est un désassembleur/débuggeur (au départ, nous nous intéressons uniquement au mode désassemblage). Il sert à voir les instructions d'un exécutable (*.exe en général) ou d'une (*.dll). Après l'avoir téléchargé et lancé, nous réglons d'abord quelques fonctions en cliquant sur "Disassembler" >>> "Font..." >>> "Select Font" pour choisir la police que nous voulons (ici : @Arial Unicode MS) puis sur "Ok". Puis nous cliquons de nouveau sur "Disassembler" >>> "Font..." >>> "Save Default Font". Ensuite, nous cliquons sur "Disassembler" >>> "Disassembler Options" pour cocher les 3 cases. Maintenant nous pouvons ouvrir n'importe quel fichier exécutable avec ce WinDasm, mais avant, nous devons connaitre l'utilité des différents boutons.

Figure III.3: L'usage des boutons de WinDasm.

1° Ouvrir un exécutable pour le désassembler.

2° Sauvegarder les pages du programme désassemblé, ce qui permet de le reouvrir beaucoup plus rapidement la fois suivante. Pour l'ouvrir on clique sur "Project" >>> "Open Project File...".

3° Rechercher du texte.

4° Permet de copier le texte d'une ligne, mais il faut d'abord cliquer à gauche de la ligne afin d'obtenir un point rouge. Utiliser la touche MAJ pour sélectionner plusieurs lignes à la fois.

```
:00405BCD 8B4E28                mov ecx, dword ptr [esi+28]
:00405BD0 8BE8                  mov ebp, eax
:00405BD2 8B4624                mov eax, dword ptr [esi+24]
```

5° Aller directement au début du code (il y a, au-dessus, des informations qui n'en font pas partie)

6° Aller au point d'entrée du programme.

7° Aller à la page choisie (le numéro de page étant inscrit tout en bas à gauche).

8° Aller à l'offset (l'adresse) choisi.

9° Permet d'aller là où le saut "atterrit".

10° Permet, après avoir utilisé le 9°, de revenir au saut.

11° Permet de rentrer dans le CALL.

12° Permet, après avoir utilisé le 11°, de sortir du CALL.

13° Lister et trouver les fonctions et modules importés (les APIs) du fichier désassemblé.

14° Lister et trouver les fonctions exportées.

15° Permet de voir les données du fichier en hexa et ascii.

16° Permet de voir les données de la section code du fichier affichée.

17° Permet de trouver les items des menus (ex : "copier", "coller", "nouveau"...).

18° Permet de trouver les dialogues du programme.

19° Un des plus importants. Il permet de trouver les Strings Data Références c'est à dire les messages des boîtes de dialogues (ex : "Code invalide")

20° Permet d'imprimer des pages.

Tout d'abord quand nous ouvrons un programme avec WinDasm, nous nous apercevons des lignes semblables à celles-ci :

```
: 0040100E   E839000000      Call 0040104C
:00401013    83F807            cmp eax, 00000007
:00401016    EB1A               jmp 00401032
:00401018    6A40               push 00000040
```

Le numéro en première colonne est l'adresse de la ligne de l'instruction.

Le nombre en deuxième colonne est l'instruction en hexadécimal correspondant aux instructions en assembleur sur la troisième colonne3.

Les adresses de WinDasm sont en hexadécimal et dans l'ordre croissant, mais il ne les met pas toutes comme nous pouvons le voir dans l'exemple ci-dessus.

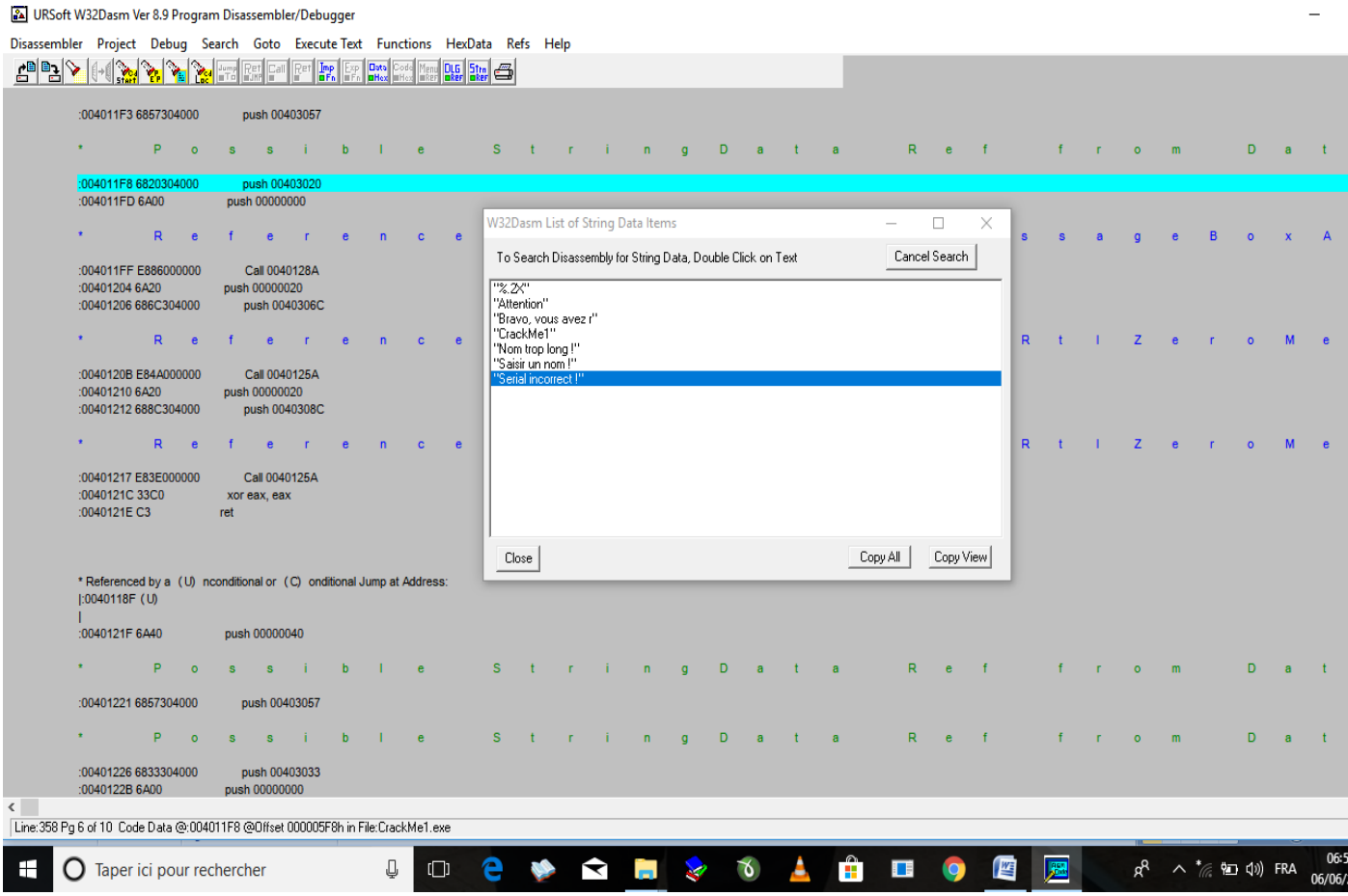
- ✓ Winhex :

C'est un outil avec lequel nous pouvons modifier des octets de n'importe quels fichiers. C'est pour ça qu'il nous faut un éditeur hexadécimal et WinDasm pour cracker. Ce dernier sert à voir ce qu'il faut modifier et où, et l'éditeur permet d'effectuer les modifications.

Figure III.4: Les boutons de WinHex.

Nous pouvons maintenant cracker le logiciel (crackme) en l'ouvrant avec WinDasm pour le désassembler et ensuite cliquons sur le bouton « string data référence (19) » pour chercher le message d'erreur, une fois trouver, double cliquons dessus et il nous indique directement la ligne correspondante dans le programme désassemblé (voir figureIII.5).

Figure III.5: Le programme désassemblé sous WinDasm



Nous voyons sur l'écran la ligne qui corresponde au message d'erreur. Nous remontons un peu jusqu'à la ligne suivante :

« *Referenced by a (U) nconditional or (C) onditional Jump at Address : | :00401181 (C) »

La valeur (00401181) indique l'adresse du saut qui atterrit au message d'erreur. Dans ce cas, il nous suffit de retrouver la ligne correspondante à cette adresse. Pour cela, cliquons sur le bouton «Goto code location (8) » en y insérant l'adresse puis sur OK. Après cela, il nous conduit sur la ligne du saut conditionnel (voir figure III.6).

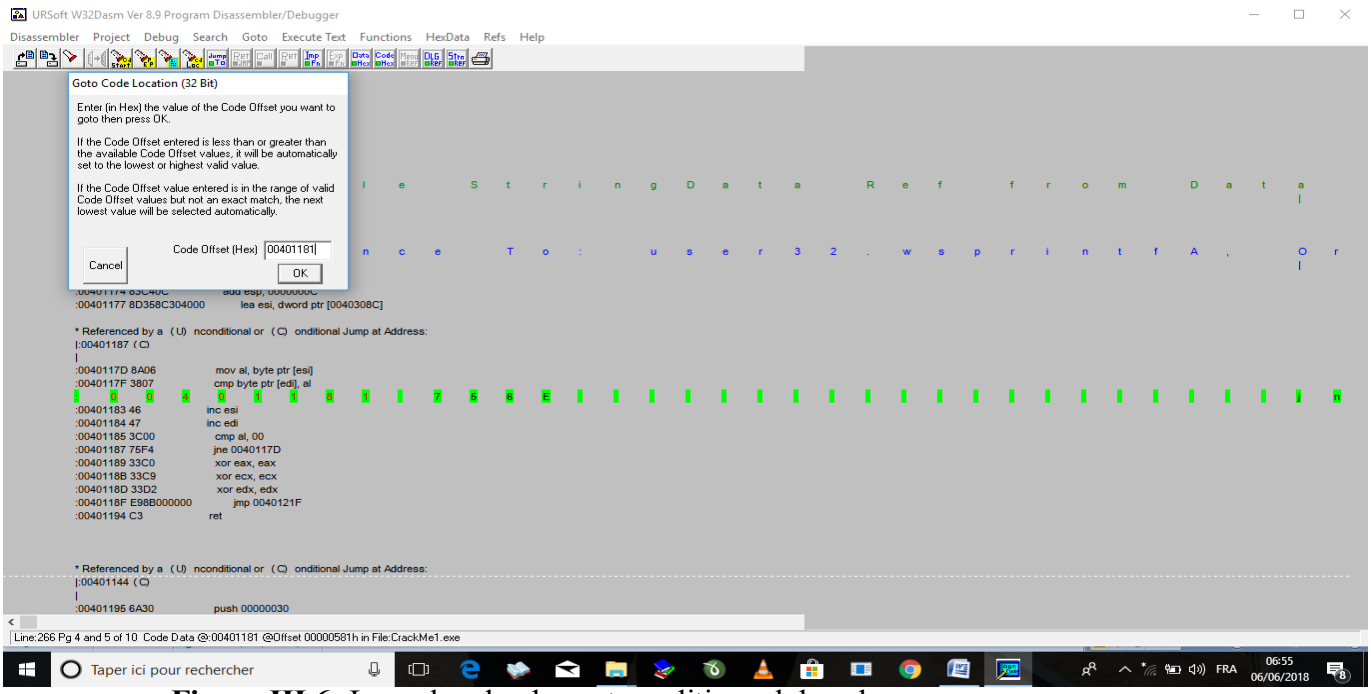


Figure III.6: La recherche du saut conditionnel dans le programme

Et directement nous relevons l'offset de la ligne en bas de l'écran qu'est (00000581h) puis nous fermons complètement WinDasm et ensuite ouvrons le crack-me dans l'éditeur hexadécimal (WinHex).

Ensuite nous rentrons dans le menu « Navigation » >>> « Aller à la position » puis y introduire la valeur de l'offset(en décimal) dans la fenêtre qui s'affiche puis sur OK, et il nous conduit directement à la partie correspondante du code à modifier (voir figure III.7).

