

BOTTER PROJECT

Neil AMARI

Julien OPPLIGER



Sommaire

I. Présentation générale

II. Fonctionnement du botnet

IV. Mise en place d'une attaque

V. Conclusion

Annexe : Liste des fonctions utilisées dans le code

I. Présentation générale

Notice légale

Étant donnée la nature de notre projet, nous devons commencer par préciser que notre programme est uniquement destiné à un usage personnel et n'a en aucun cas vocation à être diffusé.

Loi n° 78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés

Article 2

Modifié par Loi n°2004-801 du 6 août 2004

La présente loi s'applique aux traitements automatisés de données à caractère personnel, ainsi qu'aux traitements non automatisés de données à caractère personnel contenues ou appelées à figurer dans des fichiers, à l'exception des traitements mis en oeuvre pour l'exercice d'activités exclusivement personnelles, lorsque leur responsable remplit les conditions prévues à l'article 5.

Article 5

Modifié par Loi n°2004-801 du 6 août 2004

Sont soumis à la présente loi les traitements de données à caractère personnel :

1° Dont le responsable est établi sur le territoire français. Le responsable d'un traitement qui exerce une activité sur le territoire français dans le cadre d'une installation, quelle que soit sa forme juridique, y est considéré comme établi ;

Définitions

Un botnet est une forme particulière de virus informatique. Son rôle est d'effectuer une attaque sur un ensemble de machines infectées appelés « bots ». L'action conjointe d'un tel réseau de machines infectées permet de multiplier la puissance d'une attaque par le nombre de bots connectés.

Les attaques peuvent être d'un genre très divers : déni de service, déchiffrement, collecte d'informations, et bien d'autres encore dont le point commun est de pouvoir être effectué en réseau. Celles-ci peuvent être redoutables à grande échelle.

Objectifs

Notre botnet se concentre sur la collectes d'informations personnelles, celles notamment qui permettent une identification. Cela comprend les adresses de messageries, les identifiants et pseudonymes, les mots de passes et les informations secrètes servant pour la récupération de mot de passe.

Il se distingue par la volonté de dissimuler celui qui récolte les données. La récolte d'informations personnelles étant illégale dans la quasi-totalité des pays du monde, il est naturel qu'un pirate ait comme première préoccupation de se soustraire aux mains et aux yeux de la justice. C'est à ce besoin, somme toute légitime, que répond notre Botnet !

II. Fonctionnement du botnet

Principes de fonctionnement

Notre botnet repose sur trois principes qui régissent son fonctionnement :

-échanges décentralisés :

Afin de correspondre à notre désir de rendre impossible l'identification du pirate, nous avons postulé qu'un échange décentralisé des données collectées était le moyen le plus efficace d'y parvenir. La solution opposée, à savoir passer par l'intermédiaire d'un serveur central, nous paraissait trop risqué puisqu'un observateur extérieur n'aurait qu'à concentrer son analyse sur cet unique serveur pour trouver le ou les individus qui récupèrent les données.

Ainsi, les données une fois récupérées sont envoyées, stockées temporairement par les autres machines et renvoyées à d'autre par elles. Au terme d'un certain laps de temps, toutes les machines auront reçu, conservées et renvoyées toutes les données du réseau.

C'est de ce principe d'échange décentralisé que notre botnet tire l'anonymat du pirate. Étant donné que chaque machine reçoit chaque données, toutes sont potentiellement des pirates aux yeux d'un observateur extérieur.

-identité des traitements :

Le principe d'identité des traitements est le deuxième moyen dont nous disposons pour camoufler les traces de notre pirate.

Il consiste à faire réaliser à notre client « pirate » les mêmes opérations que les clients « victimes ». Du moins est-ce le cas pour toutes les opérations en réseau, soit celles qui sont observables depuis l'extérieur. Le pirate envoyant ses propres données (factices bien entendu), recevant celles des autres et les renvoyant aux autres machines, exactement au même titre que toutes les machines victimes, celui-ci se comporte, du point de vue d'un analyste réseau, de la même façon que tous les autres bots, empêchant de fait son identification.

