

# Rapport de projet ZZ1

Création d'un programme de  
récupération de données personnelles  
(malware<sup>(1)</sup>) en Python



# Remerciements

Nous tenons à remercier monsieur Jacques LAFFONT, pour nous avoir accompagné pendant ce semestre lors de notre projet et pour nous avoir accordé sa confiance pour la réalisation de ce projet.

# Sommaire

Introduction	4
Objectif du projet	5
Résultat obtenu	6
Aspect pédagogique	7
Aspect sécurité / Aspect simplicité de mise en œuvre	8
Aspect python	9
Aspect polymorphe	10
Limites du projet	11
Détail et structuration de l'implémentation	12
Capture des données	13
Bilan technique	16
Conclusion	17
Lexique	18
Annexes	19

# Introduction

Durant le second semestre de notre première année d'école d'ingénieurs à l'ISIMA, nous avons pour objectif de faire un projet d'une vingtaine d'heures chacun sur un sujet au choix.

Venant tous les deux de DUT, mais n'ayant pas vraiment d'expérience dans la création de malware, nous avons tout d'abord pensé à utiliser nos bases de programmation pour créer un BotNet permettant l'envoi de requêtes sur plusieurs machines distantes afin de récupérer des données à l'insu de l'utilisateur.

Nous avons donc demandé conseil à Monsieur Laffont et lui avons proposé de nous accompagner dans notre projet en devenant notre tuteur de projet.

Cependant, nous avons rapidement compris qu'il allait d'abord falloir programmer un malware avant de pouvoir le transmettre sur plusieurs machines et de récupérer leurs données.

Le but de ce projet a donc été d'essayer de voir ce qu'il était possible de faire en tant que "débutants", concernant la récupération de données sur un ordinateur distant, et ce, avec un investissement faible en terme d'heures, le temps consacré au projet tuteuré de première année étant seulement d'une vingtaine d'heures par personnes.

## Objectif du projet

L'objectif de notre projet était de mettre en place un programme permettant de récupérer des données sur un ordinateur distant. Il fallait mettre en place une connexion entre 2 ordinateurs (un client et un serveur), afin que le client exécute des instructions puis renvoie les données récupérées au serveur.

De plus, il fallait faire en sorte que le programme soit le plus discret possible afin que le client ne détecte pas d'activité suspecte en cours d'exécution sur l'ordinateur.

Par ailleurs, le programme avait au départ pour but d'être cross-platform<sup>(2)</sup> afin de fonctionner sur n'importe quel système d'exploitation, ou en tout cas sur les principaux étant Linux, Mac et Windows.

## Résultat obtenu

Concernant nos résultats après une cinquantaine d'heures de développement sur le projet, le programme fonctionne sur Linux et MacOS, cependant, quelques fonctionnalités ne fonctionnent pas encore complètement sur Windows.

De plus, la mise en place du programme n'a pas mis très longtemps étant donné qu'en une cinquantaine d'heures nous avons réussi à récupérer de nombreuses données, et que la récupération et l'envoi de ces données n'est pas détectable par les antivirus (hormis par Kaspersky dans le cas de l'exécutable Windows).

Par ailleurs, le programme permet d'envoyer depuis le serveur jusqu'au client une suite d'instructions à exécuter afin de récupérer que ce soit une impression d'écran, un enregistrement audio, un enregistrement vidéo, ou bien la saisie d'un mot de passe administrateur.

Cependant, le programme possède tout de même certaines limites.

Premièrement, il est obligatoire sur MacOS d'accepter l'utilisation d'outils tels que la webcam ou bien le microphone lors de leur première utilisation par le terminal (ou n'importe quel autre programme) ce qui indique à l'utilisateur que ces outils fonctionnent sans qu'il ne l'ait demandé.

De plus, une LED s'affiche sur nos ordinateurs lors de l'utilisation de la webcam. L'utilisateur est donc au courant qu'il est filmé ou en tout cas que sa webcam est en cours d'utilisation par un programme alors qu'aucun ne devrait l'utiliser sur le moment.

Enfin, sur Linux et MacOS, l'écoute des saisies au clavier nécessite que la focalisation soit sur la fenêtre écoutant le clavier, empêchant la création d'un keylogger<sup>(3)</sup>. Nous avons donc contourné le problème en créant une fausse fenêtre de saisie de mot de passe administrateur récupérant ce dernier et l'envoyant ensuite au serveur.