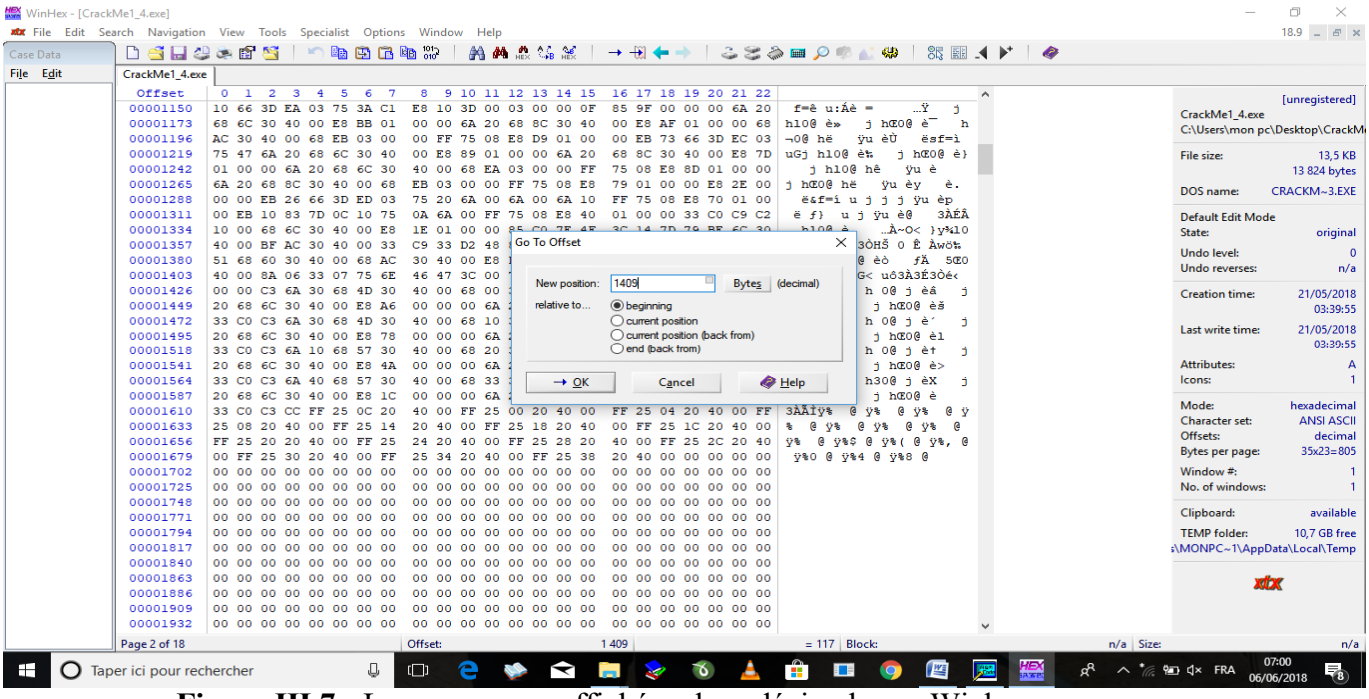


Figure III.7 : Le programme affiché en hexadécimal sous Winhex

L'éditeur hexadécimal permet d'afficher les instructions assembleur en code hexadécimal que nous pouvons modifier. Ainsi, chaque instruction du langage assembleur a une valeur équivalente en hexadécimal. En voici quelques unes :

Table2: Assembleur-Hexadécimal

Instruction	**Valeur hexa**	**Description**
jne	75 ou 0F85	*Saut si inégal*
je	74 ou 0F84	*Saut si égal*
jmp	EB ou E9	*Saut*
nop	90	*Pas d'opération*
test	85	*Comparaison logique*
cmp	39	*Comparaison arithmetique*
xor	33	*Ou exclusif logique*

Pour information les 75, 74, EB sont des sauts courts et 0F85, 0F84, E9 des sauts longs. Après les sauts courts nous avons le nombre (en hexadécimale) d'octets sautés (allant de 00 à FF et donc pour les sauts de plus de FF octets nous utilisons des sauts longs à la suite desquelles nous mettons l'adresse de destination. Par exemple EB02 représente un saut (JMP) de 2 octets).

Ensuite nous remplaçons les valeurs 756E (saut si inégale) que nous voyons sur l'écran par 746E (saut si égale) ou par 9090 (pas d'opération) puis enregistrer sous un nom différent avec l'extension (*.exe) comme nous pouvons le voir sur les figure III.8 et figure III.9.

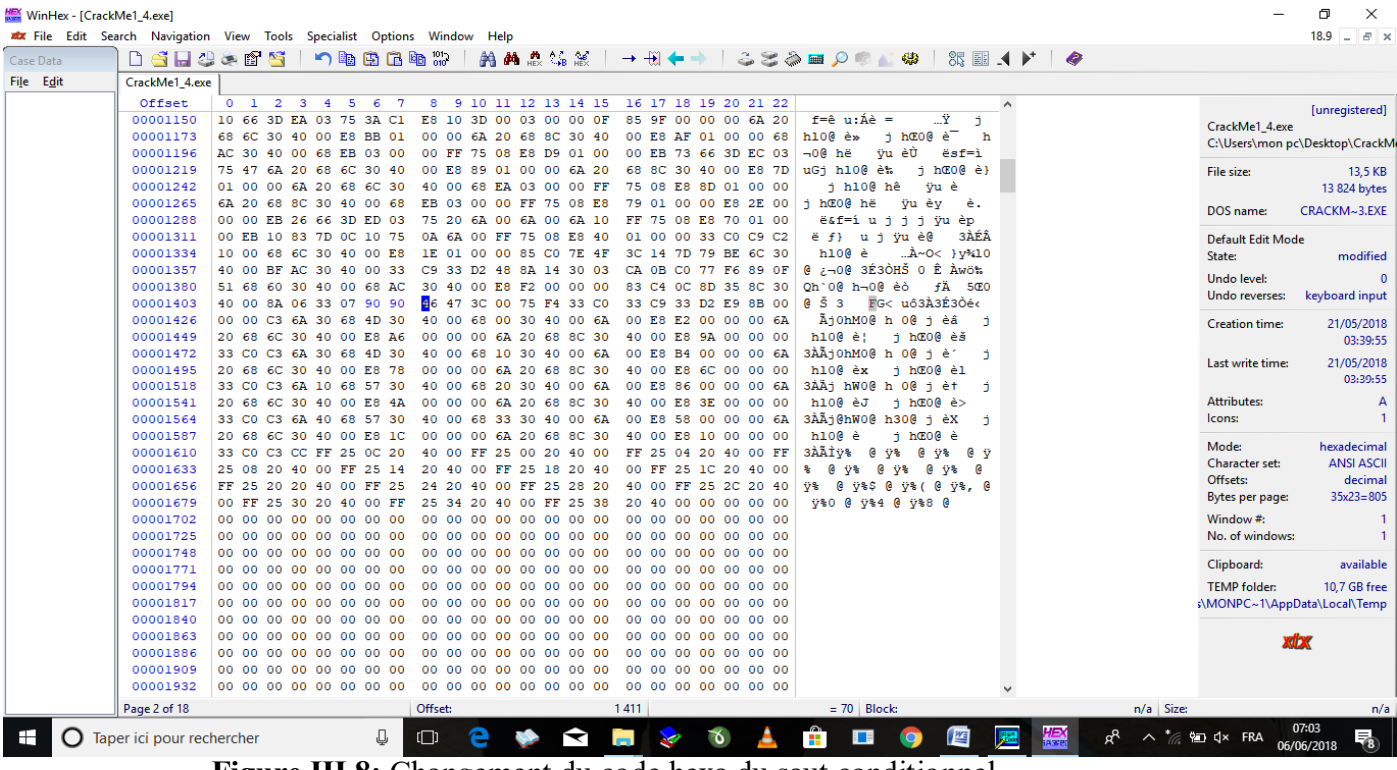


Figure III.8: Changement du code hexa du saut conditionnel

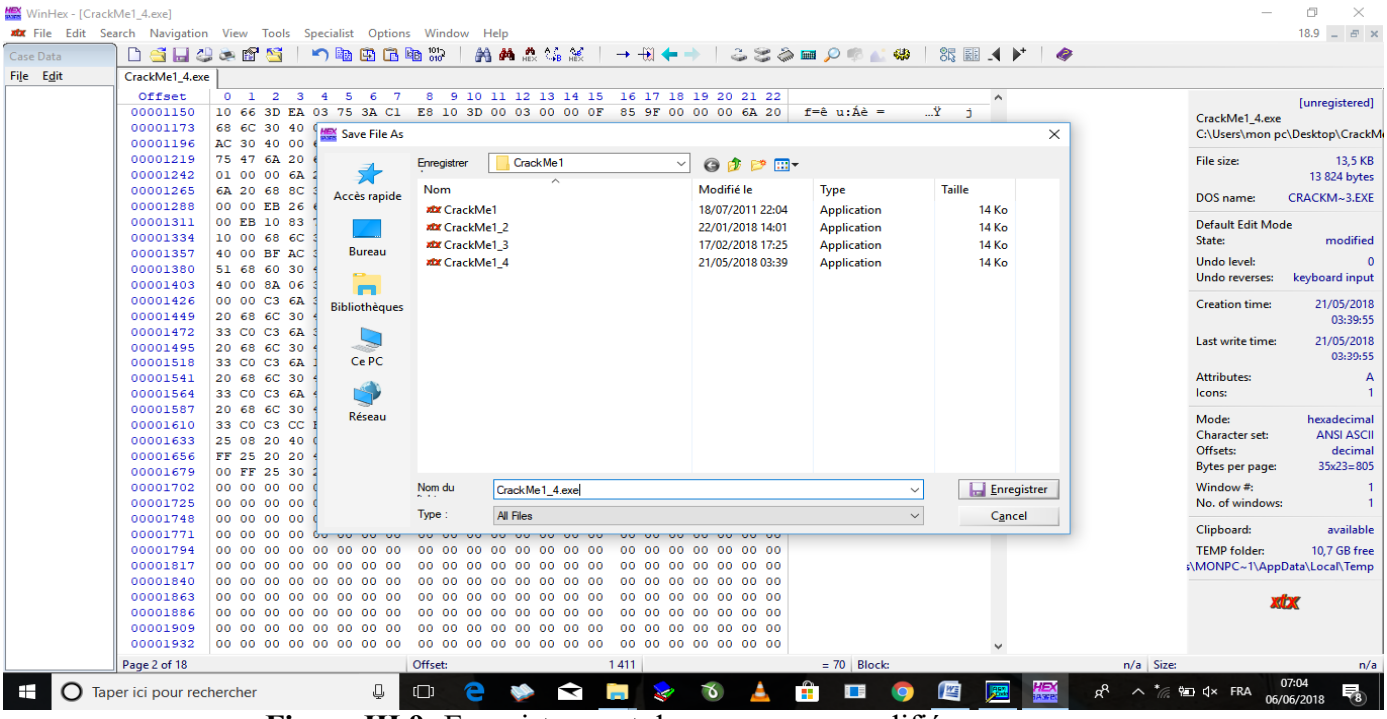


Figure III.9: Enregistrement du programme modifié

Maintenant ça y est le crack-me est cracké et nous pouvons accéder au logiciel avec n'importe quel mot de passe, comme nous pouvons le voir sur la figure III.10.

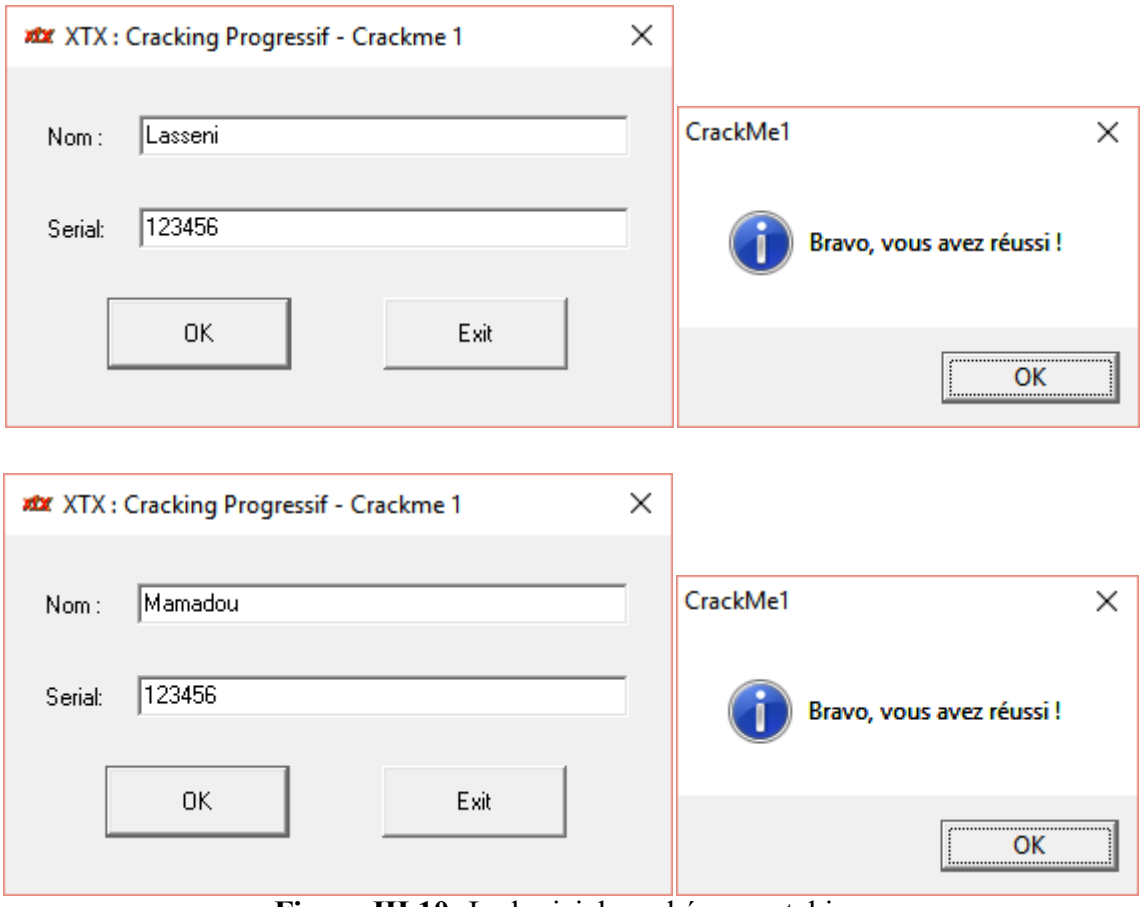


Figure III.10: Le logiciel cracké par patching

Il nous laisse entrer sans vérifier le sérial, ce qui est bien normal, car nous avons remplacé l'instruction JNE (saut si inégal) par NOP (aucune opération). De ce fait, le logiciel laisse entrer quelque soit le résultat de la vérification.