-conservation temporaire des données d'autrui :

Les données collectées sur une machine par le botnet sont conservées durablement sur la machine en question. C'est le seul fichier qui soit dans cette situation. Les fichiers reçus par le réseau en provenance des autres machines sont conservés temporairement. Deux raisons nous ont poussées à cette conservation temporaire :

-Elle permet d'une part de pouvoir renvoyer les fichiers reçus, permettant ainsi la circulation de toutes les données, y compris celles des bots qui ne sont pas connectés, limitant ainsi les pertes de données.

-Elle contribue d'autre part au camouflage du pirate, puisque du point de vue légal, la sauvegarde, même temporaire de données personnelles récupérées frauduleusement constitue une infraction. Ainsi, toutes les victimes sont potentiellement coupables, tant que l'installation du botnet à l'insu de leur plein gré n'a pas été démontré. Cela permet de complexifier terriblement les procédures judiciaires, et ainsi laisser une marge de manœuvre au pirate pour l'effacement de ses traces.

Fonctionnement technique

Quelles machines peuvent être infectées par le botnet ?

Pour qu'une machine soit infectée par le botnet, il faut qu'elle remplisse quelques conditions préalables :

-Qu'elle ait pour système d'exploitation un système d'exploitation Microsoft Windows 98 ou une version ultérieure. L'API Winsock2 que nous utilisons ne fonctionne pas nativement sur des versions plus ancienne.

-Que la machine soit destinée à un usage non technique, qu'il soit personnel ou professionnel. En effet même si le botnet peut techniquement s'exécuter sur des machines Windows Server, la récolte d'information qu'il effectue sur les données de navigation n'est pas pertinente pour ce type d'installation.

-Qu'elle dispose d'un accès au réseau internet. Ceci est nécessaire à la fois à son infection et à la récolte de données. Cependant, une fois les données récoltées et échangées au moins une fois, celles-ci seront échangées par les autres bots et acquerront ainsi son autonomie.

Comment une machine est ajouté au réseau botnet ?

Une fois la procédure de la partie « Mise en place d'une attaque » de cette documentation a été réalisée, la victime a installé et exécuté pour la première fois le botnet. Celui-ci va alors se faire connaître auprès des autres machines du réseau infecté en contactant un serveur de map qui va ajouter son adresse ip à la liste des bots disponibles et va le marquer comme connecté. Le serveur de map va ensuite envoyer la map du réseau, contenant toutes les adresses ip connectées à tous les bot de ladite liste. L'envoi de la map du réseau est réalisée à intervalles régulière afin d'être à jour sur les connexions et déconnexions des différentes machines.

Le serveur de map n'est-il pas un serveur central ?

Par serveur central, nous entendons ici un serveur qui centralise les données. C'est bien ce type de serveur qui constitue un risque pour l'identification du pirate. Le serveur de map dont il est question ici s'occupe seulement du routage, les données tant qu'à elles sont directement échangées entre les machines infectés. De ce fait, toutes les machines se comportent de la même façon à l'égard du serveur de map,

ce qui respecte les deux principes énoncés précédemment d'identité des traitements et d'échanges décentralisés.

De plus le serveur de map ne conserve aucun log et change régulièrement l'ordre des adresses ip, empêchant ainsi de remonter à la première ip enregistrée.

Comment sont récupérées les données ?

Les données sont récoltées grâce à un sniffer dont le rôle est d'écouter sur le port 80 toutes les entrées et sorties à la recherche de certains mots clés relatifs aux données que nous souhaitons récupérer. Si dans le champ de données d'un paquet TCP échangé sur le port 80 est contenu le mot clé « password » par exemple, alors toutes les paires clé/valeur du champ de données vont être récupérées et stockées sur un fichier.