## Aspect pédagogique

Un des aspects les plus importants de ce projet fut l'aspect pédagogique. En effet, nous avons pu découvrir de nombreuses choses durant le développement du projet que nous n'imaginions pas forcément aussi simples dans leur mise en place.

Par exemple, nous avons pu voir à quel point il était rapide de récupérer des données sur un ordinateur à partir du moment où un accès à celui-ci était disponible. De plus, les données que nous pensions être les plus importantes et les plus sensibles n'ont pas forcément été celles avec lesquelles nous avons eu le plus de mal. La webcam par exemple nous paraissait être un élément très sensible du fait des images qu'elle pouvait récupérer. Cependant, une LED s'allume lors de son utilisation rendant la récupération des données beaucoup moins discrète. Malgré cela, de nombreuses personnes la cachent pour ne pas être espionnées alors qu'elles ne font rien pour bloquer leur micro ou l'impression de leur écran qui eux peuvent être tout aussi sensibles et ne nous préviennent pas lors de leur utilisation.

Par ailleurs, nous avons pu comprendre qu'il ne suffisait pas d'avoir accès à l'ordinateur et donc aux données, mais qu'il fallait aussi mettre en place un envoi sécurisé et indétectable de celles-ci. Actuellement, les données sont envoyées au centre de commande en clair et d'un seul coup sur le réseau dès qu'elles sont récupérées par le client. Cependant, cette méthode est détectable si quelqu'un sniffe le réseau. Il faudrait donc envoyer les données par petits paquets à différents moments tout en chiffrant les données pour rendre la transmission moins détectable.

Enfin, ce projet nous a permis de comprendre que sans réelle formation ou base en programmation système, il était possible de créer très rapidement un programme très peu détectable qui suffit à faire des dégâts. Avec peu d'investissement étant donné que nous n'y avons passé qu'une cinquantaine d'heures au total, et un langage peu adapté comme le python étant d'un très haut-niveau contrairement au C, voire à l'assembleur, nous avons réussi à espionner un ordinateur relativement facilement et de manière peu, voire pas, détectable, le tout en étant uniquement des étudiants.

Le résultat du même projet par une équipe de développeurs/pirates vétérans, dans un langage adapté, doit donc être bien plus efficace, ce qui peut amener à réfléchir.

## **Aspect sécurité / Aspect simplicité de mise en œuvre**

D'un point de vue sécurité et simplicité de mise en œuvre, nous n'avons pas eu de mal à accéder aux données et à les récupérer étant donné que tout est expliqué sur internet. De plus, nous avons utilisé des modules python qui sont utilisés par des applications légitimes. Il est donc difficile de détecter/empêcher l'utilisation de ces modules pour sécuriser le système.

Par exemple, OpenCV-python est un module très utilisé pour la capture et le traitement d'images que ce soit pour de la détection de mouvement ou même pour de la détection de motifs. Il est donc utilisé pour des projets utilisant des caméras autres que celle du PC lui-même.



## Aspect python

Concernant le développement de l'application, nous avons utilisé le langage python qui est un langage de programmation interprété, multi-plateformes, offrant des outils haut niveau et une syntaxe simple à utiliser. Ce langage nous a donc permis d'utiliser le programme sur Linux, MacOS, et même Windows pour certaines fonctionnalités.

De plus, n'étant pas long à apprendre, nous avons pu nous mettre rapidement dans le projet et n'avons eu à utiliser que des bibliothèques existantes et nous permettant d'effectuer tout ce dont nous avons besoin, seulement en spécifiant les bons paramètres aux fonctions que nous utilisons.

Par ailleurs, il nous a été possible de créer un exécutable Windows pour notre programme permettant l'installation de tous les éléments nécessaires à son bon fonctionnement automatiquement. Dorénavant, il nous est possible d'installer et de lancer le programme automatiquement sur un ordinateur simplement en lançant l'exécutable. Mais il pourrait être encore plus rapide de brancher une clé USB dans un ordinateur qui ferait tout pour nous.