III.2.2. La modification de l'instruction de comparaison :

D'abord en assembleur nous avons deux types d'instruction de comparaison à savoir :

- L'instruction **CMP** qui fait une comparaison entre deux valeurs en faisant la soustraction de celles-ci et ensuite juger en fonction du résultat. Pour ce type de comparaison, il est préférable de procéder à la modification du saut conditionnel.
- L'instruction **TEST** qui fait une comparaison entre deux valeurs en faisant un **ET** logique de celles-ci. Pour ce type de comparaison, il nous suffit de remplacer l'instruction **TEST** (85 en hexadécimal) par l'instruction **XOR** (33 en hexadécimal) pour le que logiciel nous laisse entrer sans faire de comparaison.

III.3. Le serial fishing:

Comme nous l'avions expliqué dans le chapitre précédent, cette technique consiste à chercher le vrai mot de passe dans le programme du logiciel sans modifier quoi que ce soit dans ce dernier. Il faut d'abord savoir que le mot de passe de tout système informatique (logiciel, système d'exploitation, site web, etc.) est caché quelque part dans son programme. Ainsi, une fois qu'on introduit un mot de passe, le programme fait appel au vrai mot de passe qui se trouve déjà dans le programme et le compare avec celui rentré. Si les deux mots de passe sont les mêmes, le système donne l'accès, sinon il rejette l'accès (voire Figure III.2: Schéma de protection d'un système informatique).

Dans cette partie, nous avons cracké un logiciel (Crackme) en cherchant son mot de passe caché dans son programme.

Tout d'abord nous essayons de nous introduire dans le logiciel en entrant un mot de passe quelconque et le logiciel nous refuse l'accès (comme nous pouvons le voir sur la figure III.11).

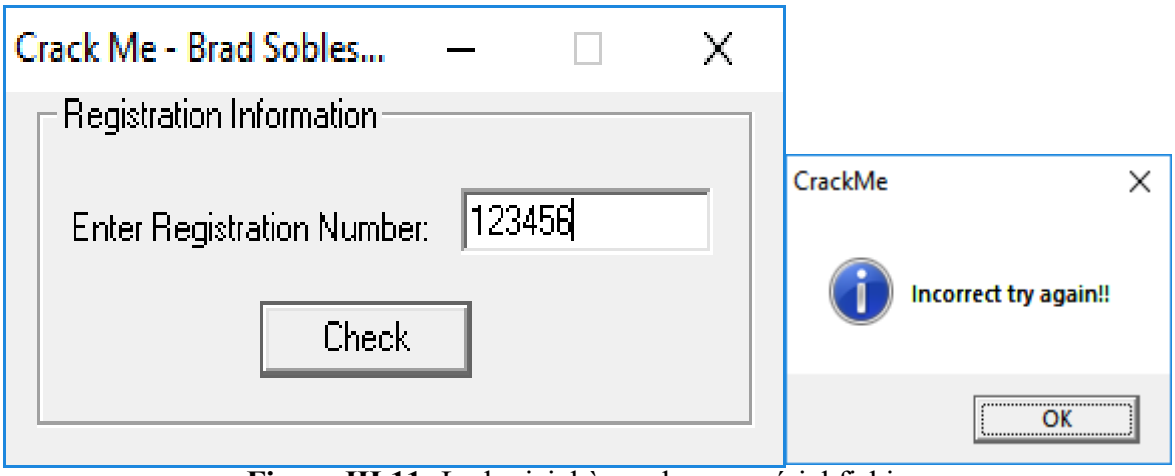


Figure III.11: Le logiciel à cracker par sérial fishing

Nous affichons maintenant le code source du logiciel en assembleur à l'aide du logiciel (WinDasm) pour retrouver le mot de passe. Ensuite cliquons sur le bouton « string data référence (19) » pour chercher le message d'erreur, une fois trouver, double cliquons dessus et il nous indique directement la ligne correspondante dans le programme désassemblé (voir figure III.12).

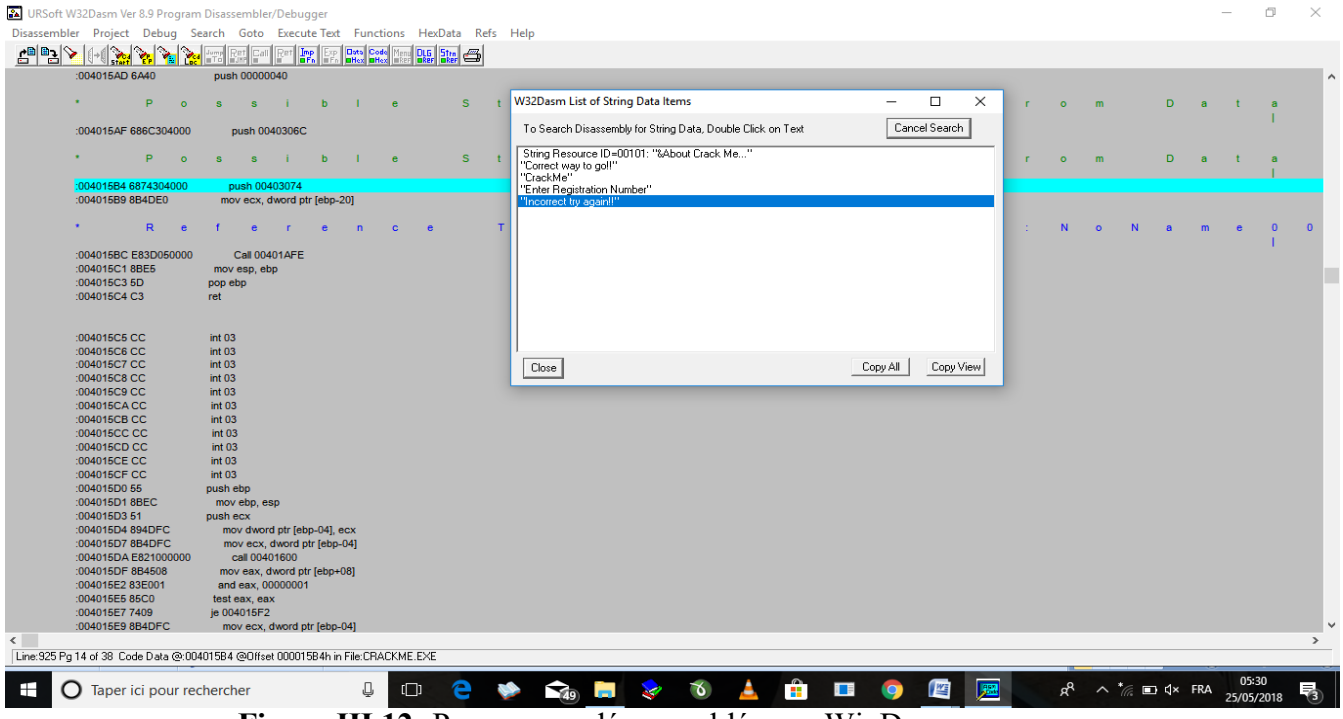


Figure III.12: Programme désassemblé avec WinDasm

Nous voyons sur l'écran la ligne qui corresponde au message d'erreur. Nous remontons un peu jusqu'à la ligne suivante :

« *Referenced by a (U) nconditional or (C) onditional Jump at Address: |:00401595 (C) »

La valeur (00401595) indique l'adresse du saut qui atterrit au message d'erreur. Dans ce cas, il nous suffit de retrouver la ligne correspondante à cette adresse. Pour cela, cliquons sur le bouton «Goto code location (8) » en y insérant l'adresse puis sur OK. Après cela, il nous conduit sur la ligne du saut conditionnel (voir figure III.13).

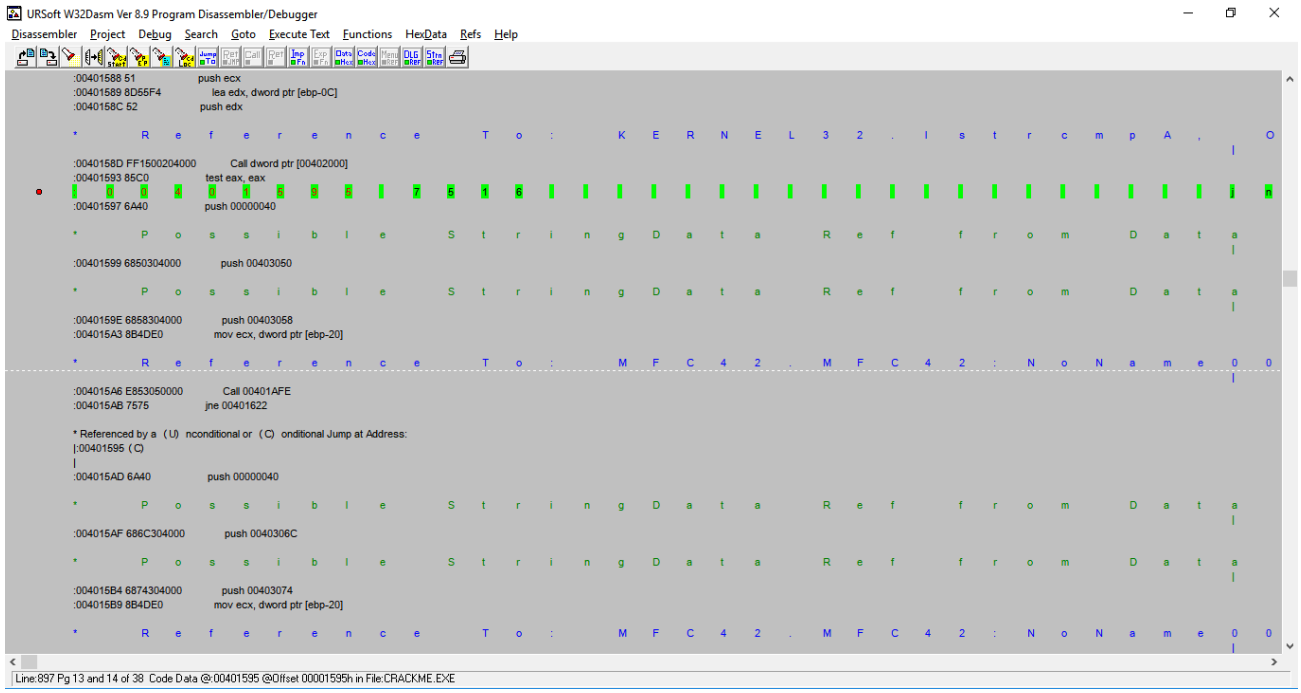


Figure III.13: La recherche du saut conditionnel dans le programme

Nous tombons sur les trois lignes suivantes :

:0040158D FF1500204000 Call dword ptr [00402000]

:00401593 85C0 test eax, eax

:00401595 7516 jne 004015AD

La première ligne indique l'instruction CALL qui fait appel au mot de passe du programme.

La deuxième ligne indique l'instruction TEST qui fait la comparaison entre le mot de passe appelé et le mot de passe rentré.

La troisième ligne indique l'instruction JNE qui va sauter vers le message d'erreur si les mots de passe comparés ne sont pas les même. C'est pourquoi nous somme tombés sur (**Incorrect try again!!**) à l'adresse 004015AD.

Maintenant, nous devons poser un **break point** sur l'instruction CALL pour intercepter le mot de passe lorsque cette dernière lui fait appel. Pour cela, nous utilisons la fonction

débuggeur de WinDasm en faisant (Ctrl+l) puis en cliquant sur **Load**. Et là il nous affiche trois fenêtres :

1- La première corresponde au programme désassemblé dans laquelle nous devons sélectionner la ligne de l'instruction **CALL** et poser un break point en faisant (Ctrl+F2) après avoir cocher les cinq (5) cases de la dernière fenêtre.

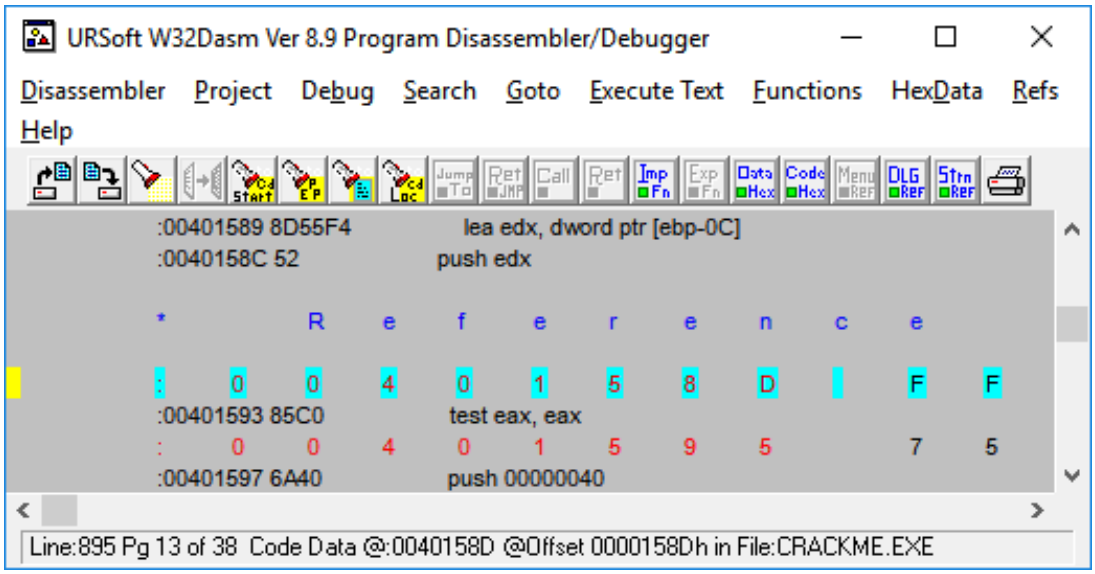


Figure III.14: Le break point sous WinDasm

2- La seconde corresponde aux valeurs des registres du programme.