Comment sont écrites les données ?

Les fichiers sur lesquels sont stockés les données ont une extension « .ter », qui est une extension propre à notre botnet, qui stock les données de manière très compacte.

Le format d'écriture des données est le suivant :

```
i=XXX.XXX.XXX.XXX;u=www.test.io;d=[key=value;key=value;];
```

« i » est le diminutif de « ip ». Il est présent une fois par fichier et identifie la machine que concerne tel ou tel fichier de données.

« u » est le diminutif de « url ». Il est présent autant de fois que la victime consulte un site dont des mots clés ont été extraits.

« d » est le diminutif de « data ». Il se présente sous la forme d'un tableau qui contient l'ensemble des paires clés/valeurs récupéré dans le champ de données d'un paquet TCP. Il y a autant de tableau de données qu'il y a de paquets contenant des mots clés. Les tableaux de données sont placés dans le fichier les uns à la suite des autres après une url afin d'y être rattachés.

Comment sont échangées les données ?

Une fois les données collectées et inscrites sur un fichier, la phase d'envoi des fichiers commence. Le botnet se réfère à la map du réseau qu'il a reçus du serveur de map pour envoyer toutes les X secondes des données à l'ip suivante dans la liste. Dès que l'ip à contacter est sélectionné une communication s'établit selon le schéma suivant :

1. PC source demande le fichier de PC destination
2. PC destination envoie son fichier à PC source
3. PC destination envoie la liste des autres fichiers qu'il possède à PC source
4. PC source choisie X fichiers parmi ceux qu'il n'a pas eu depuis longtemps
5. PC destination envoie les fichiers demandés par PC source

6. Fin de la communication

Comment le pirate est camouflé ?

Le pirate est camouflé grâce aux principes énoncés dans le chapitre précédent. Le fait qu'il se comporte en tous points comme les autres bots au niveau du réseau est la garantie de discrétion du pirate. La seule différence qu'il existe entre le pirate et la victime se passe au niveau local, empêchant ainsi l'identification. Seul le où les pirates enregistrent durablement les données qui sont stockées temporairement par les victimes. La conséquence est que pour qu'un observateur extérieur mette la main sur le pirate, il lui faut d'abord analyser les disques durs de toutes les machines du réseau.

IV. Mise en place d'une attaque

Mise en place du serveur de map

Une fois le dossier du botnet en votre possession, la première chose à faire est d'installer le serveur de map sur un ordinateur dédié exclusivement à cette fonction. L'idéal est de l'héberger sur un serveur offshore, dans un pays hors de l'union européenne où la protection des données est respecté comme en Islande.

Pour se faire il suffit juste de déplacer « server.exe » dans le dossier « démarrage » de votre serveur Windows, ce qui aura pour effet de lancer le serveur automatiquement à chaque redémarrage. Il suffit ensuite de le laisser tourner afin que les bots puissent s'y connecter.

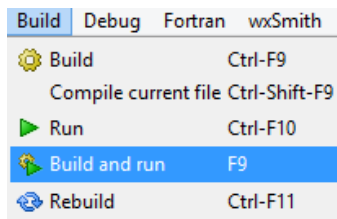
Enfin il faut noter l'adresse ip de ce serveur afin de compiler avec la bonne destination le botnet, ce que nous verrons dans le chapitre suivant. Pour cela ouvrez un terminal « powerShell » ou « cmd » et tapez la commande : `ipconfig`

Notez ensuite l'adresse ipv4 qui vous sera donnée.

Recompiler le botnet

Afin que tous les bots se connecte d'eux-même au serveur de map sans l'intervention de la victime, l'adresse du serveur doit être inscrite dans le code, ce qui implique qu'il faille recompiler le code avec l'adresse que vous avez récupéré précédemment.