Cependant, il y a quelques limites à l'utilisation de ce langage. En effet, Python étant un langage haut niveau, il est difficile, voire impossible, de faire un programme qui ne soit pas détectable tout en permettant de récupérer les données que l'on envoie à intervalles de temps réguliers, voire même seulement lorsqu'une application est en cours d'utilisation contrairement à du C.

## Aspect polymorphe

Une autre fonctionnalité intéressante du programme est son aspect polymorphe. Il est possible de modifier les actions effectuées sur l'ordinateur de la victime sans y avoir accès ultérieurement. En effet, le client demande au serveur de lui envoyer les instructions à réaliser lors de son exécution. Il interprète ensuite ce qu'il reçoit dynamiquement (au runtime) et renvoie les données récupérées. Il est donc possible de lui demander d'exécuter du code python différent à chaque exécution.

Le programme peut donc paraître inoffensif sur l'ordinateur de la victime pendant un certain temps, puis récupérer le maximum de données possibles avant d'être détecté et arrêté.

## Limites du projet

Cependant, le projet a tout de même des limites.

Premièrement, malgré le fait que les fonctions utilisées par le programme ne soient pas détectées comme malveillantes par les antivirus (d'après un scan virus total disponible en annexe), l'antivirus Kaspersky détecte l'exécutable Windows permettant l'installation et le lancement du programme comme un programme dangereux et sûrement malveillant. De plus, certaines des fonctionnalités de notre programme ne fonctionnent pas encore sur Windows. Il n'est donc pas encore totalement cross-platform pour le moment.

Par ailleurs, toutes les données sont envoyées en clair et en même temps du client au serveur lorsqu'elles sont récupérées. Il est donc possible pour quelqu'un s'y connaissant un minimum de détecter l'envoi de données non désiré vers un serveur et ainsi d'être au courant de l'exécution d'un programme malveillant sur l'ordinateur.

Concernant la récupération de données depuis la webcam, celle-ci allume une LED lors de son utilisation que ce soit pour prendre une vidéo de 5 secondes ou seulement pour prendre une photo en quelques millisecondes. Il est donc préférable de ne pas prendre de vidéo pour être le plus discret possible.

Enfin, le python n'est pas le langage le plus adapté pour être exécuté sans être détecté, et malgré le fait qu'il soit très rapide à mettre en place et donc nécessaire dans le cas d'un projet ou d'un programme basique devant fonctionner rapidement, il n'est pas recommandé de développer un programme malveillant sur python mais plutôt à l'aide d'un langage bas niveau qui sera beaucoup plus long à mettre en place mais surtout beaucoup plus discret.

## Détail et structuration de l'implémentation

Concernant la structuration de notre projet, il est séparé en 2 parties. L'une concerne la partie client avec la réception, l'exécution des instructions, la récupération et l'envoi des données récupérées. L'autre concerne la partie serveur avec l'envoi des instructions et la réception des données récupérées sur l'ordinateur victime.

Le but de cette structuration était au départ d'avoir des classes les plus petites possible pour éviter la détection et permettant un traitement simplifié des informations.

Cependant, au fur et à mesure des ajouts de fonctionnalités, nos classes ont grossi et se sont retrouvées avec plusieurs utilités. Il aurait donc sûrement fallu diviser ces classes en plusieurs sous-classes permettant pour chacune soit le traitement des instructions à exécuter, soit le chiffrement des données ainsi que leur encodage avant l'envoi, soit l'envoi des données au serveur. Cela aurait par exemple permis à un éventuel groupe de reprendre notre projet pour y ajouter l'envoi sécurisé et indétectable des données sans avoir à s'imprégner dans un premier temps du fonctionnement de notre programme fichier par fichier. Cependant, il nous aurait fallu un peu plus de temps que nous n'avons pas eu.

# Capture des données

## Capture d'écran<sup>(4)</sup>

La capture d'écran utilise le module **pyscreenshot**. Ce module fonctionne uniquement sur les systèmes UNIX, il fonctionne donc uniquement sur les différents Linux et sur MacOS et non sur Windows.

Lorsque le module effectue une impression d'écran, l'image qui est reçue est une matrice permettant de l'envoyer sur le réseau simplement, elle est transformée en image PNG et envoyée sous forme d'octets. Comme l'image est envoyée sous forme d'octets, la sauvegarde côté serveur est triviale, il suffit d'écrire les octets reçus dans un fichier PNG ouvert en écriture binaire.