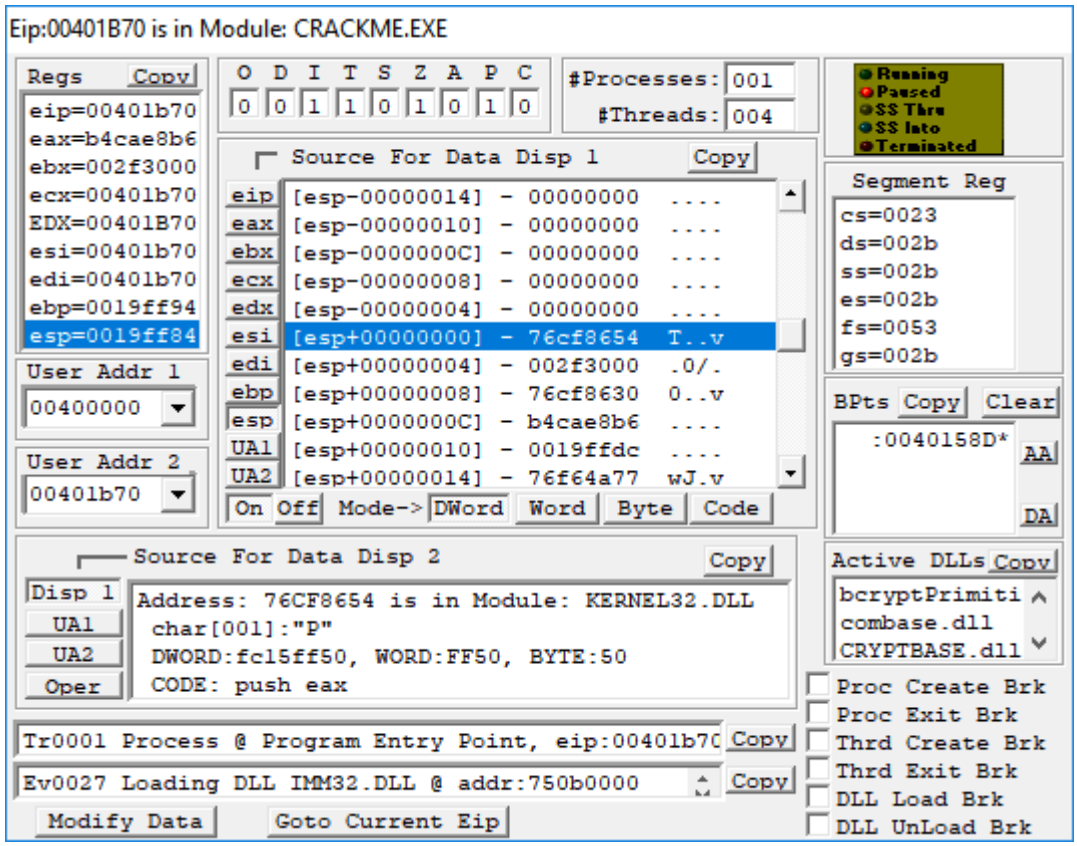


Figure III.15: Valeurs des registres

3- La dernière corresponde aux valeurs des instructions en exécution.

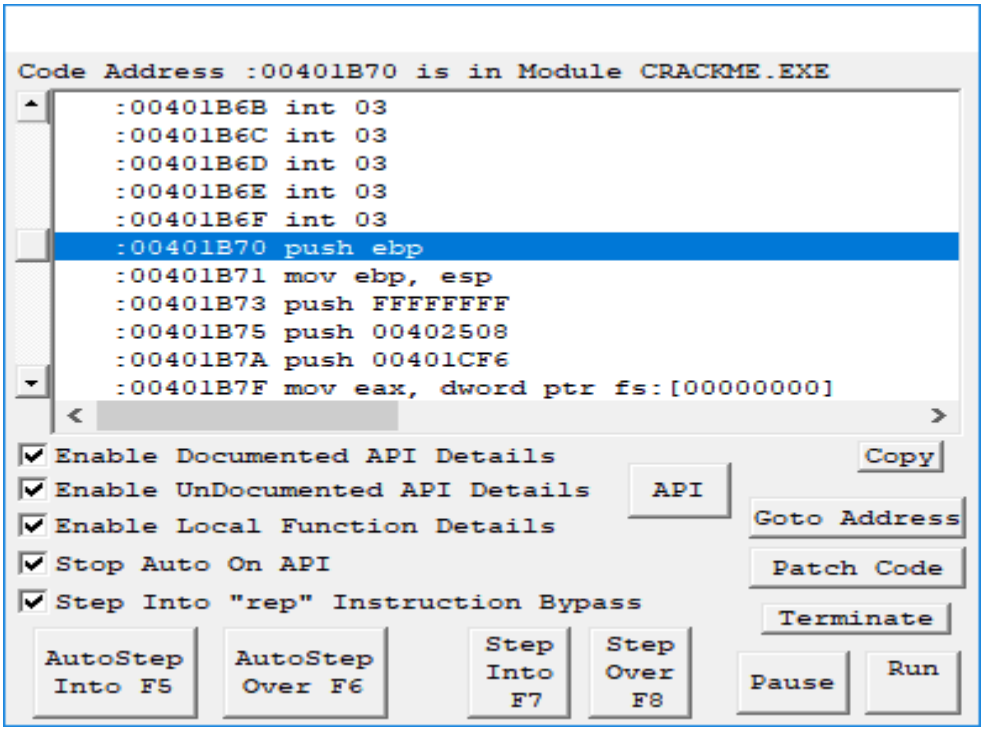


Figure III.16: Les instructions en exécution

Nous cliquons ensuite sur le bouton **Run** de cette dernière fenêtre et il nous affiche l'exécutable de notre logiciel afin que nous réintroduisons le faut mot de passe. Il nous affiche ensuite une liste qui contient le vrai mot de passe du logiciel.

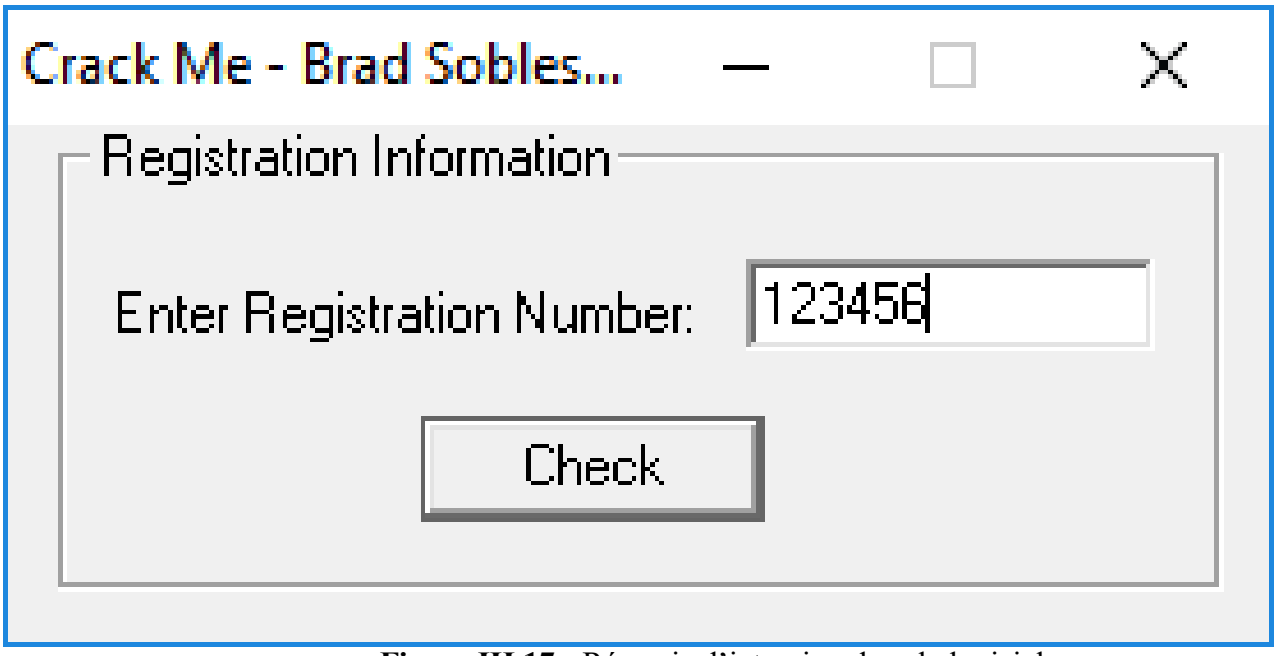


Figure III.17 : Réessaie d'intrusion dans le logiciel

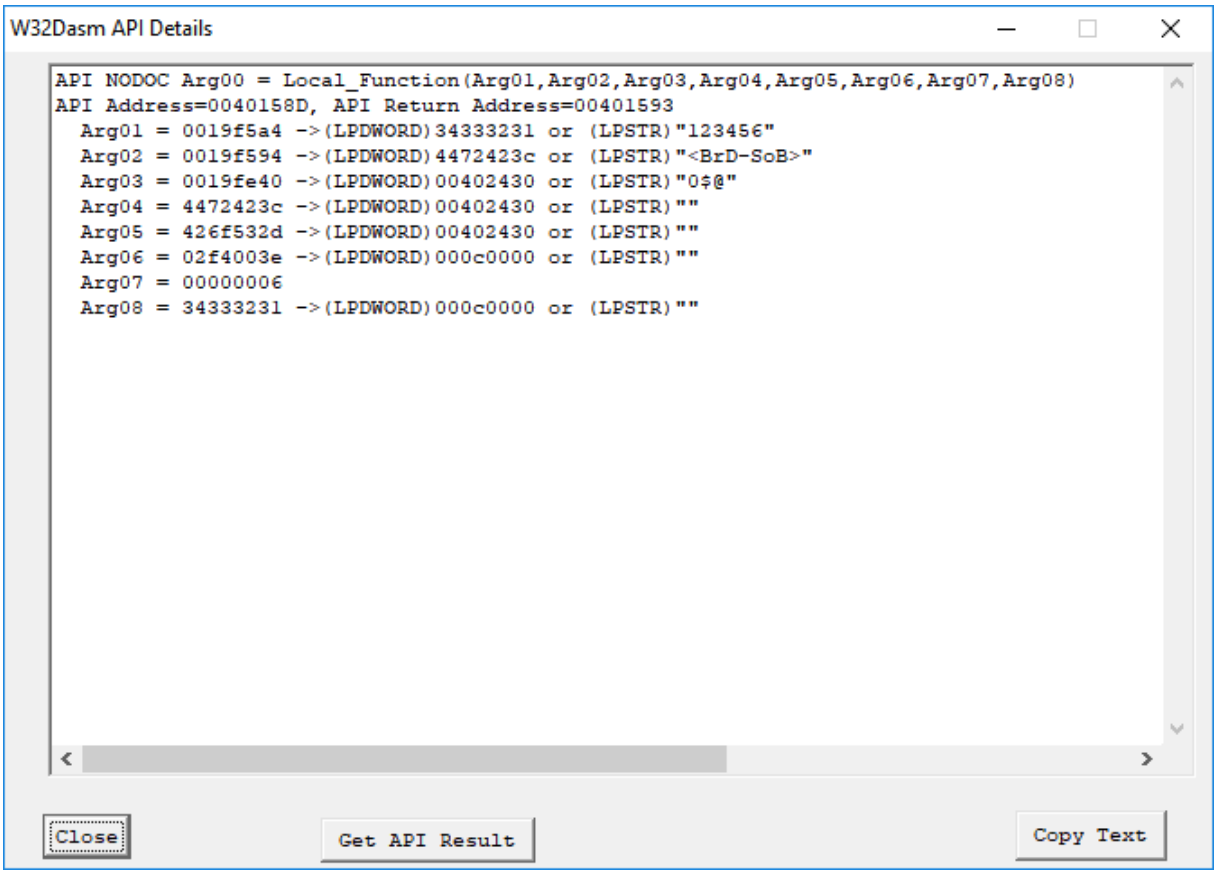


Figure III.18: Liste des API

Nous voyons clairement sur la liste à la 3ème ligne le faut mot de passe (123456) que nous avons rentré et juste en dessous le vrai mot de passe du logiciel (<BrD-SoB>). Nous pouvons maintenant entrer dans le logiciel sans aucun problème.