Pour ce faire téléchargez l'ide code-block à cette adresse. Rendez vous dans le sous-dossier « src » de votre dossier « BotTer », puis ouvrez le fichier « bot.cbp » avec code-block. Une fois le logiciel, voyez l'arborescence à gauche et cliquez sur le dossier « Headers ». Ouvrez ensuite le fichier « mapaddr.h » et modifiez l'adresse ip existante par celle de votre serveur de map.



Une fois ceci fait , rendez vous dans le menu « Build » et appuyez sur le bouton « Build and run ».

Ouvrez ensuite de la même manière le projet « pirate.cbp » et compilez le immédiatement. Les deux projets partageant les mêmes fichiers à quelques petites différences près, les modifications ont aussi été prises en compte dans ce projet.

Code-Block a généré dans votre dossier « src » un fichier « bin » dans lequel se trouve un dossier « release ». Ouvrez les, vous y trouverez « bot.exe » et « pirate.exe ».

Le programme « pirate.exe » vous est destiné, il va vous permettre de choisir vous-même les données que vous souhaitez envoyer aux autres bots, ainsi que de récupérer les données, comme nous le verrons peu après.

Le programme « bot.exe » est celui qui va permettre d'infecter les victimes. C'est ce programme que vous devrez leur envoyer, surtout pas l'autre.

Configurer le bot pirate

Maintenant que vous disposez de votre client « pirate.exe », vous devez écrire un fichier de données sur le modèle du format donné dans la partie du chapitre précédent : « Comment sont écrites les données ? ». Un fichier « data.txt » est donné comme exemple dans le dossier que vous avez téléchargé. Nous vous invitons vivement à changer les valeurs et à en mettre d'autres qui soient réalistes, sans quoi votre identification sera immédiate et le botnet dénué d'intérêt. Ne faites pas non plus d'erreurs de syntaxe dans la rédaction de ce fichier sans quoi vous serez également identifiable.

Une fois que vous avez rempli ce fichier d'informations factices (nous vous invitons à noyer les fausses informations critiques au sein de vraies informations insignifiantes), placez le fichier dans le même dossier que « pirate.exe »