L'image restant stockée en mémoire, il est très difficile de détecter qu'une impression d'écran a été prise. Elle n'est ni sauvegardée dans un fichier à la racine du programme, ni sauvegardée dans un fichier temporaire, ce qui pourrait indiquer que quelque chose s'est passé. De plus, il n'y a aucune indication permettant de savoir que l'écran a été photographié.

## Capture de l'audio par le microphone

Le programme utilise le module **python-sounddevice** pour enregistrer le son à l'aide du microphone. Le fonctionnement général de l'enregistrement du microphone est le même que pour la capture d'écran : l'image est stockée sous forme de matrice, convertie en fichier binaire en mémoire (ici un fichier WAV) puis envoyée sur le réseau. La réception suit également le même principe : les octets reçus sont simplement écrits dans un fichier ouvert en écriture binaire.

Pour la détection, à l'image de la capture d'écran, il est difficile de détecter que le son est enregistré : le fichier est stocké en mémoire vive et jamais écrit sur le disque de quelque manière que ce soit. Certains systèmes ou antivirus peuvent cependant notifier que le son est en train d'être enregistré.

## Capture de la vidéo par la webcam

Pour enregistrer depuis la webcam, le programme utilise le module **opencv-python**, qui, comme son nom l'indique, est une implémentation python d'OpenCV. Cette implémentation fonctionne sur tous les OS, nous permettant de capturer depuis la webcam sur Windows, Linux ainsi que MacOS.

Cependant, le module opencv-python a été développé de façon à ce que l'on ne puisse pas choisir quelle méthode de sauvegarde doit être utilisée. Il nous est donc impossible de garder les données de la webcam en mémoire vive puisque le module les sauvegarde dans un fichier sur le disque et il n'y a aucun moyen de contourner ce problème. Il est également impossible de stocker ces données dans un fichier temporaire, moins détectable : il doit être sauvegardé à la racine du programme. Pour l'envoyer, nous ouvrons donc le fichier en mode binaire après capture de la webcam et envoyons les octets sur le réseau. Ils sont enregistrés comme pour la capture d'écran et l'enregistrement audio côté serveur.

Sur les ordinateurs portables, la webcam possède la plupart du temps une led qui s'allume lorsqu'elle enregistre, nous empêchant d'exporter des données discrètement. Certains antivirus voire même systèmes d'exploitation peuvent également détecter qu'une application demande à utiliser la webcam et prévenir l'utilisateur.

## Capture du clavier

Pour capturer le clavier, nous avons utilisé le module **tkinter** qui sert à l'origine à faire des interfaces graphiques. Il permet d'enregistrer les appuis de touches lorsqu'elles sont tapées dans des champs de texte.

Cependant, il permet uniquement d'enregistrer les frappes lorsque la fenêtre est focalisée par l'utilisateur, nous empêchant de créer un keylogger. Nous avons donc créé une fausse page de saisie de mot de passe administrateur. Une fois que le mot de passe est entré et validé, il est envoyé sous forme d'octets sur le réseau, et encore une fois, enregistré dans un fichier ouvert en écriture binaire côté serveur.

La détection de la fausse page de mot de passe tient à la méfiance de l'utilisateur. Si ladite page est bien construite, elle peut ressembler complètement à une vraie page de mot de passe et être rendue complètement indétectable.

## Requête des instructions à exécuter

Lorsqu'il est exécuté, le programme crée une connexion avec le serveur et lui demande les instructions à utiliser. Ces instructions sont en python, stockées dans un fichier particulier sur le serveur qui les lit et les envoie au client. À l'aide de la commande **exec**, le client est en mesure d'interpréter directement les instructions qui lui sont envoyées, permettant de lui faire exécuter n'importe quel code python. Nous avons également ajouté une sécurité en cas d'instruction non-valide, le programme passera simplement à l'instruction suivante sans s'arrêter à cause d'une erreur.

Cette fonction exec est très pratique pour rendre le code polymorphe, mais il peut également le rendre beaucoup plus détectable. Tous les langages interprétés possèdent une fonction permettant d'interpréter du code arbitraire, et elles sont toutes réputées pour être mauvaises d'un point de vue sécurité, car elles interprètent absolument tout.