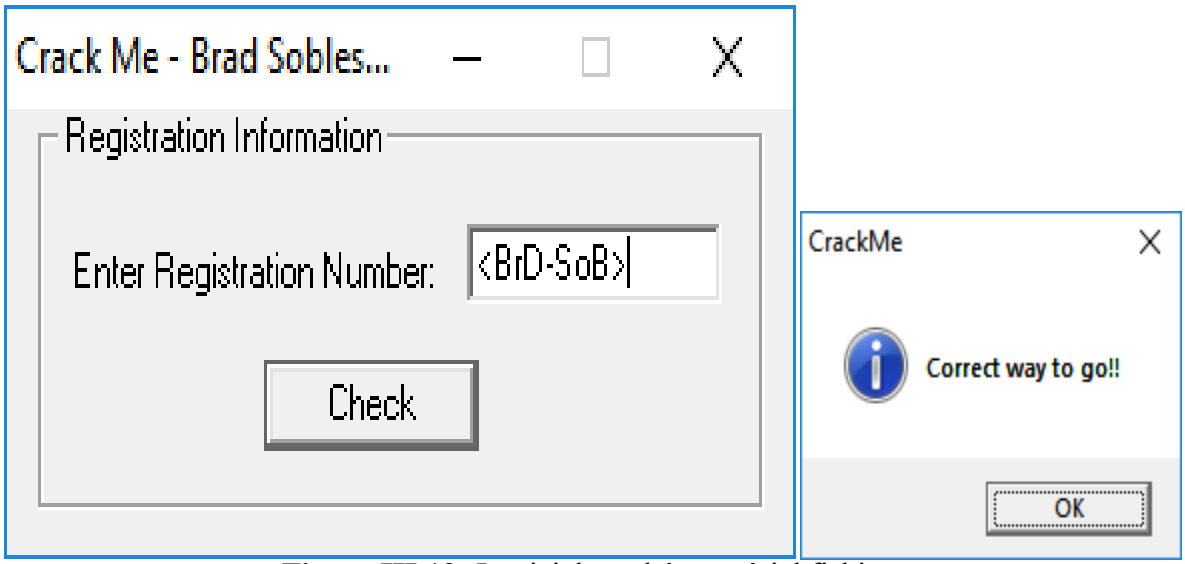


Figure III.19: Logiciel cracké par sérial fishing

Voila le logiciel est cracké.

III.4. Le keygenning:

Dans cette partie, nous avons cracké un autre logiciel (KeyMakerGen) avec la technique keygenning qui consiste à retrouver la partie du programme qui génère la clé du logiciel afin de fabriquer un programme similaire pouvant générer une clé valide à partir de n'importe quel faut clé entrée. D'où son nom keygenning qui veut dire génération de clé.

III.4.1. Keygenning avec une seule clé générée :

Tout d'abord nous essayons de nous introduire dans le logiciel avec un identifiant et un serial (mot de passe) quelconques et il nous refuse l'accès en affichant un message d'erreur du type « **Erreur d'ouverture du fichier** ».

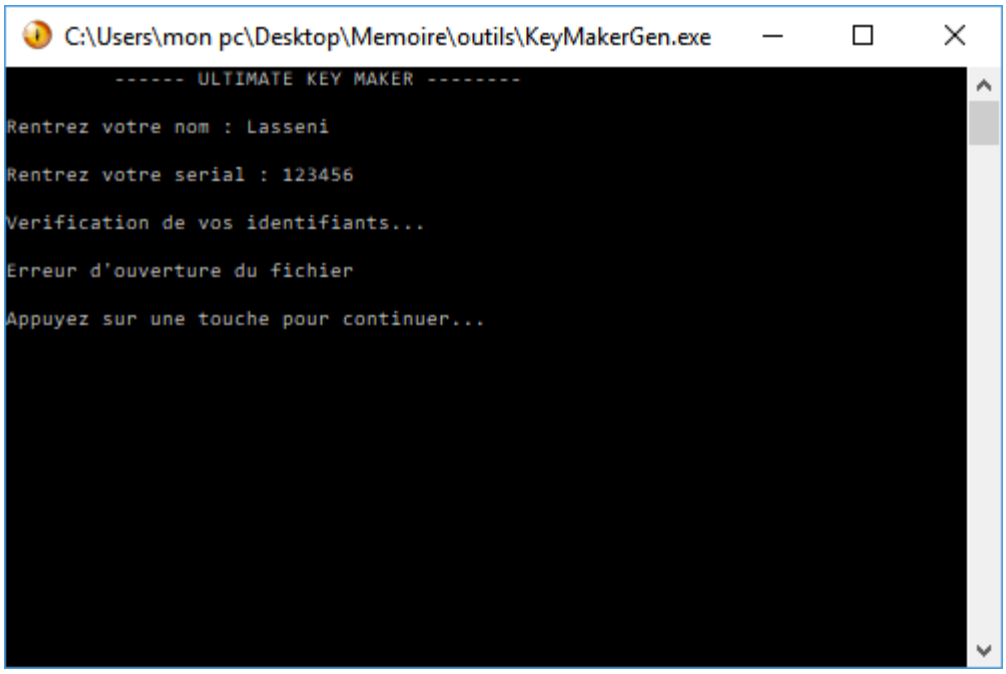
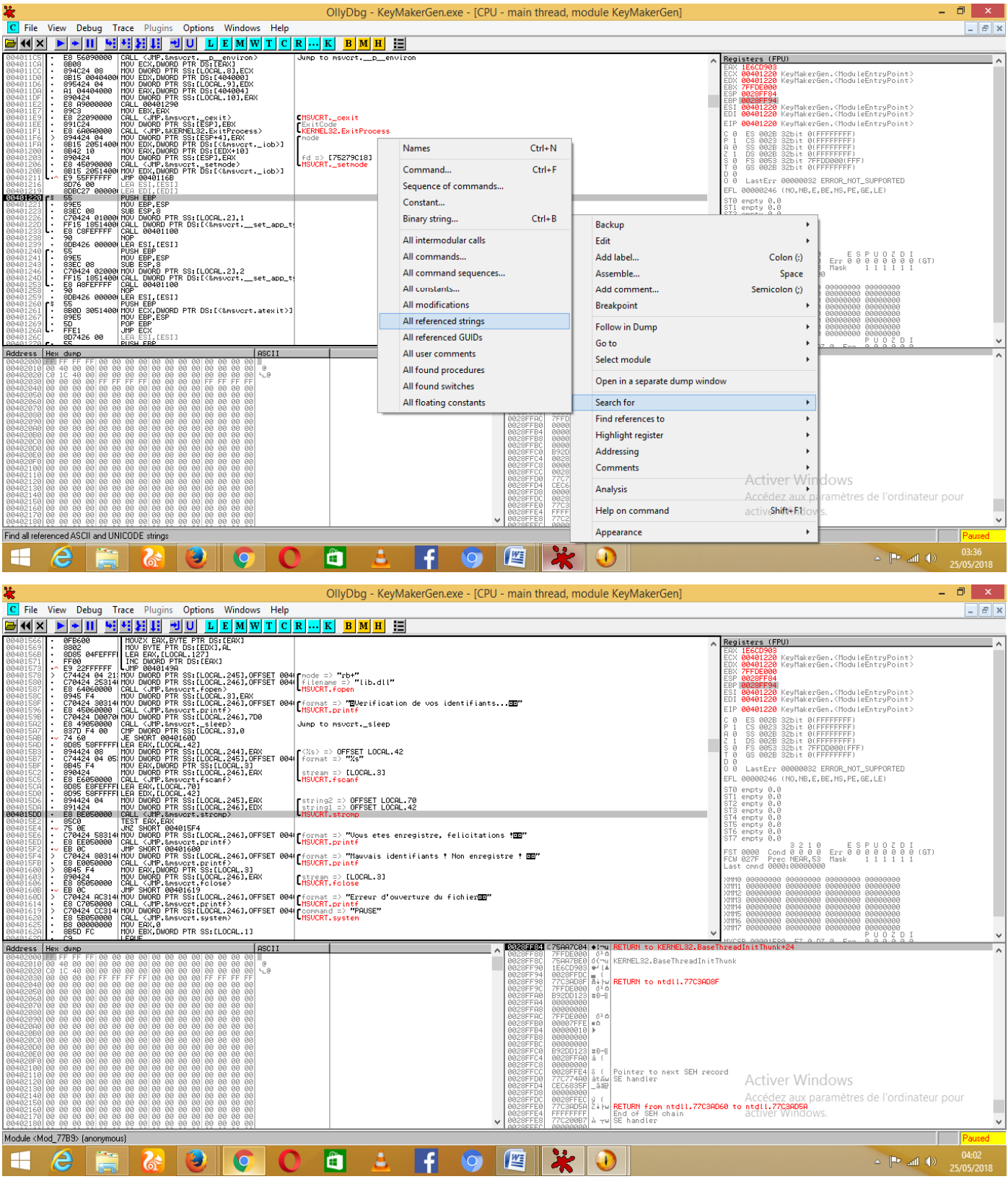


Figure III.20: Le logiciel à cracker par keygenning

Maintenant pour cracker ce logiciel, nous devons analyser le code source du programme avec un débuggeur (ici OllyDBG voire figureII.2) qui va nous permettre de retrouver l'algorithme de génération de clé. Pour cela, lançons l'application OllyDBG en y ouvrant le logiciel (KeyMakerGen) puis nous nous rendons dans la boite de dialogue des messages en faisant un clique droit >> Search for>> All referenced strings (voire figure).

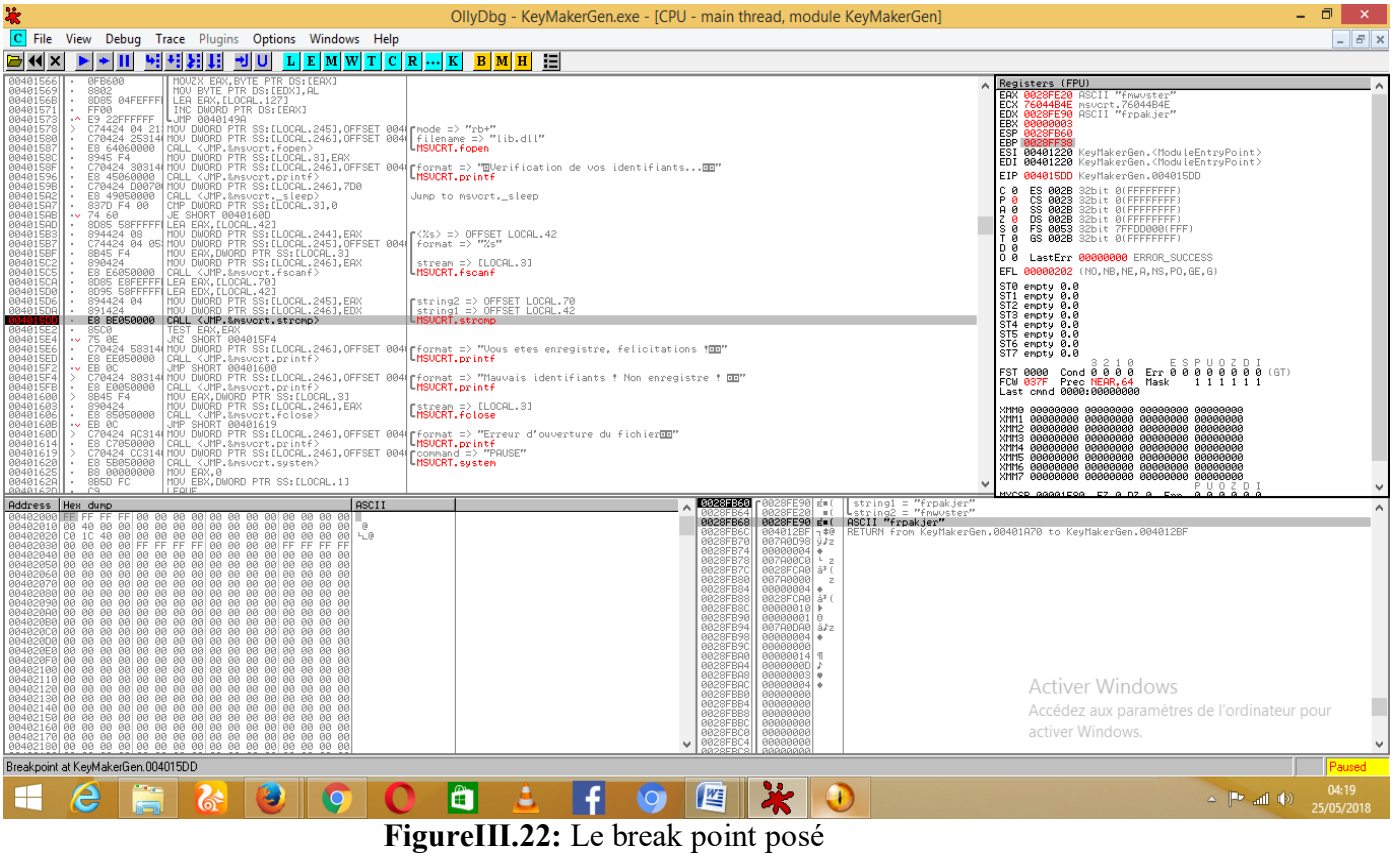


FigureIII.21: Le programme désassemblé sous OllyDBG

Comme nous voyons sur l'écran, il nous affiche les instructions en assembleur et les messages de dialogue juste devant. Nous nous intéressons d'abord aux messages de dialogue afin de trouver le bon message (pour notre cas : **Vous êtes enregistré**), c'est ce qui

s'afficherait si nous avions rentré le bon mot de passe depuis le début. Une fois trouvé, nous regardons juste un peu au dessus dans la partie assembleur et nous voyons l'instruction **CALL** (qui fait appel au bon mot de passe avant de faire la comparaison avec le mot de passe rentré) et ensuite nous posons un **break point** sur la ligne en faisant (Ctrl+F2 ou simplement F2).

Maintenant, nous lançons le programme en faisant (Ctrl+F9) et nous réintroduisons l'identifiant (Lasseni) et le mot de passe (123456), ensuite nous regardons dans OllyDBG (en haut vers la droite).