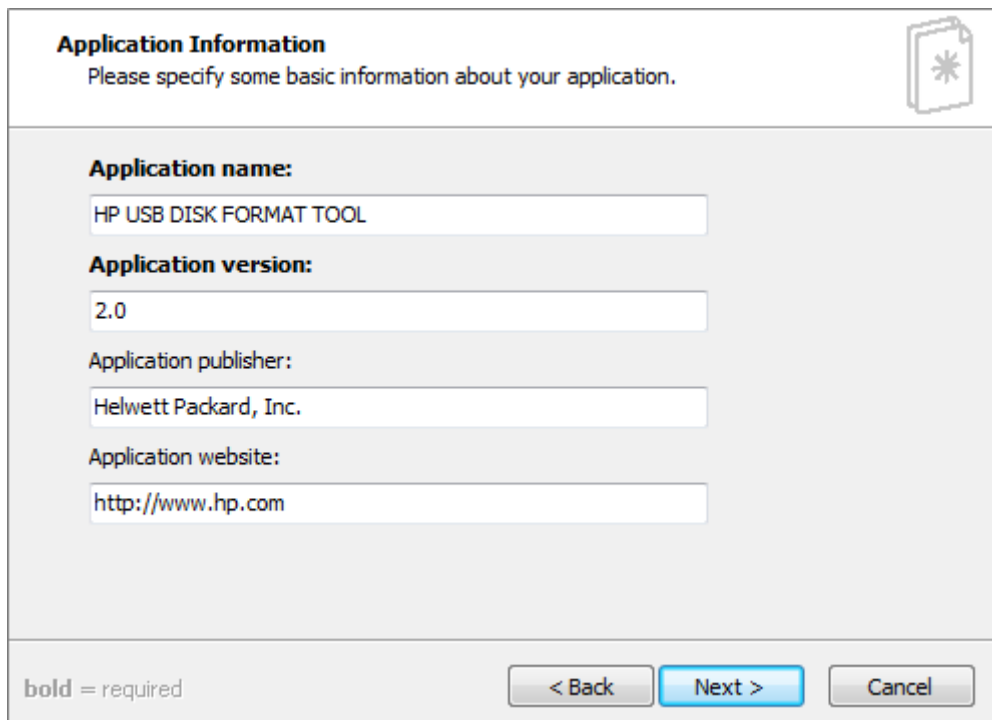
Diffuser le botnet

Pour que l'attaque fonctionne, il faut que des individus téléchargent et installent notre botnet. Nous vous proposons ici une méthode simple à mettre en place, vous êtes cependant libre de trouver la méthode de diffusion qui vous paraîtra la plus efficace.

Pour cela nous allons devoir télécharger et installer le logiciel Inno Setup. Il nous faut également un utilitaire quelconque sans installation, pour l'exemple nous prendrons HP USB Disk Storage Format Tool qui sert à formater efficacement des clés USB, il est très léger et ne demande pas d'installation pour être utilisé.

L'objectif va être de créer avec Inno Setup une fausse installation de HP USB Disk, qui installera en douce notre « bot.exe » dans le dossier « démarrage » de Windows.

Ouvrez Inno Setup Compiler. Cliquez sur « create a new script file using the Script Wizard » et appuyez sur « next ». Tâchez ensuite de rentrer des informations qui semblent légitimes :




The screenshot shows the 'Application Information' dialog box in Inno Setup. It contains the following fields:

- Application name:** HP USB DISK FORMAT TOOL
- Application version:** 2.0
- Application publisher:** Hewlett Packard, Inc.
- Application website:** http://www.hp.com

At the bottom, there is a legend indicating that bold text means 'required'. There are three buttons: '< Back', 'Next >' (highlighted in blue), and 'Cancel'.

Appuyez sur « Next » et cochez la case « The application doesn't need a folder ». Indiquez ensuite le chemin vers votre programme légitime et ajoutez y le chemin de « bot.exe ». Appuyez ensuite sur « Next » jusqu'à la fin de l'installation.

Vous devriez être confronté à un script Inno Setup. A la fin de ce script se trouve une section **[Files]** dans laquelle se trouvent les sources et destinations des fichiers à installer. Au niveau de la ligne qui a « bot.exe » pour source, modifiez la destination « DestDir », remplacez « {win} » par « {userstartup} ». Cela aura pour effet d'installer « bot.exe » dans le dossier « Démarrage » de la victime, lançant ainsi le botnet à chaque démarrage de l'ordinateur.

 Compilez et enregistrez ensuite votre script en appuyant sur « run ».

Une fois votre faux installeur créé, essayez-le. S'il marche, importez-le sur un site de téléchargement direct ou bien sous forme de torrent sur un site spécialisé, mettez enfin le lien ainsi obtenu sur un forum où les personnes sont susceptibles d'être intéressées par le programme légitime. Voilà votre botnet prêt à infecter la terre entière !

Réceptionner les données

Pour récupérer les données rien n'est plus simple, il suffit de démarrer « pirate.exe », de sélectionner l'option « Démarrer » et de laisser tourner le programme autant de temps que l'on veut accumuler des données. Celles-ci sont enregistrées au format « .ter ». Afin que celles-ci soient lisibles nous avons mis à votre disposition une interface web php qui permet de lire ces fichiers. Vous devez bien sûr installer cette interface web sur un serveur. Vous pouvez télécharger wamp [ici](#).

V. Conclusion

L'attaque que nous avons conçus présente de nombreux avantages, pourvu que nous ayons consciences de ses limites et ainsi de ce qu'elle exige pour être mené à bien dans les meilleures conditions.