## Envoi et réception des données

La communication entre le client et le serveur se fait à l'aide d'une socket TCP. Le serveur est en attente permanente de communication. Lorsque le client se connecte, le serveur lui envoie ses instructions (comme dit juste au-dessus), puis le client les exécute et envoie le résultat. Comme nous envoyons 4 types de données différents (de l'audio pour l'enregistrement du microphone, de l'image pour la capture d'écran, de la vidéo pour la webcam et du texte pour la saisie du mot de passe), nous utilisons un identifiant (un entier) pour savoir quel type de données est envoyé. La demande d'instruction est également un envoi simple d'identifiant. Une fois que le serveur sait ce qu'il va recevoir et a reçu les données, il appelle la fonction, correspondant au type de données, qui va ensuite se charger de la sauvegarde dans le format adéquat.

L'envoi et la réception sont encore des points sensibles dans le projet. Tout est envoyé en clair donc une personne sniffant le réseau serait en mesure de voir les trames. De plus, tout est envoyé d'un seul coup, entraînant un pic de trafic lors de l'envoi de données. Pour sécuriser le tout, il faudrait chiffrer les données côté client et les déchiffrer côté serveur, et envoyer les données par petits paquets pour limiter l'utilisation du réseau.

## Bilan technique

Au terme de ce semestre de projet, le programme fonctionne et permet la récupération de données depuis un ordinateur client vers un ordinateur serveur. Que ce soit l'audio à travers le microphone, la vidéo avec la webcam, une capture d'écran, ou bien même la récupération du mot de passe administrateur à travers une fausse fenêtre administrateur. Malgré le fait que l'on n'ait pas eu le temps d'améliorer la structure du code qui aurait permis une réutilisation simplifiée du programme en ne prenant en compte que les fonctionnalités souhaitées, toutes les fonctionnalités que l'on avait prévues au début du projet sont fonctionnelles, et même multi-plateforme pour la plupart.

Cependant, l'envoi des données reste détectable pour une personne s'y connaissant. En effet, toutes les données sont pour le moment envoyées en même temps et en clair. Il faudrait donc mettre en place une méthode afin d'envoyer les données récupérées petit à petit que ce soit par intervalle de temps ou seulement lorsqu'une application est en cours d'utilisation. De plus, il faudrait chiffrer les données en utilisant un chiffrement RSA par exemple. Cela empêcherait un individu sniffant le réseau de voir ce que nous envoyons et de potentiellement déduire à quoi servent les différentes trames envoyées. Par ailleurs, nous avons fait le maximum pour que l'application fonctionne que ce soit sur Linux et sur Mac. Mais n'ayant pas souvent de quoi le tester sur Windows, certaines fonctionnalités ne fonctionnent pas encore totalement.

Ces améliorations conceptuelles auraient sans doute nécessité de nombreuses heures de projet supplémentaires pour être développées et n'étaient pas nécessaire au bon fonctionnement de l'application, c'est pourquoi nous ne nous y sommes pas intéressés et avons préféré nous pencher sur la récupération du maximum de données possibles. Notre réalisation est fonctionnelle et pourra servir de base pour permettre, par la suite, à d'autres étudiants de récupérer des données de manière plus déguisée.

Pour permettre à d'autres personnes de réutiliser le programme, celui-ci se devait d'être clairement documenté. Une contrainte que nous estimons avoir respectée au mieux.



## Conclusion

Ce projet a constitué pour nous une première expérience dans le monde de la sécurité informatique. Tout d'abord, il nous a permis de voir à quel point il était simple et rapide de récupérer des données sur un ordinateur sans réelle formation.

Par ailleurs, les réunions avec notre tuteur de projet afin de faire un point sur l'avancée du programme nous ont été très utiles puisqu'elles nous ont permis de comprendre comment fonctionnait un malware afin d'être le moins détectable possible. Nous avons ainsi pu comprendre que le plus long n'était pas d'avoir accès à l'ordinateur et d'en récupérer les données, mais de faire en sorte que le programme ne soit pas détecté, que ce soit par l'antivirus, par le système d'exploitation ou bien même par l'utilisateur.