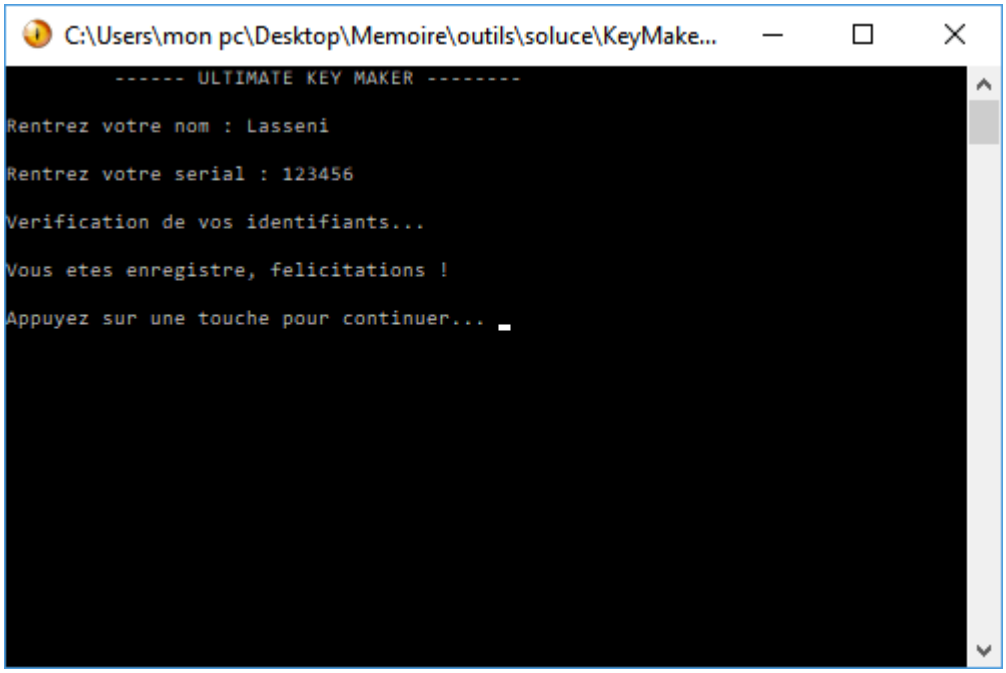


FigureIII.22: Le break point posé

Et nous voyons ceux-ci : « **qrpqkjyr** » et bien c'est le mot de passe généré par le programme qui correspond à l'identifiant (Lasseni) et le mot de passe (123456).

Maintenant, pour pouvoir nous introduire dans le logiciel, nous devons tout d'abord créer un document texte dans le même répertoire que le logiciel et y recopier le mot de passe

(qrpqkjyr) et ensuite le convertir en fichier dll (dynamic link library) avec le nom (lib.dll). Nous pouvons ensuite lancer le logiciel et y introduire l'identifiant (Lasseni) et le mot de passe (123456).



FigureIII.23: Le logiciel cracké par keygenning

Et voilà le logiciel est cracké il nous laisse entrer sans problème mais seulement avec l'identifiant (Lasseni) et le mot de passe (123456).

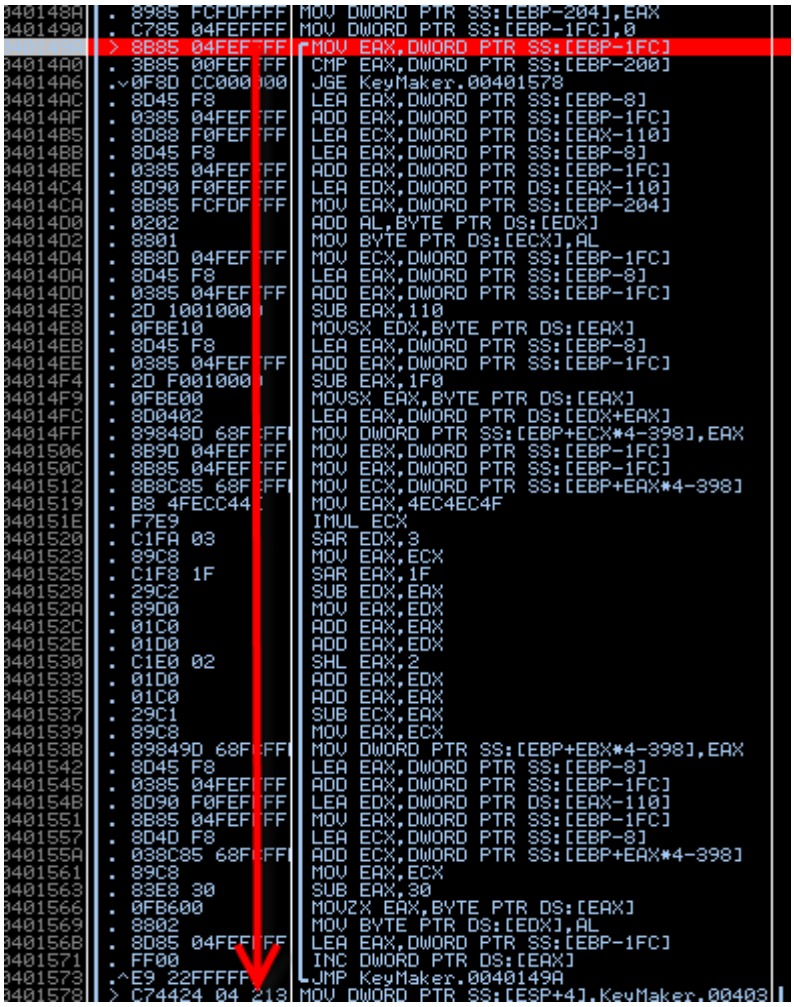
III.4.2. Keygenning avec n'importe quelle clé :

Cependant, si nous voulons rendre public notre logiciel cracké afin que n'importe qui puisse l'utiliser avec n'importe quel identifiant et mot de passe (c'est ce que fait la plupart des crackers d'ailleurs), nous devons étudier le fonctionnement de l'algorithme de génération de clé. Pour ce faire, nous étudions la relation entre le mot de passe généré et celui que nous avons rentré afin de créer un programme exécutable (keygen) pouvant faire une telle génération avec n'importe quel mot de passe.

Pour ce logiciel (KeyMakerGen) nous recommençons l'analyse du programme avec OllyDBG en suivant les mêmes étapes que ce que nous venons de faire jusqu'au niveau du mot de passe généré par le programme.

Après avoir supprimé le **break point** sur l'instruction **CALL**, nous le posons sur la ligne qui représente le début de la routine de génération de clé. Ensuite nous appuyons sur F9 pour relancer le logiciel et y mettre un identifiant et un mot de passe. Pour facilité l'explication de

la routine de génération de clé, nous avons choisi dans ce cas de mettre un identifiant et un mot de passe plus court (exemple : a et a). Ensuite nous cliquons sur F8 pour descendre ligne après ligne et à chaque fois, nous regardons la partie registre sur l'écran (en haut vers la droit) pour remarquer les changements jusqu'à la fin.

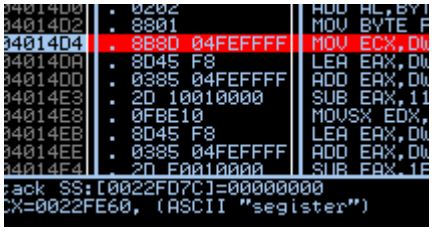


FigureIII.24: Routine de génération de clé

Après avoir analysé le programme de génération de clé, nous avons compris qu'il fonctionne de la manière suivante :

1. Il part du mot « register », quand nous introduisons un identifiant (nom) et un mot de passe (sérial).

2. Il commence la routine lettre par lettre.
3. Il ajoute la longueur du sérail à la 1ère lettre de « register » (r). Par exemple ici le sérial est (a) et sa longueur est (1), donc 1+r =s et nous aurons « segister ».



4. Il va ajouter la valeur (ASCII) du caractère du sérial et du caractère du nom, par exemple, 's' (115) en 1ère lettre du nom + 'a' (97) en 1ère lettre du sérial, ça nous donne 212.
5. Il fait un modulo 26 (modulo = division avec reste, le modulo récupère seulement le reste et pas le quotient) 212 % 26 = 4.
6. Dans un alphabet inversé, il va chercher la lettre correspondante à 4 (v), et remplace la lettre qu'il y avait avant (s) par la nouvelle lettre (v) ce qui donne « vegister ».

Il recommence cette boucle pour tous les caractères du nom, en se basant toujours sur le mot « register ».

Maintenant nous créons un programme appelé crack (keygen) qui pourra réaliser ces différents calculs afin de permettre l'utilisation du logiciel avec n'importe quel couple (identifiant/ mot de passe). Pour cela, le programme doit répondre aux conditions suivantes :

- ✓ Le programme doit demander le nom / serial
- ✓ Il doit faire les calculs rapidement
- ✓ Puis inscrire directement la réponse dans le fichier lib.dll, s'il n'existe pas, il sera créé.

Nous avons choisi le langage de programmation C/C++ pour écrire notre programme.

Début du programme

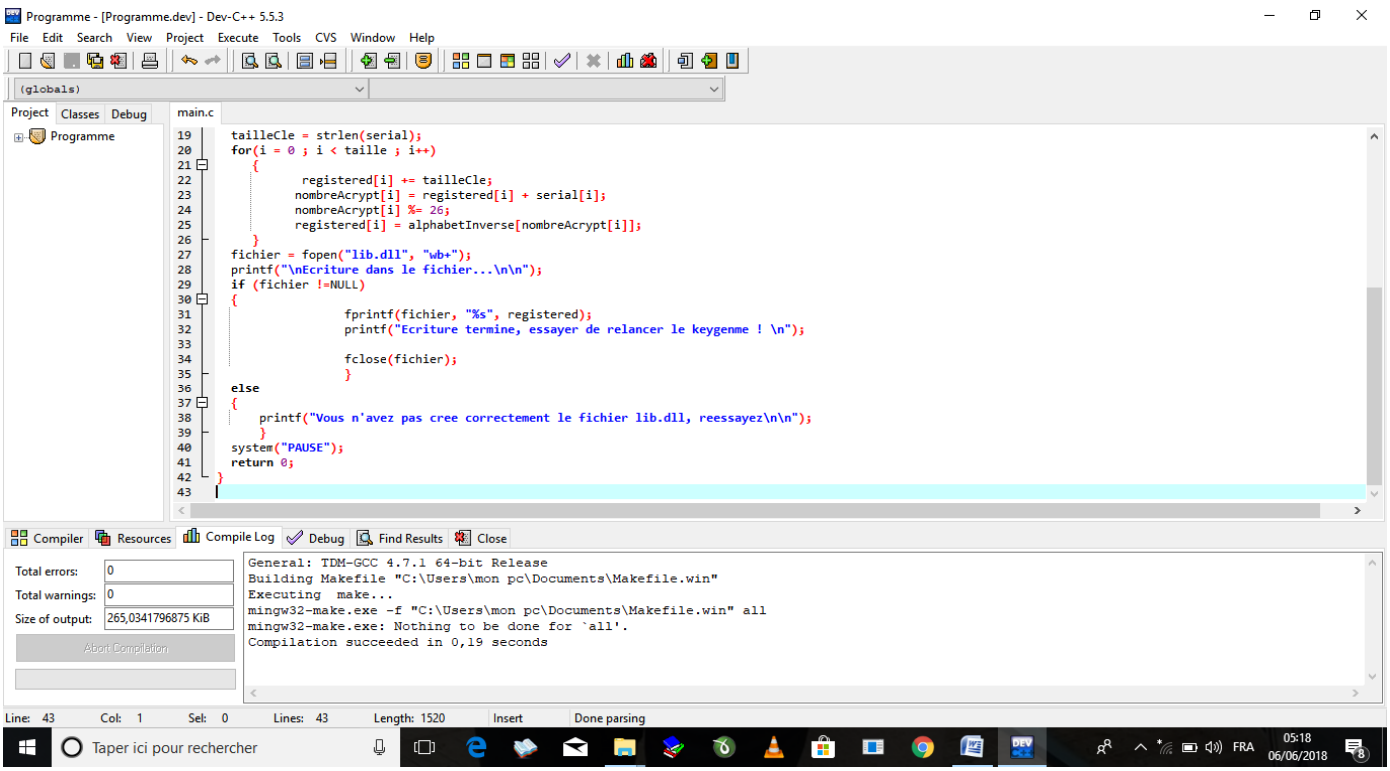
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(int argc, char *argv[])
{
```