Comme nous l'avons vu dans les chapitres précédents, ce qui distingue le pirate de la victime c'est la conservation des données au niveau local. De plus, la liste des adresses ip connectées (et donc des machines infectées) est régulièrement envoyé à tous les membres du réseau. Il faut bien avoir à l'esprit que quiconque utilise ce botnet fait parti de la liste des suspects potentiel. Notre attaque tire son efficacité de la complexité qu'elle est en mesure de générer, cependant cette complexité dépend directement du nombre et de la localisation des machines infectés.

En effet la contre-mesure que peut mettre en place la police peut être de jouer la carte de l'exhaustivité et de faire une perquisition massive de toutes les machines infectés dont ils connaissent la liste grâce au serveur de map. Cela leur est très simple tant que les victimes sont en nombre restreint et demeurent dans l'union européenne. Si par contre les victimes se comptent en centaine de milliers et sont répartis sur différents continents aux intérêts diplomatiques contraires, et bien la charge de travail et les difficultés des relations internationales rendront le où les pirates statistiquement inatteignablent.

Dans le cas où quelqu'un voudrait, grâce à ce botnet s'essayer à braver la justice, la sanction qu'il encourrait serait terrible étant donnée la nécessité d'internationaliser l'attaque et ainsi de violer les lois nationales de tous les pays dans lesquels le virus a été implanté, en plus d'encourir des sanctions internationales.

Voici un lien énonçant la loi française en vigueur à ce sujet :

<https://www.cnil.fr/fr/loi-78-17-du-6-janvier-1978-modifiee>

Annexe : Liste des fonctions

Display.c

`void printIntro()` ; Affiche la bannière du Bot.

`void viewMain()` ; Affiche le menu principale.

`void viewMap()` ; Affiche la liste des machines zombies du réseau.

`void printError(int errorCode)` ; Affiche l'erreur associé au code correspondant.

File.c

`int initFile(FILE** file, const char* dest);` Crée et initialise le fichier de ses propres données.

`int writeFile(char* dest, char* data, unsigned int size);` Ecrit les données reçues par les autres bots.

`int startBlock(FILE* file, char *url);` Ecrit le début d'un bloc de données.

`int addData(FILE* file, char *key, char *value);` Ecrit les données du bloc.

`int endBlock(FILE* file);` Ecrit la fin d'un bloc de données.

`unsigned int getFileContent(char* ip, char** content);` Lis les données du fichier personnel.

map.c

`void getMap();` recupere la liste des machines zombies depuis le serveur.

Routeur.c

`void removeSubstring(char* str, const char *toremove);` Supprime une sous-chaine d'une chaine.

`int initRouter();` Liste les fichiers disponible dans le répertoire courant.

`void listener ();` Ecoute toutes le connexions entrante et démarre le protocole de communication.

`void sender ();` Envoi a rythme régulier les données aux autres bots du réseau.

`char* selectAddress();` Sélectionne une adresse a contacter.

`void listenerHandler(ARG *arg);` Exécute le protocole de réception.

`void senderHandler(SOCKET connexion, char conn_ip[16]);` Exécute le protocole d'envoi.

sniffer.c

`int initSniffer();` Cree un socket d'écoute du réseau et lance l'analyse.

`void startSniff(SOCKET sniffer);` lance la réception des paquets.

`void processPacket(char* Buffer, int size);` Prépare les données au traitement.

`int checkRequest(char* Buffer, int size);` Converti les données brut en chaînes de caractères.

`char * findKeywords(char* tcpdata, int size);` Récupère les données pertinentes.

`int checkWords(char * data);` Vérifie l'existence d'un mot clé dans la chaîne.

`void separatePair(char* request);` Sépare les paires clés valeurs de la chaîne et envoie le résultat à l'écriture de fichier.

Chained list.c

`int addToList (LINK **list, char* content);` Ajoute un maillon à la fin d'une liste chaînée.

`int removeFirst (LINK **list);` Retire le premier élément d'une liste.

`unsigned int getListSize (LINK *list);` Retourne le nombre de maillons de la liste.

`int itemExist (LINK *list, char* item);` Vérifie si une donnée est présente dans la liste.