Enfin, ce projet nous a permis, en autonomie, d'acquérir des compétences en cybersécurité et en conception de programme.

# Lexique

**Malware :** un malware ou logiciel malveillant est un programme développé dans le but de nuire à un système informatique, sans le consentement de l'utilisateur dont l'ordinateur est infecté.

**Cross-platform :** un logiciel cross-platform ou multi-plateforme est un logiciel conçu pour fonctionner sur plusieurs plateformes et dans notre cas sur plusieurs systèmes d'exploitation (Linux, MacOS, Windows).

**Keylogger :** un keylogger ou enregistreur de frappe est un logiciel espion qui espionne l'utilisateur d'un ordinateur. Il peut par exemple enregistrer les touches saisies au clavier.

**Screenshot :** un screenshot ou capture d'écran est une image dont le contenu est celui qui a été affiché à un instant donné sur un écran d'ordinateur, de télévision ou de tout autre dispositif d'affichage.


Cette image est donc semblable à une photo que l'on aurait prise de l'écran à ce même instant en ayant bien pris soin de cadrer uniquement le rectangle d'affichage de l'écran.

# Annexes

## Annexe 1 : Scan du fichier python par VirusTotal

<div><div><div><div><div></div><div>0 / 56</div></div></div><div><div>No engines detected this file</div><div><div>SHA-256191a3d95c106870c50c9ab34844055b8d1146c6d382d5c8091df5ecaf67100be</div><div>File namepayload.py</div><div>File size4.11 KB</div><div>Last analysis2019-05-02 11:06:53 UTC</div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div></div>					<div><div><div><div><div></div><div>Kaspersky</div><div>✓ Clean</div></div><div><div></div><div>Kingsoft</div><div>✓ Clean</div></div><div><div></div><div>Malwarebytes</div><div>✓ Clean</div></div><div><div></div><div>MAX</div><div>✓ Clean</div></div><div><div></div><div>MaxSecure</div><div>✓ Clean</div></div><div><div></div><div>McAfee</div><div>✓ Clean</div></div><div><div></div><div>McAfee-GW-Edition</div><div>✓ Clean</div></div><div><div></div><div>Microsoft</div><div>✓ Clean</div></div><div><div></div><div>NANO-Antivirus</div><div>✓ Clean</div></div><div><div></div><div>Panda</div><div>✓ Clean</div></div><div><div></div><div>Qihoo-360</div><div>✓ Clean</div></div><div><div></div><div>Rising</div><div>✓ Clean</div></div><div><div></div><div>Sophos AV</div><div>✓ Clean</div></div><div><div></div><div>SUPERAntiSpyware</div><div>✓ Clean</div></div><div><div></div><div>TACHYON</div><div>✓ Clean</div></div><div><div></div><div>Tencent</div><div>✓ Clean</div></div><div><div></div><div>TheHacker</div><div>✓ Clean</div></div><div><div></div><div>TotalDefense</div><div>✓ Clean</div></div><div><div></div><div>VBA32</div><div>✓ Clean</div></div><div><div></div><div>ViRobot</div><div>✓ Clean</div></div><div><div></div><div>Yandex</div><div>✓ Clean</div></div><div><div></div><div>Zillya</div><div>✓ Clean</div></div><div><div></div><div>ZoneAlarm</div><div>✓ Clean</div></div><div><div></div><div>Zoner</div><div>✓ Clean</div></div><div><div></div><div>Acronis</div><div>Unable to process file type</div></div><div><div></div><div>Alibaba</div><div>Unable to process file type</div></div><div><div></div><div>CrowdStrike Falcon</div><div>Unable to process file type</div></div><div><div></div><div>Cybereason</div><div>Unable to process file type</div></div><div><div></div><div>Cylance</div><div>Unable to process file type</div></div><div><div></div><div>eGambit</div><div>Unable to process file type</div></div><div><div></div><div>Endgame</div><div>Unable to process file type</div></div><div><div></div><div>Palo Alto Networks</div><div>Unable to process file type</div></div></div></div></div>				
<div><div>Detection</div><div>Details</div><div>Relations</div><div>Behavior</div><div>Community</div></div>					<div><div><div><div><div></div><div>Ad-Aware</div><div>✓ Clean</div></div><div><div></div><div>AegisLab</div><div>✓ Clean</div></div><div><div></div><div>AhnLab-V3</div><div>✓ Clean</div></div><div><div></div><div>ALYac</div><div>✓ Clean</div></div><div><div></div><div>Antiy-AVL</div><div>✓ Clean</div></div><div><div></div><div>Arcabit</div><div>✓ Clean</div></div><div><div></div><div>Avast</div><div>✓ Clean</div></div><div><div></div><div>Avast Mobile Security</div><div>✓ Clean</div></div><div><div></div><div>AVG</div><div>✓ Clean</div></div><div><div></div><div>Avira</div><div>✓ Clean</div></div><div><div></div><div>Babable</div><div>✓ Clean</div></div><div><div></div><div>Baidu</div><div>✓ Clean</div></div><div><div></div><div>BitDefender</div><div>✓ Clean</div></div><div><div></div><div>Bkav</div><div>✓ Clean</div></div><div><div></div><div>CAT-QuickHeal</div><div>✓ Clean</div></div><div><div></div><div>ClamAV</div><div>✓ Clean</div></div><div><div></div><div>CMC</div><div>✓ Clean</div></div><div><div></div><div>Comodo</div><div>✓ Clean</div></div><div><div></div><div>Cyren</div><div>✓ Clean</div></div><div><div></div><div>DrWeb</div><div>✓ Clean</div></div><div><div></div><div>Emisoft</div><div>✓ Clean</div></div><div><div></div><div>eScan</div><div>✓ Clean</div></div><div><div></div><div>ESET-NOD32</div><div>✓ Clean</div></div><div><div></div><div>F-Prot</div><div>✓ Clean</div></div><div><div></div><div>F-Secure</div><div>✓ Clean</div></div><div><div></div><div>FireEye</div><div>✓ Clean</div></div></div></div></div>				