```
 FILE* fichier = NULL;
 char alphabetInverse[27] = {'z', 'y', 'x', 'w', 'v', 'u', 't', 's', 'r', 'q', 'p', 'o', 'n', 'm', 'l', 'k', 'j', 'i', 'h',
'g', 'f', 'e', 'd', 'c', 'b', 'a', '\0'};
 char chaine[100], registered[100] = {"register"}, nom[100] =  {""}, serial[100] = {""};
 long i = 0, taille=0, tailleCle=0, nombreAcrypt[100] =  {0}, number=0;

 printf("\t ------ ULTIMATE KEY MAKER -------- \n\n");

 printf("Avant toute chose, Veillez bien a me mettre dans le même répertoire que le
keygenme\n");


 printf("\nRentrez votre nom : ");
 scanf("%s", nom);
 taille = strlen(nom);
 printf("\nRentrez votre serial : ");
 scanf("%s", serial);
 tailleCle = strlen(serial);
 for(i = 0 ; i < taille ; i++)
   {
        registered[i] += tailleCle;
       nombreAcrypt[i] = registered[i] + serial[i];
       nombreAcrypt[i] %= 26;
       registered[i] = alphabetInverse[nombreAcrypt[i]];
   }
 fichier = fopen("lib.dll", "wb+");
 printf("\nEcriture dans le fichier...\n\n");
 if (fichier !=NULL)
 {
            fprintf(fichier, "%s", registered);
            printf("Ecriture termine, essayer de relancer le keygenme ! \n");

            fclose(fichier);
```

```
            }
 else
 {
    printf("Vous n'avez pas cree correctement le fichier lib.dll, reessayez\n\n");
    }
 system("PAUSE");
 return 0;
}
```

Fin du programme

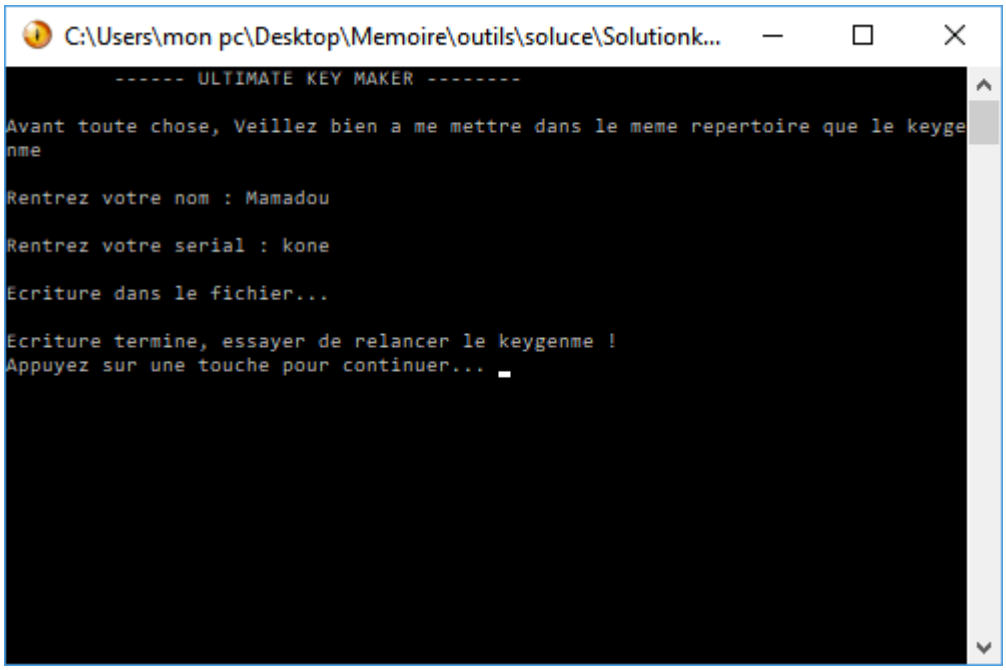
Après avoir écrire notre programme dans un éditeur de texte, nous l'avons compilé avec un logiciel de compilation du langage C (ici Dev-C++).



FigureIII.25 : Compilation du programme sous Dev-C++

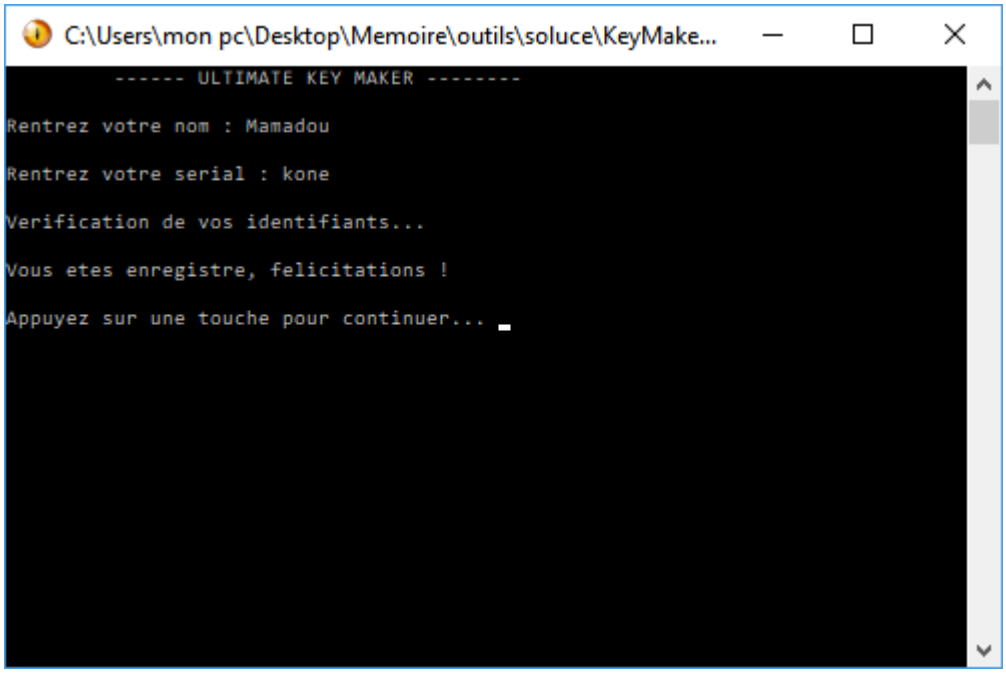
Après cela, nous enregistrons le programme en exécutable sous le nom (SolutionKeyMaker.exe) ensuite nous le plaçons dans le même répertoire que le logiciel puis nous créons le fichier vide (lib.dll).

Nous double cliquons maintenant sur le programme compilé (SolutionKeyMaker.exe) et il nous demande d'entrée un nom et un sérial.



FigureIII.26: Le programme compilé

Nous voyons tout à coup sur l'écran le message d'écriture dans le fichier, il s'agit du fichier (lib.dll) que nous avons créé dans le répertoire du logiciel. Le programme va écrire dans ce fichier, le mot de passe généré à partir du couple (nom/sérial) que nous avons rentré. A partir de là, nous pouvons utiliser le logiciel avec ce couple (nom/sérial).

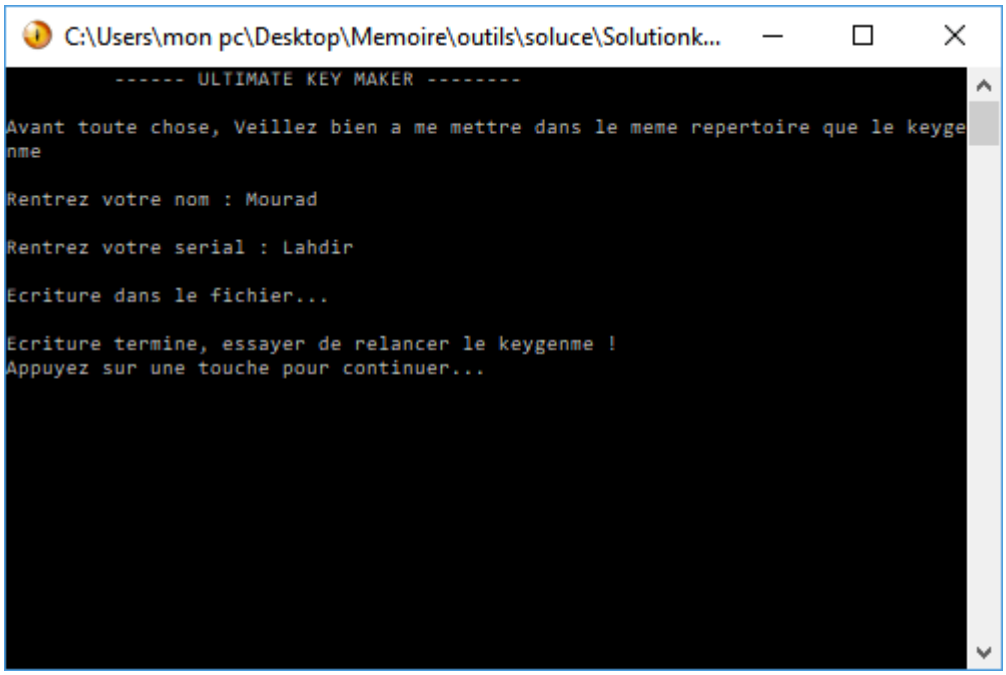


FigureIII.27: Le logiciel cracké par keygenning

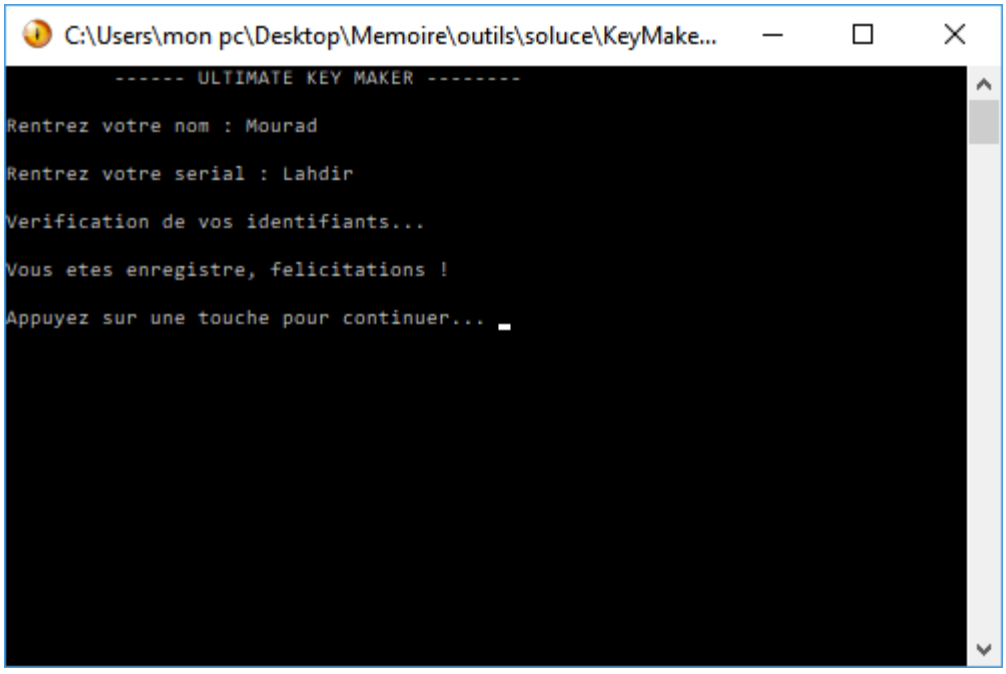
Et voilà le logiciel nous laisse entrer **(Vous êtes enregistré, félicitations !)**.

Nous pouvons aussi faire d'autres exemples pour voir si le programme est utilisable par tous.

Exemple1: Essaie avec l'identifiant (Mourad) et le mot de passe (Lahdir)



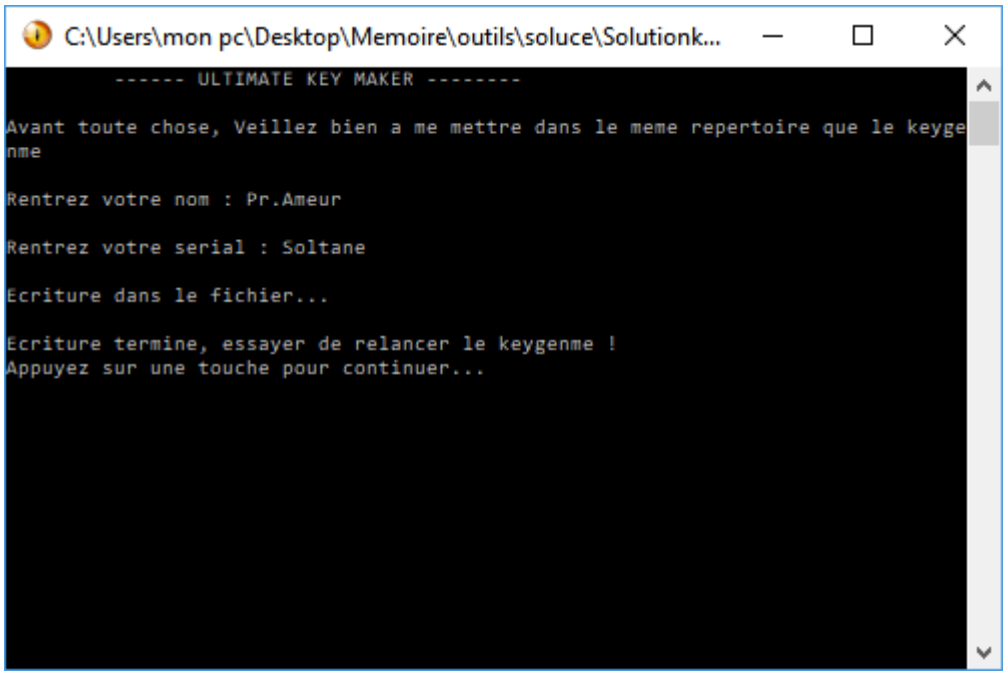
FigureIII.28: Le programme compilé



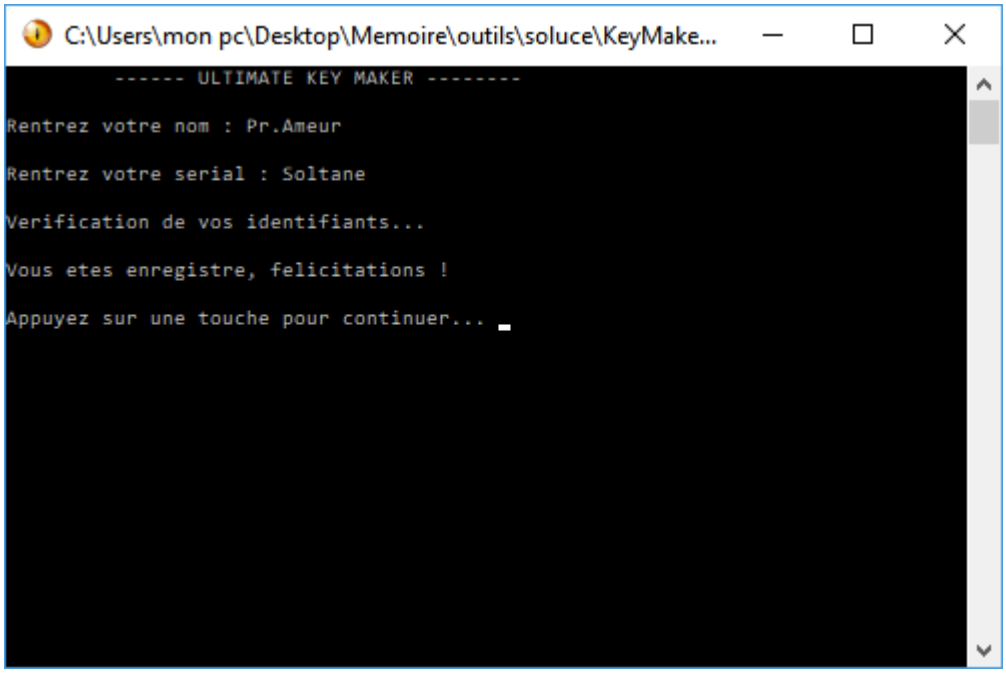
FigureIII.29: Le logiciel cracké par keygenning

Le logiciel laisse entrer le couple (Mourad/Lahdir).

Exemple2: Essaie avec l'identifiant (Pr.Ameur) et le mot de passe (Soltane)



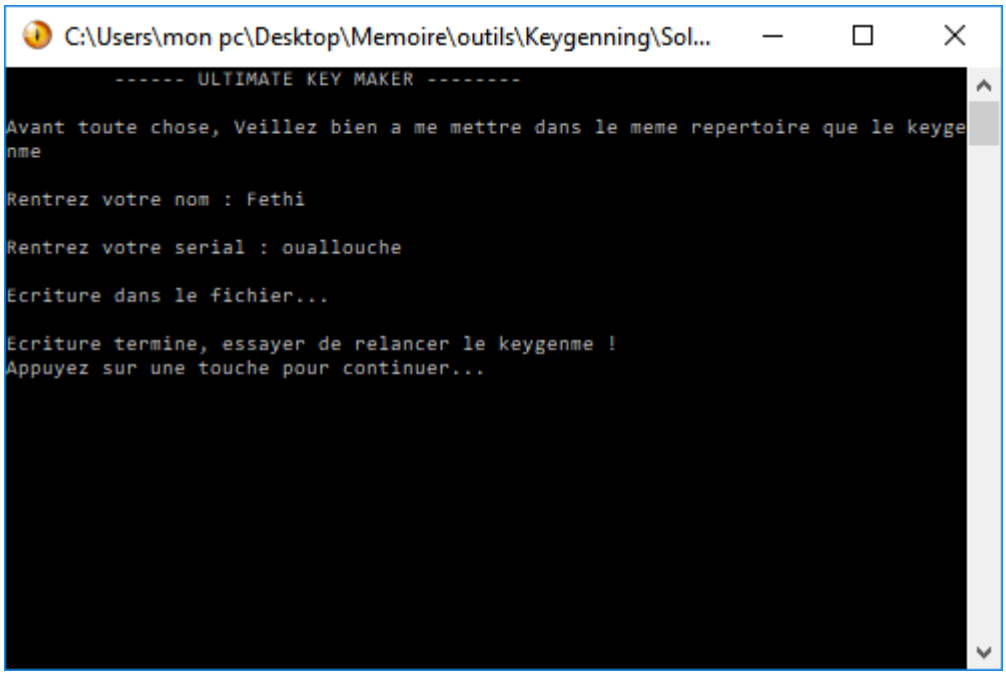
FigureIII.30: Le programme compilé



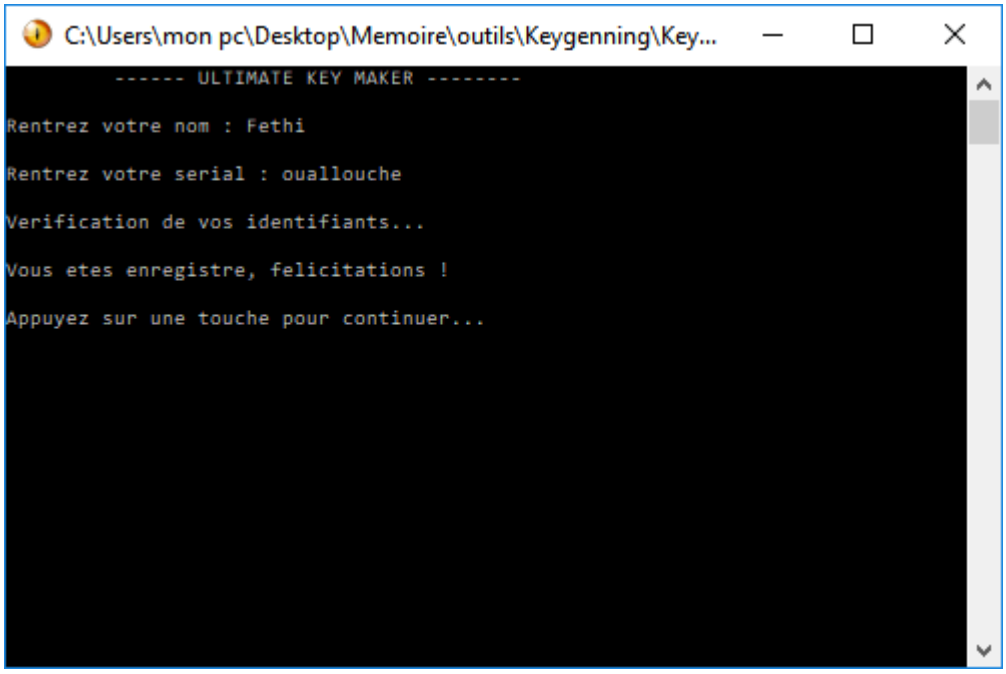
FigureIII.31: Le logiciel cracké par keygenning

Le logiciel laisse entrer aussi le couple (Pr.Ameur/Soltane).

Exemple3 : Essaie avec l'identifiant (Fethi) et le mot de passe (ouallouche)



FigureIII.32: Le programme compilé



FigureIII.33: Le logiciel cracké par keygenning

Et enfin le logiciel laisse entrer le couple (Fethi/ouallouche).

Maintenant nous pouvons remarquer que le programme a bien marché et peut permettre à toutes personnes d'utiliser ce logiciel. Cependant, ce programme ne peut être utile que pour ce logiciel uniquement. Par conséquent, il est nécessaire d'écrire un programme spécifique pour

chaque logiciel puisque tous les logiciels n'ont pas le même algorithme de génération de clé, cela en raison de la diversité des logiciels et leurs éditeurs.

III.5. Discussions :

Les différentes techniques de cracking que nous avons mises en application sont très utilisées pour cracker des logiciels, surtout la dernière. Cependant, ces techniques peuvent ne pas fonctionner sur d'autres logiciels en raison de la manière de l'écriture du programme ou la politique de sécurité appliquée sur ces derniers.

La technique patching à l'avantage d'être simple à réaliser, mais elle est inefficace pour la plupart des logiciels car il suffit que l'éditeur du logiciel ait connaissance de cette technique pour qu'il multiplie l'instruction du message d'erreur dans son programme. Ce qui fait qu'elle est moins utilisée par rapport aux autres techniques.

La technique serial fishing est efficace car elle permet de trouver le mot de passe dans le programme sans rien modifier. Cependant, l'analyse du programme ne permet pas de voire le mot de passe en clair de certains logiciels si celui-ci est crypté.

La technique keygenning quant à elle, est la plus utilisée par les crackers, car elle a l'avantage de permettre l'utilisation du logiciel cracké par un grand nombre de personnes. Cependant, elle requiert une très bonne maitrise des langages de programmation pour la compréhension et la reproduction du programme de génération de clé. Cette technique comme les autres est totalement impossible lorsque le code source du logiciel est crypté, à moins que le cracker réussisse d'abord à casser l'algorithme de chiffrement utilisé.

En somme, ce qu'il faut retenir est que, l'utilisation de ces techniques dépend du logiciel qu'il faut cracker. Il faut donc à chaque fois analyser le code source du logiciel afin de déterminer la technique adéquate à appliquée. C'est ce qui fait que cela demande beaucoup de temps et beaucoup d'efforts.

Conclusion :

Conclusion

Durant ce travail, nous avons mis en lumière la prépondérance de la sécurité des systèmes informatiques en occurrence les logiciels. Ainsi, nous avons traité les différentes notions de piratage informatique, les motivations des pirates informatiques et surtout les outils et mécanismes par lesquels ces derniers parviennent à leur objectif.

L'objectif que nous nous étions fixé à été atteint, car nous sommes parvenus à comprendre et à mettre en application les différentes techniques de cracking avec succès. Cela à permis de comprendre également l'importance du cracking dans la sécurité du logiciel. En effet, le cracking a l'avantage d'être utiliser pour tester le niveau de sécurité des logiciels après leur conception. Il permet également d'analyser le code source des logiciels pour extraire les programmes malveillants destinés à voler des informations, afin d'assurer la sécurité des systèmes informatiques.

Cependant, l'application du cracking n'est pas la même pour tous les logiciels, il nécessite une étude du code source de chaque logiciel avant de pouvoir le cracker. Cela représente une complexité dans l'apprentissage du cracking, car ça demande du temps et d'efforts malgré que la méthodologie reste la même.

Par ailleurs, le cracking ne fonctionne pas sur un logiciel dont le code source est crypté, puis que cela rend le programme incompréhensible à moins que le cracker procède d'abord par casser l'algorithme de chiffrement. En effet, la cryptographie apparait comme une solution de sécurité pour les éditeurs de logiciel, mais un grand obstacle pour les pirates informatiques. De ce fait, les crackers de leur coté cherchent par tous les moyens, des solutions pour développer des outils capables de casser n'importe quel algorithme de chiffrement. A ce rythme, de nouvelles techniques de cracking peuvent voire le jour à tout moment, puis que le monde informatique est un monde en perpétuelle évolution.

Nous espérions pouvoir dire que la compréhension des trois techniques du cracking suffise pour s'assurer de la sécurité des logiciels en occurrence les systèmes informatiques, mais cela n'est pas dans l'ordre du possible. En effet, la sécurité d'un système informatique n'est que temporaire avant que les pirates ne réussissent à la déjouer. Il faut donc toujours suivre le cours de l'évolution de l'informatique et s'informer constamment sur les nouvelles techniques de piratages (ou cracking) pour sécuriser les systèmes (logiciels) en conséquence.

Bibliographie

❖ **Livres :**

[1] Vladimir AMAN, (2014), Concevoir la Sécurité Informatique en Entreprise, édition Creative Commons, P.153.

[2] Jean-François PILLOU, Jean-Philippe BAY, (2009), Tout sur la sécurité informatique, 2e édition, DUNOD, Paris. P.232.

[3] Nzeka GILBERT alias KHAALEL, (2004), HACKING/SECURITE HAND-BOOK, P.275. Disponible sur http://www.cksecurity.fr.fm ET http://cksecurity.free.fr

[4] Alexandre GOMEZ-URBANIA, (2010), HACKING INTERDIT*, 4ème édition, Micro Application, Paris, P.480.

[5] Cédric LLORENS, Laurent LEVIER, Denis Valois, (2006), **Tableaux de bord de la sécurité réseau**, 2ème édition, EYROLLES, Paris, P.560.

[6] Touradj EBRAHIMI, Franck LEPREVOST, Bertrand WARUSFEL, (2006), Cryptographie et sécurité des systèmes et réseaux, 4ème édition, LAVOISIER, Paris. P.307.

❖ **Sites :**

[A] https://fr.m.wikipedia.org>php>cracking (Articles sur le cracking).

[B] https://www.leblogduhacker.fr/ (Les articles et les cours sur le hacking).

[C] http://hack.over-blog.com/pages/Devenir_Un_Hacker (Information sur le hacking).

[D] https://www.securiteinfo.com/attaques/hacking/ (Site de sécurité informatique).

[E] https://www.developpez.net/forums/ (Le club des professionnels en informatique).

[F] http://beta.hackndo.com/le-hacking-cest-quoi/ (Article en ligne sur hacking).

[G] http://www.ii.uib.no/Larsr/bc.html (liste des attaques connues).

[H] http://www.securiteinfo.com (L'actualité de la sécurité informatique).

[I] http://duveym.e-monsite.com (Hacking ET Cracking).

[J] http://bork0.dl2.free.fr/php/cracking/keygenning (Cours de keygenning)

[K] http://www.daemoncrack.fr.st (Cours et pratique de cracking).

[L] http://xtx.free.fr (Site de défi de cracking logiciel).

[M] http://bork0.dl2.free.fr/php/cracking/ (Site dédié au cracking logiciel).

[N] http://deamonftp.free.fr/deamoncrack/index0.htm (Cours et pratique de cracking).

[O] http://pifoman.free.fr/cours_cracking/cours_assembleur.htm (cours de l'assembleur).

[P] https://anti-cybercriminalite.fr (Site de lute contre la cybercriminalité, actualités).

[Q] http://secure.nesquiikz.com (Site de hacker informatique).