## Annexe 2 : Scan de l'exécutable Windows (.exe) par VirusTotal

<div>  <div> 2 engines detected this file </div> <div> <div>2 / 69</div> </div> </div> <div> <div>SHA-256</div> <div>21b48def0baea5e8b37e11332dc1177b1772c0591c654c5cae68e261518eae9c</div> </div> <div> <div>File name</div> <div>payload.exe</div> </div> <div> <div>File size</div> <div>60.64 MB</div> </div> <div> <div>Last analysis</div> <div>2019-05-02 14:34:07 UTC</div> </div>	
Detection	Details
Kaspersky	HEUR:Trojan.Win32.Generic
ZoneAlarm	HEUR:Trojan.Win32.Generic
Acronis	Clean
Ad-Aware	Clean
AegisLab	Clean
AhnLab-V3	Clean
Alibaba	Clean
ALYac	Clean
Antiy-AVL	Clean
Arcabit	Clean
Avast	Clean
Avast Mobile Security	Clean
AVG	Clean
Avira	Clean
Babable	Clean
Baidu	Clean
BitDefender	Clean
Bkav	Clean
CAT-QuickHeal	Clean
ClamAV	Clean
CMC	Clean
Comodo	Clean
CrowdStrike Falcon	Clean
Cybereason	Clean
Cyren	Clean
DrWeb	Clean
eGambit	Clean
Emsisoft	Clean
Endgame	Clean
eScan	Clean
ESET-NOD32	Clean
F-Secure	Clean
FireEye	Clean
Fortinet	Clean
GData	Clean
Ikarus	Clean
Jiangmin	Clean
K7AntiVirus	Clean
K7GW	Clean
Kingsoft	Clean
Malwarebytes	Clean
MAX	Clean
McAfee	Clean
McAfee-GW-Edition	Clean
Microsoft	Clean
NANO-Antivirus	Clean
Palo Alto Networks	Clean
Panda	Clean
Qihoo-360	Clean
Rising	Clean
SentinelOne	Clean
Sophos AV	Clean
Sophos ML	Clean
SUPERAntiSpyware	Clean
TACHYON	Clean
Tencent	Clean
TheHacker	Clean
TotalDefense	Clean
Trapmine	Clean
TrendMicro-HouseCall	Clean
Trustlook	Clean
VBA32	Clean
VIPRE	Clean
ViRobot	Clean
Webroot	Clean
Yandex	Clean
Zillya	Clean
Zoner	Clean